2003

Design of power efficient multicast algorithms for sparse split WDM networks

Kavitha Devi Buddharaju
Louisiana State University and Agricultural and Mechanical College, kbuddh1@lsu.edu

DESIGN OF POWER EFFICIENT MULTICAST ALGORITHMS FOR SPARSE SPLIT WDM NETWORKS

A Thesis

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
In partial fulfillment of the
Requirements for the degree of
Master of Science of Electrical Engineering

In

The Department of Electrical Engineering

By
Kavitha Devi Buddharaju
Bachelor of Engineering in Electrical Engineering
Andhra University, Viskhapatnam, 2001
December 2003.

Acknowledgements

I would like to thank my advisor Prof. Ahmed El. Amawy for making this work possible. No words of thanks are enough for his mentorship, guidance, support and patience. I must acknowledge the freedom he gave me in pursuing research topics that I found interesting. His keen observations and tremendous experience helped me a great deal in defining path for my research. I have learned a lot from him in the last couple of years both as a researcher and as a person.

I would like to thank Prof. J.Ramanujam and Prof. Hsiao-Chun Wu, for taking their valuable time out and agreeing to be on my thesis committee. I would like to thank my entire family for their support and love throughout. Rana thanks a lot for all the help.

No acknowledgement is complete without mentioning the friends I made at LSU. I will always remember them for making my stay away from home a most memorable one. Thank you one and all for always being there. Finally, I would like to thank my best friend and husband Lalit, without whose love and support none of this would have been possible.

Table of Contents

List of Figures

Abstract

Recent years witnessed tremendous increase in data traffic as new Internet applications were launched. Optical networks employing recent technologies such as DWDM and EDFA`s emerged as the most prominent and most promising solutions in terms of their ability to keep with the demand on bandwidth. However for a class of applications bandwidth is not the only important requirement, These applications require efficient multicast operations. They include data bases, audio/video conferencing, distributed computing etc.

Multicasting in the optical domain however has its own unique set of problems. First, an optical signal can be split among the outputs of a node but the power due to splitting can be significantly reduced. Second, the hardware for split nodes is relatively expensive and therefore we cannot afford to employ it at every node. Third, there are other sources of losses such as attenuation losses and multiplexing /de-multiplexing losses.

This thesis deals with the important issue of *Power Efficient* multicast in WDM optical networks. We report three new algorithms for constructing power efficient multicast trees and forests. Our algorithms are the first to take into account all possible sources of power losses while constructing the trees.

We utilize the techniques of backtracking and tree pruning judiciously to achieve very power efficient multicast trees. The first two algorithms use modified versions of the shortest path heuristic to build the tree. The third algorithm however, uses a novel concept and considers power at every tree building step. In this algorithm, the order of

inclusion of destination nodes into the tree is based on the power distribution in the tree and not distance. All three algorithms prune the trees if the power levels at the destinations are not acceptable.

The performance of these three algorithms under several constraints is studied on several irregular topologies. All three algorithms reported in this work produce significant improvements in signal strength at the set of destinations over the existing multicast algorithms. Numerical results show that our third algorithm outperforms the first two algorithms as well as the existing multicasting algorithms.

Chapter 1

Introduction

One of the pressing issues in the networking industry today is the tremendous demand for bandwidth. This continuous demand for networks of higher bandwidths at low costs can only be met efficiently with *optical networks*. The optical fiber provides an excellent medium for transfer of huge amounts of a data (nearly 50 terabits per second [11]). Besides the high bandwidth, optical fibers also have very low bit error rates, low power requirement and low signal distortion. Optical networks are more secure than their copper counterparts. It is extremely difficult to tap into an optical signal secretly without detection. All the above-mentioned factors make optical networks the networks of the future.

The relatively high speed of optical networks is attributed to the fact that they convert data to photons. These photons unlike electrons weigh less and do not carry a charge. So they are not affected by the stray photons outside the fiber. Light also has higher frequencies and hence shorter wavelengths. Thus smaller lengths of fiber could contain more bits compared to the same length of copper.

Networks are usually classified into three generations depending on the physical technology employed. The first generation networks used copper-based technologies (e.g. Ethernet). Second generation networks used a combination of copper and optical fibers. However, these networks still perform the switching of data in the electronic domain while the transmission of data is done in the optical domain. Third generation networks are those networks which employ *Wavelength Division Multiplexing* (WDM) technology.

Both the transmission and the switching of data are done in the optical domain here. All Optical networks are not yet feasible due to many technological challenges.

As fiber replaced copper, a variety of optical networks came into existence. Examples include the high performance parallel interface, FDDI (Fiber Distributed Data Interface) which uses dual fiber optic token rings and provides 100-200 megabits per second, SONET (Synchronous Optical Network in North America), etc. SONET is a standard for optical telecommunications transport formulated by the Exchange Carriers Standards Association (ECSA) for the American National Standards Institute (ANSI), which sets industry standards in the U.S. for telecommunications and other industries [21]. This standard defines a hierarchy of interface rates that allow data streams at different rates to be multiplexed. SONET establishes Optical Carrier (OC) levels from 51.8 Mbps (about the same as a T-3 line) to 40 Gbps. With the implementation of SONET communication carriers throughout the world can interconnect their existing digital carrier and fiber optic systems [11]. OC-768 achieves the highest realizable electronic speed of 40 Gbps.

1.1 Optical Transmission System

Optical networks employ a fiber-optic system which is similar in certain ways to the copper wire system that fiber-optics is replacing. The difference is that fiber-optics use light pulses to transmit information down fiber lines instead of using electronic pulses to transmit information down copper lines. Figure 1.1 gives a better idea of the method of transmission.

Here at one end of the system is a transmitter, which accepts electrical signals as input. It then processes and translates that information into equivalently coded light pulses. A light-emitting diode (LED) or an injection-laser diode (ILD) can be used for generating the light pulses. Using a lens, the light pulses are funneled into the fiber-optic medium where they travel down the line. Light pulses move down the fiber-optic line using a principle known as total internal reflection. “This principle of total internal reflection states that when the angle of incidence exceeds a critical value (which is dependent on the refractive index of the two mediums), light cannot get out of the glass; instead, the light bounces back in”[11]. So if the light signal is made to be incident at an angle less than the critical angle, it keeps getting reflected as shown in Figure 1.2 till it reaches the other end of the fiber. At that end the receiver receives the light signal and converts it back to an electrical signal using an optical detector [11]. So, we essentially have a unidirectional transmission system which accepts an electrical signal, converts and transmits it by light pulses through the fiber medium and then reconverts the light pulse to an electrical signal at the receiving end.

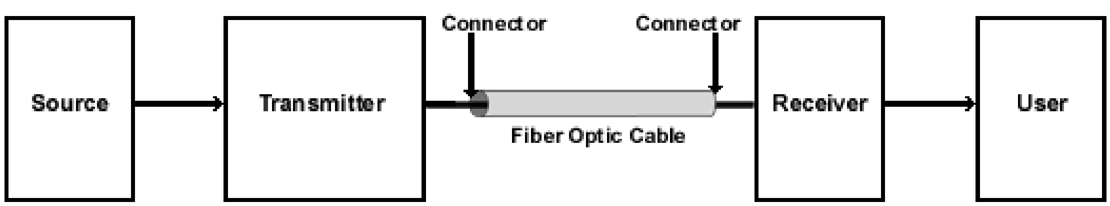


Figure 1.1 Optical transmission system.

1.2 Optical Fiber Principles

Before we go into the actual details of WDM let us discuss the principles underlying fiber transmission. Basically, a fiber optic cable is composed of two concentric layers known as the core and the cladding.

Figure 1.3 illustrates these two layers. The core and cladding have different indices of refraction with the core having n_1 and the cladding n_2 ($n_1 < n_2$) [25]. Light is sent through the core. The fiber optic cable has an additional protective coating around the cladding known as the jacket. The jacket usually consists of one or more layers of polymer. It protects the core and cladding from shocks that might affect their optical or physical properties.

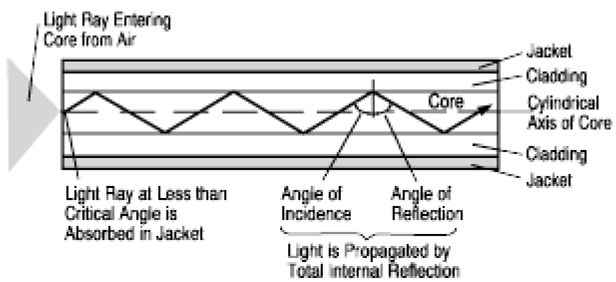


Figure 1.2 Propagation of light ray down a fiber optic cable.

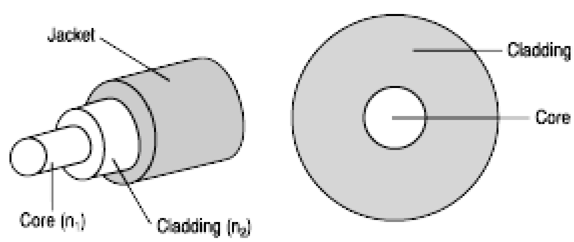


Figure 1.3 Three dimensional view of the fiber optic cable.

A ray of light incident on the core-cladding interface at an angle less than the critical angle will get reflected repeatedly till it reaches the other end of the fiber. There are two modes of fiber propagation known as the multi-mode and single-mode. The single-mode fiber optic cable provides better performance but at a higher cost.

The multimode fiber has a graded refractive index profile, due to which many rays of light can bounce at different angles. Each ray is said to have a different mode, hence, the name multimode fiber. This type of fiber is used in premise data communication applications [25]. This method however may cause the rays to interfere with each other. If a stepwise refractive index is used, the fiber will act like a waveguide and the light will travel in a straight line along the center axis of the fiber. Such fibers are known as the single mode fibers. A single mode fiber has lower attenuation and less time dispersion. However they are more expensive than the multimode fiber. These fibers are used mainly in Wide Area Networks with premises data communications [25].

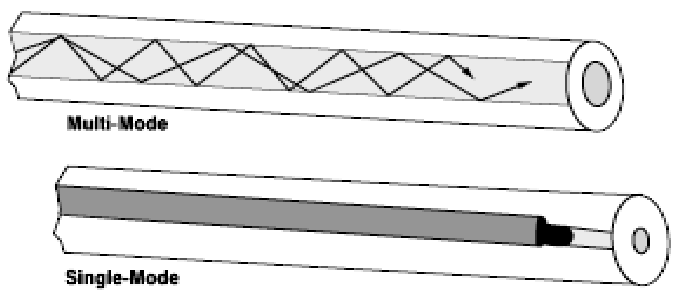


Figure 1.4 Different modes of fiber propagation.

1.3 Wavelength Division Multiplexing

Until two decades ago optical fiber communications was mainly confined to transmitting a single optical channel. Because of the fiber attenuation involved this signal had to be detected and regenerated at regular intervals. Such regenerations required heavy optoelectronic conversions and hence could handle only a single wavelength. The invention of Erbium doped fiber amplifiers (EDFA's) simplified this problem. With the help of these amplifiers the fiber can now accommodate much more bandwidth than the traffic on a single wavelength. To increase the system capacity several different independent wavelengths can be transmitted simultaneously down the fiber. Therefore, the intent of WDM was to develop a multiple-lane highway, with each lane representing data traveling on a different wavelength. Thus, a WDM system enables the fiber to offer very substantial bandwidth on a single fiber. In theory a single fiber can carry up to 50Tbps.

1.3.1 Basic Operation of WDM System

In a simple WDM system, several laser sources are used. Each laser emits light at a different wavelength and all different wavelengths are multiplexed together onto a single optical fiber [11]. The multiplexed signal is transmitted through a high-bandwidth optical fiber, and is de-multiplexed at the receiving end by distributing the total optical power to the output ports. Each of the several receivers selectively recovers only one wavelength by using a tunable optical filter

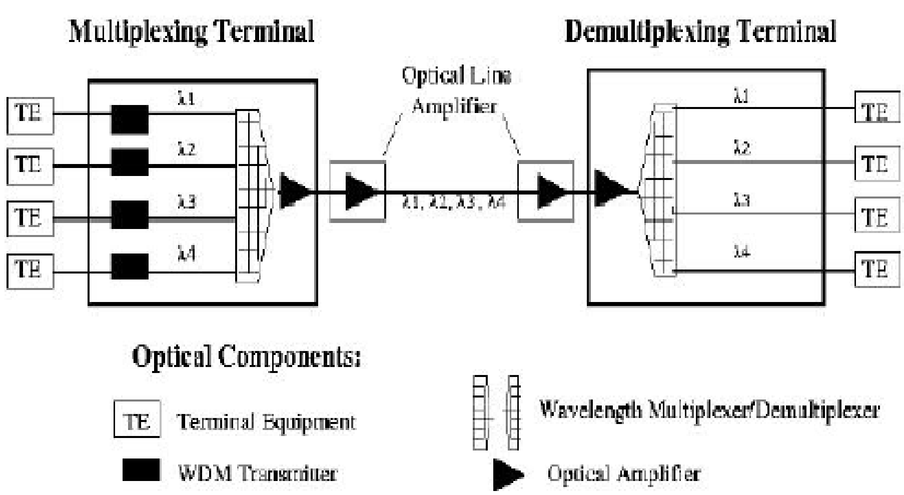


Figure 1.5 Wavelength division multiplexing.

1.3.2 Basic Components of WDM System

The basic components which comprise a WDM system are described in this section.

1.3.2.1 Optical Amplifiers

The commercial development of WDM networks is made possible by the development of EDFA’s (Erbium doped Fiber Amplifiers) which provide a way to optically amplify all the wavelengths at the same time, regardless of their individual bit rates, modulation schemes or power levels [23]. EDFAs are as the name suggests, are silica based optical fibers that are doped with erbium. The element erbium (Er_{68}) boosts the power of wavelengths and eliminates the need for regeneration. Before EDFA’s were invented electronic regenerators were used. The optical signal is converted to an electronic form, amplified and then reconverted to the optical form. These regenerators work for single wavelengths. The development of EDFA`s has rendered Wavelength Regenerators obsolete. However in EDFA`s, as the number of wavelengths (channels) on a single fiber is increased, the distance between any two channels becomes smaller. This results in some inter-channel crosstalk

If it is desirable to increase the amplifier's bandwidth while eliminating crosstalk then Silica Erbium fiber-based Dual-Band Fiber Amplifiers (DBFA`s) are more suitable. The most important feature of the DBFA`s is their bandwidth (1528nm-1610nm) [23]. They have been shown to generate terabit transmission successfully. The DBFA has two sub-band amplifiers. The first is in the range of the EDFA and the second extends the range and is known as *Extended Band Fiber Amplifier (EBFA)*. EBFA has several advantages compared to the traditional EDFA [23].

1.3.2.2 Wavelength Add Drop Multiplexer

In many WDM networks it is necessary to drop some traffic at intermediate points along the route between the end points. A wavelength add drop multiplexer (WADM) is used for that purpose. A typical WADM is shown in Figure 1.6. A WADM can be realized using a 2x2 switch and a demultiplexer. If the control switch is in the *bar state*, then the signal on the corresponding wavelength passes through the WADM. If the switch is in the *cross state*, then the signal on the corresponding wavelength is dropped locally, and another signal of the same wavelength may be added [11].

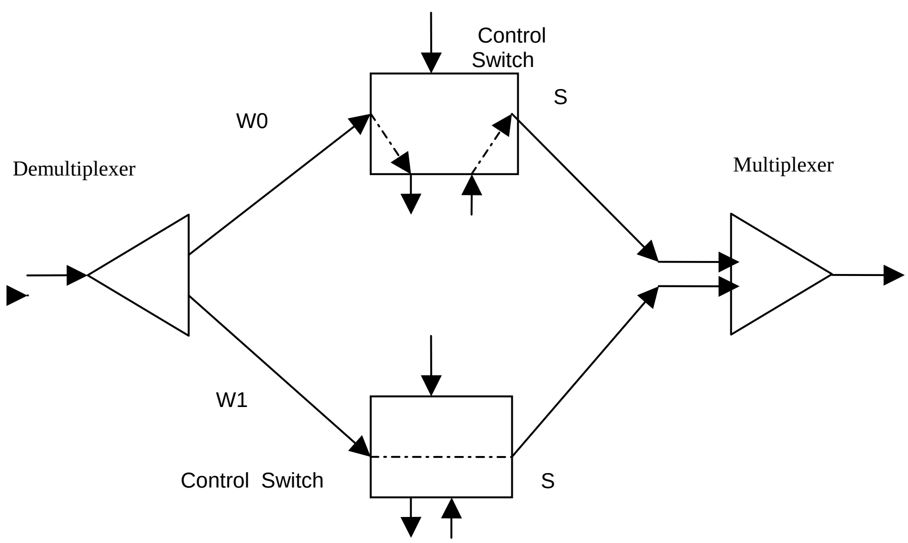


Figure 1.6 Wavelength add/drop multiplexer.

1.3.2.3 Wavelength Crossconnect

Wavelength crossconnect is another essential component of a WDM system. The function of this element is to provide (under network control), the ability to connect or switch any input wavelength channel from an input fiber (or port) to any one of the output fibers (or ports) in the optical domain [24]. A wavelength crossconnect can also drop wavelengths and is often referred to as wavelength selective crossconnect (WXC) or wavelength routing switch. A 2x2 WXC is illustrated in Figure 1.7.

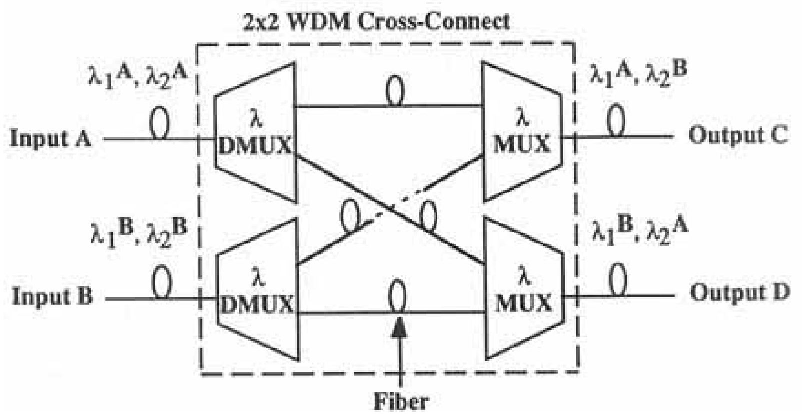


Figure 1.7 WDM cross connect.

1.4 WDM Architectures

WDM architectures can be broadly divided into three categories: Broadcast-and-select networks, wavelength routed networks, and linear light wave networks. These three network architectures are explained in some detail next.

1.4.1 Broadcast-and-Select Networks

These networks have a star coupler connected to all the nodes in the network. This coupler mediates all communications between the nodes. Each node is equipped with a fixed number of tunable transmitters and fixed number of tunable receivers. If any node is transmitting at a particular wavelength, all the destinations tune their receivers to that wavelength in order to receive this message. Different nodes transmit at different wavelengths simultaneously. The star coupler combines all these signals and then broadcasts the combined signal to all the nodes. Multiple communications can take place concurrently by appropriately tuning the positive receivers. These networks are also known as the single hop networks as the message reaches the destination directly without any opto-electronic conversions. The main networking challenge for these networks is the scheduling of these broadcasts. In the absence of efficient coordination, collisions can occur (two or more nodes transmitting at the same wavelength). Several protocols have been proposed for the scheduling of transmitters and receivers [26], [27], [28].

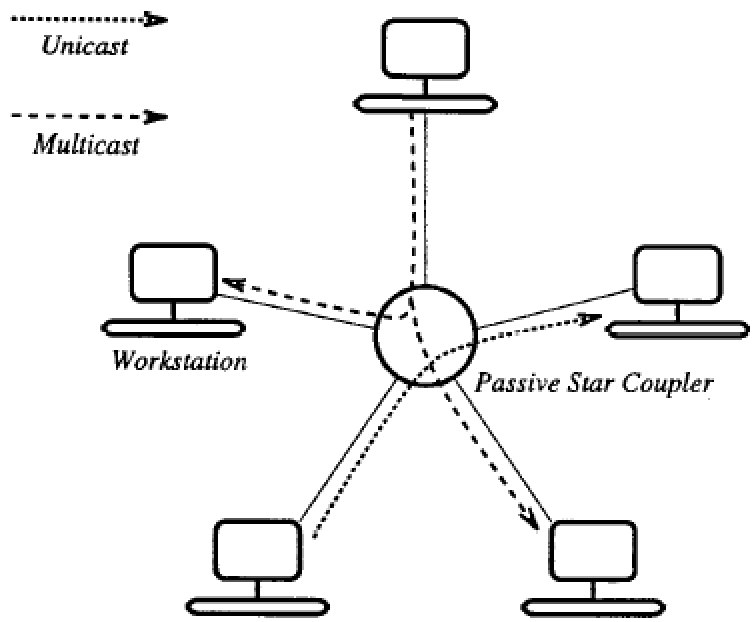


Figure 1.8 Broadcast and select networks.

1.4.2 Wavelength Routed Networks

The basic mechanism of communication in a wavelength routed network is a *lightpath*. A lightpath is an all-optical communication channel between two nodes in the network, and may span more than one fiber link [24]. The intermediate nodes in the path route the lightpath in the optical domain using their switches. This thesis only considers this class of WDM networks known as Wavelength routed networks

A wavelength routed network consists of WXC's interconnected by point-to-point fiber links in an arbitrary topology. Each end-user is connected to an active switch via a fiber link. The combination of an end-user and its corresponding switch is referred to as a network node [11]. Each node consists of transmitters and receivers, both of which may be wavelength tunable. The end-nodes tune their transmitters and receivers to the wavelength used for the lightpath.

In the absence of any wavelength-conversion devices, a lightpath is required to be on the same wavelength channel throughout. This requirement is referred to as the *wavelength-continuity constraint*. This requirement may not be necessary if we have wavelength converters in the network. The basic requirement in a wavelength-routed optical network is that no two lightpaths traversing the same fiber link can use the same wavelength channel. Thus a lightpath uses the same wavelength across all links that it traverses. As an example consider Figure 1.9.

In Figure 1.9 paths P1 and P2 are from node 1 to node 6 and node 3 to node 1 respectively. Since it is a single fiber network, the links (3, 4), (4, 5), (5, 6) are common between these two paths. Thus the same wavelength W1 cannot be used for both the paths. So we have to use two different wavelengths for the paths P1 and P2. Now consider the paths P1 and P3. These two paths do not share any link (they are disjoint). Hence the same wavelength can be used on these two paths. The concept of using the same wavelength on disjoint lightpaths is known at *wavelength reuse*, which is an important feature of WDM networks.

In wavelength routed network, if there are enough wavelengths on all fiber links, then every node pair can be connected and every route may be feasible. This is not realistic because transceivers are relatively expensive and scalability may not be feasible. These constraints place a limit on the number of WDM channels that can be supported on the same fiber. Thus, in a realistic network only a limited number of lightpaths may be successfully routed. Hence the actual challenge in wavelength routed networks is: given a set of lightpaths that need to be established in a network, and given a constraint on the number of wavelengths, a route for the lightpaths should be found and the wavelengths must be successfully assigned to the lightpaths. The routing and assignment should be done in such a way that a maximum number of lightpaths may be realized [11].

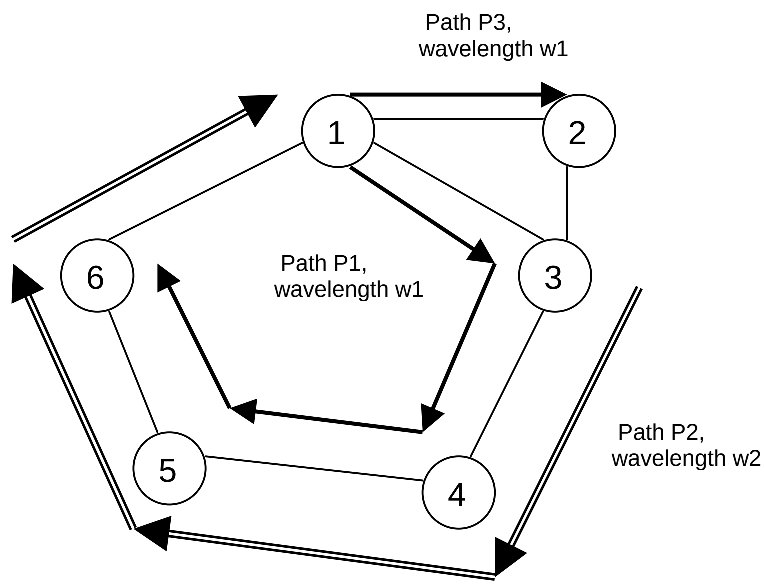


Figure 1.9 Wavelength continuity constraint and wavelength reuse.

This particular problem is referred to as the routing and wavelength assignment (RWA) problem. This problem was addressed in [11]. Routing is usually done by any of the following techniques

- **Fixed routing**: Only one route is used for any given node pair. This path is usually the shortest path
- **Alternate routing**: Two or more routes are provided for each given node pair. These routes are searched one by one in a pre-determined order. (usually increasing order of hop lengths).
  - **Exhaust routing**: All possible routes are searched for each given node pair and the most suitable one is selected. This routing produces the best result but is computationally complex.

The wavelength assignment is also usually done using one of the several schemes. A sample of such schemes is described below

- **Most used**: This method tries to pack the lightpaths so that more wavelength continuous routes are available for the requests that arrive later.
- **Least used**: In this method wavelengths are searched in non decreasing order of their utilization in the network
- **Random:** This method randomly selects a wavelength from amongst the available wavelengths.
- **Round Robin:** This method assigns wavelengths in a round robin fashion from among available wavelengths [11].

All the selected lightpaths may not utilize shortest paths. Some shortest paths may have to be sacrificed in order to maximize the number of allowable lightpaths. Thus, one may allow several alternate routes for lightpaths to be established [11]. The Lightpaths that cannot be setup due to conflicts on routes and\or wavelengths are said to be blocked.

So the corresponding network optimization problem then changes to minimizing the blocking probability.

1.4.2.1 Wavelength Converters

The wavelength continuity constraint causes bandwidth losses. One efficient way to overcome this problem is to use wavelength converters. A wavelength converter is an optical device which is capable of shifting a signal on one wavelength to another wavelength [11]. A wavelength converter of degree D can shift an incoming wavelength to any of the D possible wavelengths at that converter. If the conversion degree equals the number of wavelengths on the fiber, then the converter is said to have full conversion capability. If its degree is less than the number of wavelengths on the fiber, then it is a sparse wavelength converter.

Nodes with wavelength conversion capability are known as wavelength conversion (WC) nodes. Nodes which do not have this capability are known as wavelength interchange (WI) nodes. WC nodes are expensive and hence should be used limitedly. A wavelength convertible network performs better than a WI network as the wavelength converters relax the wavelength continuity constraint thereby giving better bandwidth utilization. Figure 1.10 illustrates this fact.

In the network shown in Figure 1.10, if there are only two wavelengths, then the path P3 between nodes 5 and 2 cannot be established as the wavelengths are locked in the existing paths P1 and P2. However if there is a wavelength converter at node 1 then path P3 can be realized. Wavelength W1 is used for the link (5, 1) and the signal is shifted to wavelength W2 at node 1 thereby using W2 for the link (1, 2). So path P3 uses 2 wavelengths. In the network shown in Figure 1.10, if there are only two wavelengths, then the path P3 between nodes 5 and 2 cannot be established as the wavelengths are locked in the existing paths P1 and P2. However if there is a wavelength converter at node 1 then path P3 can be realized. Wavelength W1 is used for the link (5, 1) and the signal is shifted to wavelength W2 at node 1 thereby using W2 for the link (1, 2). So path P3 uses 2 wavelengths.

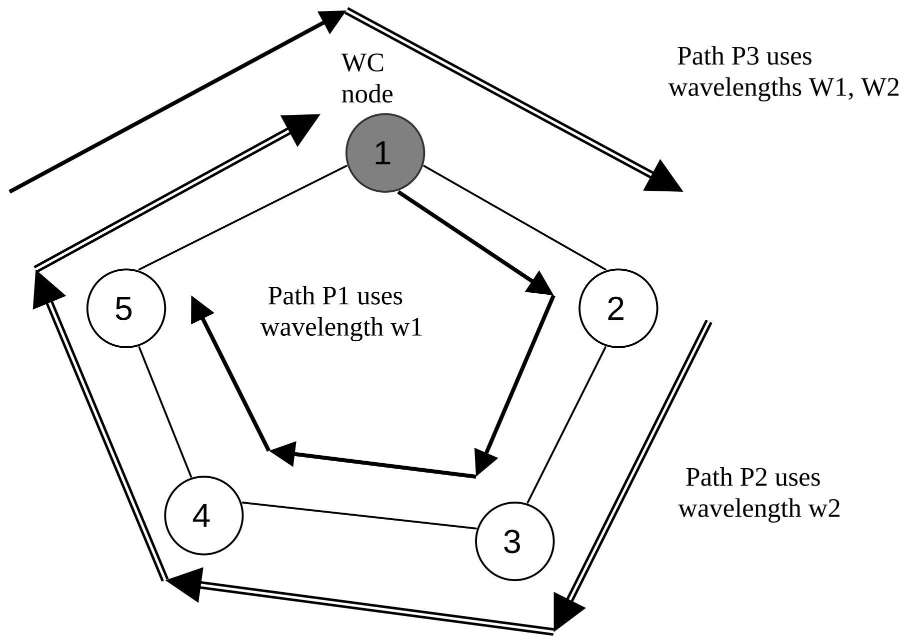


Figure 1.10 Wavelength routing with wavelength converters.

1.4.3 Linear Light Wave Networks

Wavelength routed networks use wavelength (single level) partitioning and several of these wavelengths are multiplexed on a fiber link. Linear light wave networks however use waveband (two level) partitioning and several of these wavebands are multiplexed on a fiber. On each waveband several wavelengths are multiplexed (Figure 1.11). So, unlike a wavelength routed network, linear lightwave network nodes demultiplex, switch, and multiplex wavebands not wavelengths. The hardware requirements are very different from those in wavelength routed networks.

The hardware requirements are actually much simpler than in wavelength routed networks as the number of switches is equal to the number of wavebands and not wavelengths. Since the linear lightwave network doesn’t distinguish between individual wavelengths within a waveband individual wavelengths are separated from each other at the receiving node [11].

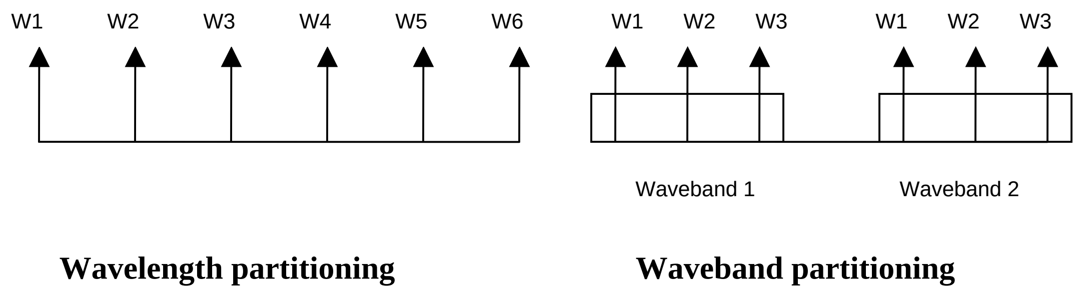


Figure 1.11 Wavelength and waveband partitioning.

Constraints like the wavelength continuity constraint do not apply to these networks. They use distinct constraints like inseparability and distinct source combining [11]. According to the inseparability constraint, channels belonging to the same waveband when combined on the same fiber cannot be separated within the network. Thus they travel together after the point where they were combined. Distinct source combining constraint states that on any fiber only signals from distinct sources are allowed to be combined. These two constraints totally differentiate these networks from wavelength-routed networks.

1.5 Future Optical Networks

WDM networks are perceived as the perfect choice for next generation internet which transports high speed IP traffic. WDM technology is also evolving from the circuit switching technology to burst switching and packet switching technologies. Burst switching is that switching technology where a wavelength channel on a link is reserved only for the duration of the burst [11]. Thus this technology may provide better bandwidth utilization. In packet switching the basic entity is a packet. Thus we need to work on packet synchronization, contention and resolution [11]. These technologies are the subject of much current research and will be for the foreseeable future.

Chapter 2

Multicasting in WDM Networks

As the size of networks increased and the need for continuous media like voice and video grew rapidly, multicasting became an important operation of communication. Multicasting refers to a communication pattern in which information from a single source is sent to multiple destinations. In a point-to point network, a transmission by a node is received only by the node at the other end of the link [1]. However in a single channel (wavelength) multicast network, transmission is received by all the nodes attached to the channel. On any channel of a WDM network at any given point of time, a transmission by a node is transmitted to all the nodes listening to on that channel. Multicasting can be implemented on either single hop networks or multi-hop networks.

Single hop networks known as the passive star couplers consist of nodes interconnected by a passive star coupler. This star coupler mediates between all the nodes. So if any node in the network is about to multicast, the star coupler makes sure that all the destination nodes in the network are tuned to that wavelength so that they can receive the information being sent by the source node. This method of communication renders it difficult to implement single hop networks for larger distances. They are ideal for local area networks [11]. The multi-hop networks are the general networks, which will be discussed in this chapter.

Multicasting was achieved in older networks by duplicating the same message and transmitting it in different destinations. A more efficient way to do multicasting is to transmit a single copy of the same message to all destinations at once. An efficient multicasting algorithm attains a good balance between these two extremes.

Multicasting at the WDM layer has several advantages over multicasting over the IP layer. The former is more efficient and provides consistent support of coding format and bit rate transparency. Moreover, optical switches are inherently capable of light splitting, a mechanism which is more efficient than copying packets. All these factors facilitate multicasting considerably in the optical domain [2], [1]

Multicasting in the optical networks can be achieved in three different ways [27].

1. Making multiple copies of the same message electronically and transmitting a copy to each destination. However this method would need many optical- electronic–optical conversions at every node which makes this method unsuitable
2. Multiple unicasts is another method, which performs multicasting. That would mean separately routing the message from the source to each destination. This method however, is not very appropriate as huge bandwidth is required on the multicast group size (number of destinations in a multicast session).
3. Making multiple copies of the same message optically. This is by far the best method as the bandwidth savings increases in this method. This method is also known as light splitting.

WDM networks do not take into account the first two methods. The third method, *optical splitting* is used widely since it utilizes the inherent quality of optical networks, which doesn't exist in other networks. It removes the complications of electronic- optical-electronic conversions. At the same time it can be used to support new high bandwidth applications such as HDTV program distribution. However due to its nature, the optical splitting approach requires a new set of algorithms and heuristics which are very different from those for regular multicasting algorithms. The differences between multicasting in different domains are diagrammatically shown in Figure 2.1. This figure clearly depicts the benefits of optical splitting over the other traditional methods. In the following sub-sections we introduce some important concepts and review related literature.

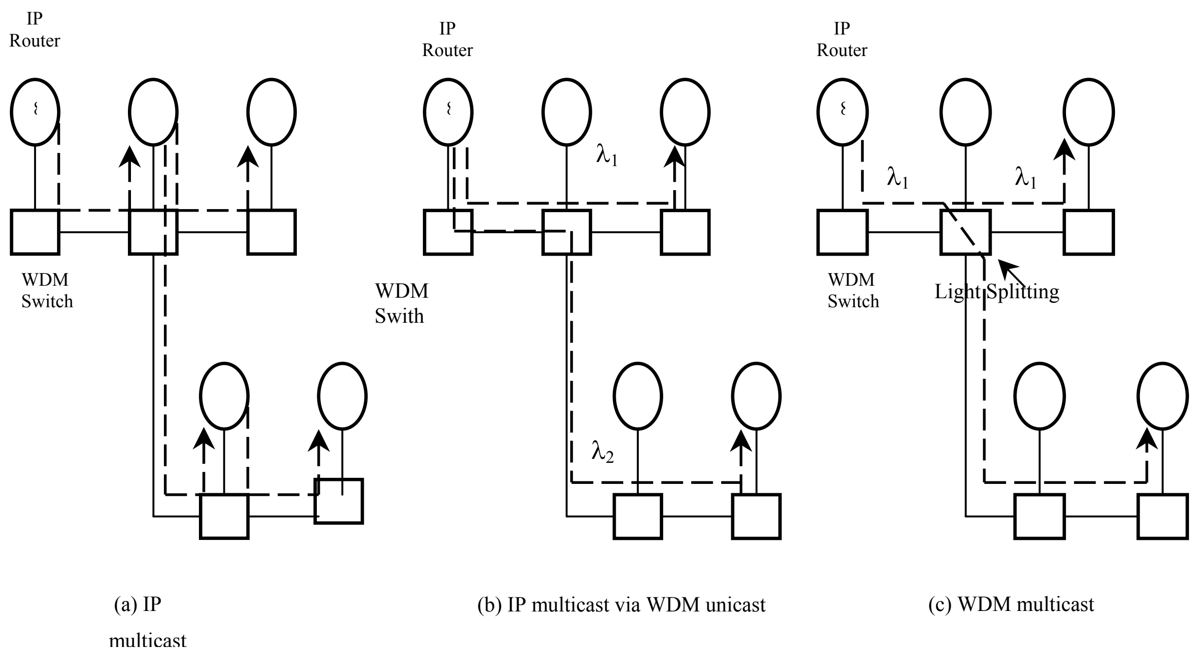


Figure 2.1. The three methods of implementing multicasting in WDM networks.

2.1 Light Trees

Several multicasting algorithms for WDM networks have been proposed in the literature. Most of them use the concept of building light trees. A light tree is a tree rooted at the multicast source and includes all the destinations. The advantage of light trees is that a single wavelength can be used for any link in the tree. So the entire tree uses a single wavelength. This approach helps significantly in facilitating concurrent multicast sessions. Let's assume that each multicast session needs just one light tree (a suitable assumption if the multicast group size is not very large). So if we have 'N' concurrent multicast sessions we would just need 'N' wavelengths to complete these multicast sessions.

All the existing multicasting algorithms in the literature use the concept of Steiner trees. For Steiner trees a cost is associated with each link. The final tree comprising of the source and all the destinations is built with the objective that the cost of the entire tree is minimized. These trees can be built using the shortest path heuristics or spanning trees or some Meta-heuristics [31].

2.2 Shortest Path Heuristics

The algorithm in [9], [29] initializes the Steiner tree with the shortest path from the source to an arbitrary multicast member. It then repeatedly includes a new member by adding the shortest path between this new destination node to the current partial tree. This entire procedure is repeated until all members of the multicast group have joined the tree. Many variants of this algorithm have been developed to improve the quality of the final tree. One such variant includes the members in the order determined by their distance to the multicast tree. Variants choose the new destination from the destination set at random [30]

2.3 Spanning Trees

The algorithm in [30] first constructs a closure graph of the multicast nodes from the original graph using the cost of the shortest path between each pair of members. A minimum spanning tree of the closure graph is obtained, and then the shortest paths in the original graph are used to replace the edges of this minimum spanning tree. Finally, the multicast tree is obtained by removing any cycles.

2.4 Light Splitting

Light splitting or power splitting is a concept, which is unique to the optical medium. A light splitter is a passive device used to distribute input signal to all the out going edges of the input node. Figure 2.2 is given here as an example, where the node *A* is required to transmit the message to three of its direct neighbors (nodes directly connected to it in the light tree) *B*, *C*, *D*.

Instead of the regular method of converting the optical signal into an electrical one and then making three copies of the same message and then converting all the copies back to the optical form, we can use the concept of optical splitting which is similar to the concept of current division. If the signal strength at node *A* is *P*, then the signal undergoes a three way split with each branch receiving a signal of strength *P/3*. After splitting each one of the three nodes *B*, *C, D* receives a signal of strength *P/3*.

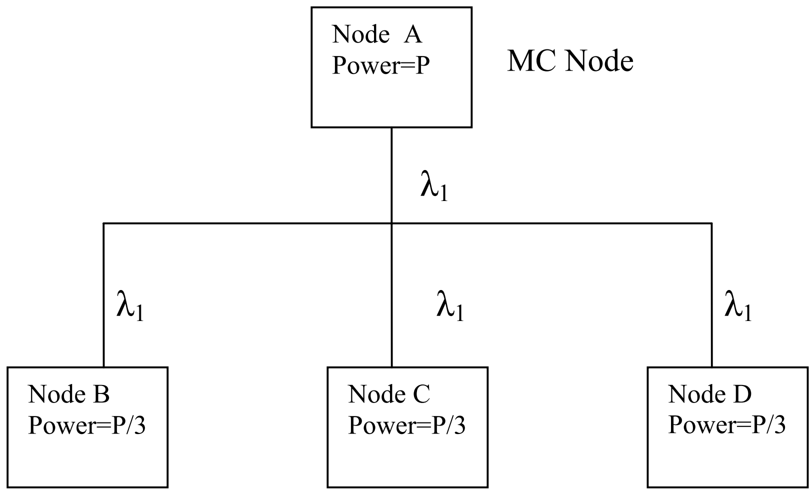


Figure 2.2 A typical multicast capable node.

A node that is capable of splitting the signal is called a multicast capable (MC) node. Not all nodes in the network will have this capability. Some nodes cannot split light. Those are called the multicast incapable (MI) nodes. Building a MC node may be slightly more expensive compared to a building a MI node. So all nodes in the network need not be MC. Such networks are known as the sparse light split networks i.e. networks in which only a subset of the nodes are multicast capable. Previous research on the architecture of the multicast networks has shown that it is not necessary that all the nodes in the network be multicast capable. To achieve good performance it was also shown through simulations that about 50% of the nodes should be multicast capable [ ]. For the same reason in this thesis we will assume that about 50% of the nodes in the network have splitting capability.

Since not all the nodes in the network have splitting capability (nodes having the branching ability and the ability to have as many children as needed), it may not be possible to reach all destinations through a single light tree. It may be required to have more than one light tree. A collection of such trees is known as a light forest [b]. If shortest paths were not required while building the tree, then it may be possible to include all destinations within the same tree. But if each destination is (required to be) reached through the shortest possible path then the chances of ending up with a light forest become higher. MC and MI nodes usually employ a Splitter-and-Delivery Cross connect or a Tap-and continue cross connect respectively for their purposes. These are described in the next section.

2.4.1 Splitter-and-Delivery Cross-Connect

The SAD is a MC cross-connect that was proposed in [7]. An NxN cross-connect consists of a set of SAD switches for each wavelength. An NxN SAD switch consists of an interconnection of N power splitters, N^2 optical gates (to reduce the excessive crosstalk), and N^2 X 1 photonic switches [7]. Figure 2.3 shows the organization of a cross-connect based on the SAD switch. In addition to the W SAD switches, N demultiplexers (multiplexers) are used to extract (combine) individual wavelengths. A SAD switch with a splitting capacity of S need not always split S times. Depending on the tree it can split M=1 or 2 or S times. A large number of splitters in the network have the negative effect of the need for optical amplifiers and high fabrication cost.

2.4.2 Tap-and-Continue Cross-Connect

Figure 2.4 shows an implementation of a TAC cross-connect. A TAC is a typical node which cannot split. It can just send a small part of the signal power to the local node and forward the remaining signal to the successor(s) of the local node. A NxN TAC cross-connect supporting W wavelengths uses a set of W 1xN Tap-and-Continue Modules (TCMs). In TCM`s, an extremely small fraction of the input signal is tapped and forwarded to the local station. The remaining power which is more than 90% is switched to any one of the other outputs. The tapping device used is fully programmable so that the tapped signal power is determined by considering the signal to noise ratio (SNR). To switch the signal to any of the outputs, switching elements (SEs) are used. An input signal is tapped in by the local station using a tapping device. A small fraction of the signal power is directed toward the local station, while the rest continues to a multistage network of SE`s.

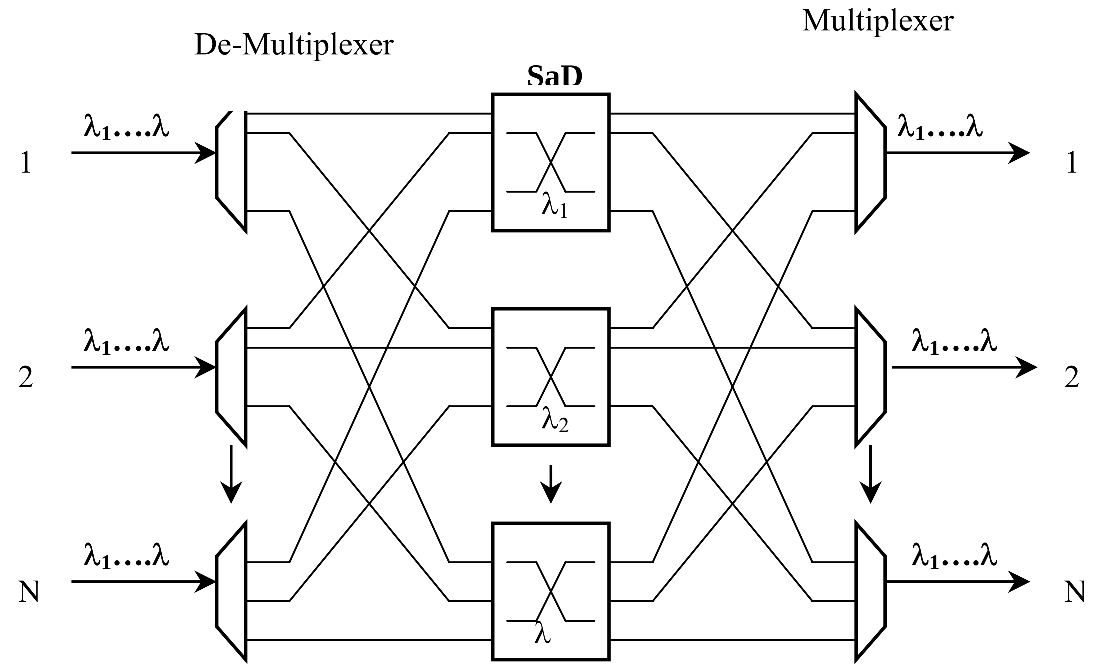


Figure 2.3. An NxN SAD cross connect.

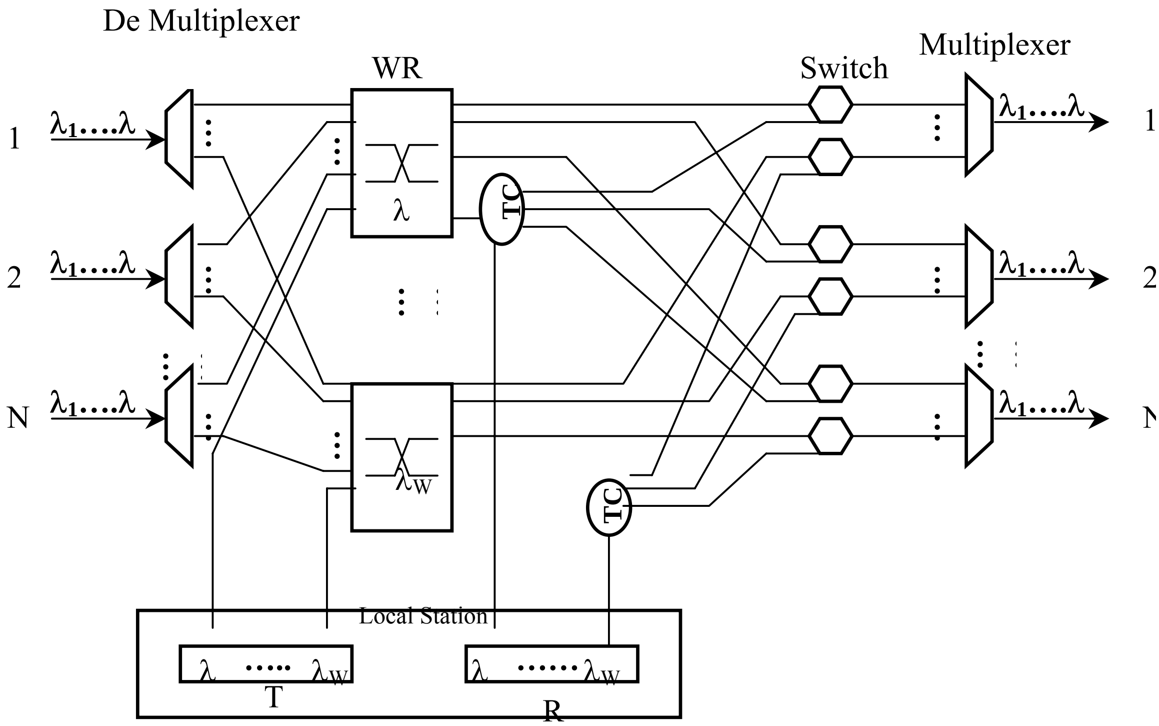


Figure 2.4 An NxN TAD cross connect.

2.5 Power Losses

The problem of power losses was not encountered in electronic packet or circuit switched networks and thus was not addressed in the existing multicast routing algorithms. In order to maintain reliability in optical networks the power or signal strength at any node in the tree must be above a fixed level [2]. This level is mandated by the ability of the light detectors at the destination to recover the information reliably. In addition a very weak signal could give rise to unacceptable signal to noise ratio. It has been suggested that for a signal level higher that 34db may be necessary for accurate recovery of the signal (assuming that the signal strength is 50mW at the source). This reflects the sensitivity of the avalanche photodiode receivers [2]. So if the signal level drops below this level at any node, then that node will not be capable of reliably recovering the received information.

As explained earlier the nodes that are employed in multicasting are the Drop And Continue (DAC) type. These nodes will typically tap about 0.1-2% of the incoming signal and will send the remaining power to the output ports of the switch at that node. This process will continue till the last destination is reached in that particular branch of the tree. The signal is no longer forwarded after reaching the last destination of that branch. However in transmitting the message (signal) from the source to the destination the signal incurs several losses and gets deteriorated. This signal deterioration should be kept under check so that the signal can still be detected reliably by the photo diode receivers at each destination node.

A simple approach that may be suggested to reduce these losses is to amplify the signal at regular intervals. But there is a limitation on the number of times a signal may be amplified [17]. Amplifiers are not devices, which function perfectly. Besides, optical amplifiers are very expensive devices. So they cannot be used at such regular intervals. The optimal placement of optical amplifier's is another problem by itself [17]. All these problems suggest that some other approach other than optical amplification must be used to minimize power losses. A better solution would be to build the multicast tree within the limitations of amplifier positioning.

Consider Figure 2.5 as an example. Assume that all the links in the figure are of equal length. Here, the dark nodes denote nodes that have amplifiers, the gray nodes denote the destination nodes and ***S*** is the source. Instead of routing (or building the tree) as in Figure 2.5 (a) we can route it as in Figure 2.5 (b). In Figure 2.5(b) even though we are using a longer route, this route offers better utilization of the amplifier placement. Instead of using the entire path between two consecutive amplifiers as in Figure 2.5 (a), we can choose a longer route, which doesn't let the signal deteriorate significantly. In Figure 2.5 (a) the signal is first amplified and then sent through a number of nodes without amplification capability before reaching the destination. In Figure 2.5 (b) the signal is allowed to attenuate somewhat before the signal requires amplification

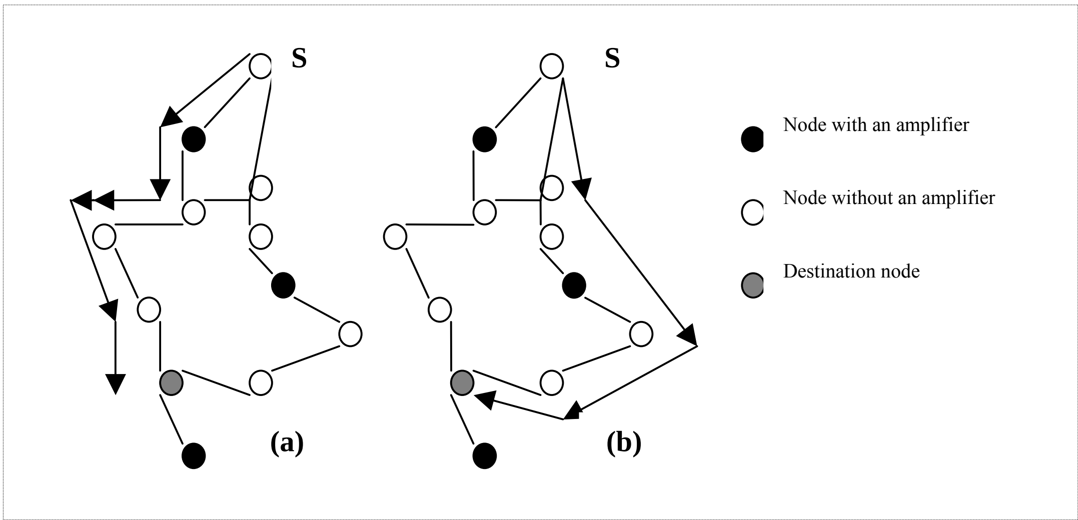
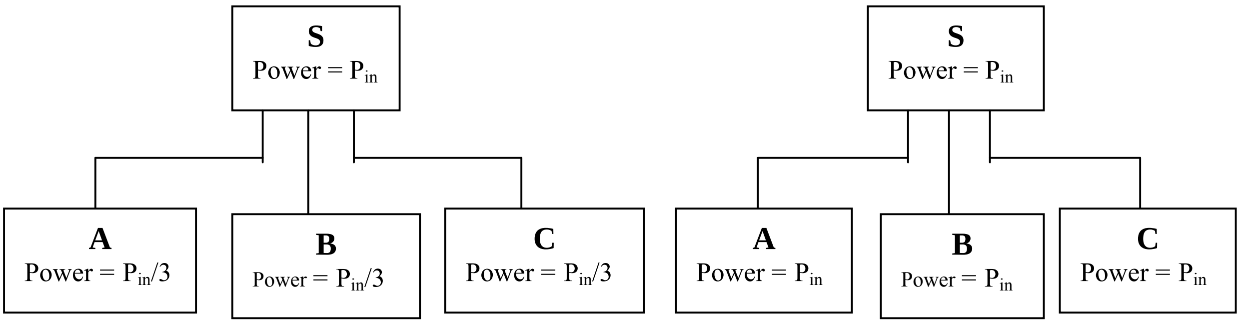


Figure 2.5 Example illustrating light tree routing considering the amplifier positioning.

The above case was given just as an example and doesn't solve the problem of reducing power loss completely. In order to come up with heuristics which reduce power loss it is very important to clearly understand these losses and the causes for the losses. The losses that an optical signal suffers as it travels from source to destination are basically of two types- losses due to optical splitting and losses due to signal attenuation.

2.5.1 Splitting Losses

An M-way splitter splits a signal of power P into M signals of power P/M. Ideally all these M signals should be of the power P if the division was done by some other mechanism other than splitting (Figure 2.6 (b)). The case of all three nodes receiving a power of P is not possible at all if splitting was used unless built-in amplifiers are used in the splitter itself. In reality the power of each successor of a splitting node will be lower than P/M due to some multiplexing losses.



(a) **(b)**

Figure 2.6 Power division (or loss) in a splitter and an ideal case of power distribution.

In order to derive the mathematical expression for the splitting loss let us consider a node S having M outgoing edges in the tree. Assume that the splitter is configurable such that a multicast optical signal does not always need to be split K times, where K is the number of input/output ports in the switch. Instead the multicast signal can split into M=1 or 2 or …...up to K times. Given these assumptions, the power loss (in dB) at an MC optical cross connect for an input signal that is split into M output signals is given by [2], [4]:

$$P_{split\ \ loss} = 10\log(M) + \beta(K) \quad \textbf{(1)}$$

In expression (1) the part *10log₁₀ (M)* captures the actual splitting losses [2], i.e. the power loss due to an ideal *M*-way splitter. The function β is used to compensate for the additional losses like the losses due to multiplexing, demultiplexing, insertion and coupling at the switching elements in the splitter node. Since the number of switching elements is proportional to the number of input/output ports *K* at the node, β is a function of *K*. These additional losses take place at every node in the tree.

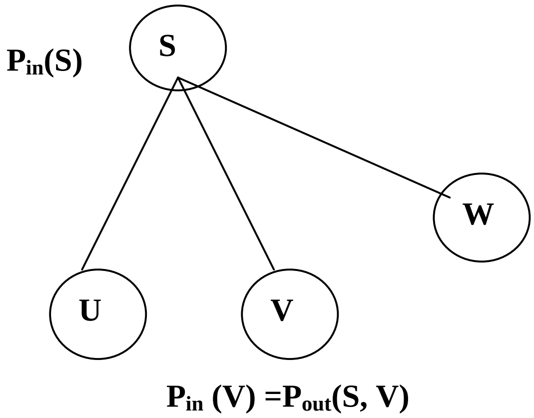


Figure 2.7 The notation used in the power expressions.

Figure 2.7 gives the notation used in the power expressions. The term $P_{out}(S, V)$ denotes the power received at node *V*, if the signal is assumed to originate at node ***S***. The signal power at any node is denoted by P_{in} *(node number)*. Thus in Figure 2.7, a signal of strength *Pin (S)* originates at node *S*, undergoes an *M*-way (*M*=3) split (thereby losing a power of $P_{split\ loss}$**)**, and reaches the nodes *U, V, W*. All these three nodes will receive a signal of the same strength (equal to *Pout(S, V)*, assuming that they are equidistant from node *S*). So *Pin (V) = Pout(S, V)*. From (1) using the power notation as in Figure 2.7, the expression for P_{out} at node *V* if the signal originates at node *S* is as follows:

$$P_{out}(S,V) = \frac{P_{in}(S)10^{-\beta(K)/10}}{M} \qquad \textbf{(2)}$$

This expression assumes that there is no amplification at the splitter. If there is an amplifier of gain *G* in the splitter (to compensate for the splitting losses) then (2) becomes:

$$P_{out}(S,V) = \frac{P_{in}(S)G10^{-\beta(K)/10}}{M} \quad \textbf{(3)}$$

If $G10^{-\beta(K)/10} = R$ in Equation 3, then we have:

$$P_{out}(S,V) = \frac{R * P_{in}(S)}{M} \quad \textbf{(4)}$$

R from (4) is a constant for a given SAD switch, and is determined by the number of ports *K* of the switch, the losses incurred at the various elements of the switch, and the amplifier gain. Thus, the output power of a signal that has undergone *M*-way splitting is given by:

$$P_{out}(S,V) = \frac{R * P_{in}(S)}{M} \leq P_{in}(S,V)$$

2.5.2 Signal Attenuation Losses

This loss is due to the propagation of light from source to destination. It has been shown previously that the maximum transmission distance in a fiber is 255km for a signal whose strength is 1550nm at the source [2]. That means that with current fiber technologies, for every 255km of signal propagation an optical amplifier is a must. A number of EDFA (Erbium Doped Fiber Amplifiers) must be added to compensate for the power attenuation so that the power at the receiving end is no less than –34db. The 255km is an idealistic figure, which doesn't take into account fiber non-linearity and receiver sensitivities.

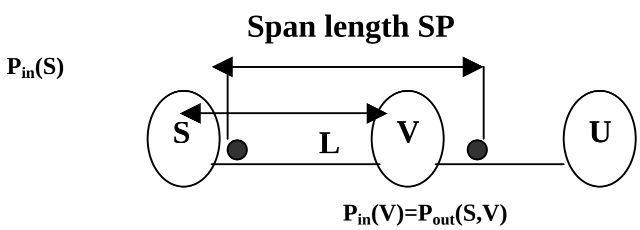


Figure 2.8 Notation used in the power loss expression for attenuation loss.

So every time the signal travels some distance, be it through splitters or without splitters it always undergoes attenuation losses, which depend largely on the distance traveled by the signal. The expression for the power losses due to signal attenuation is given by expression (5) [2],[4].

$$P_{out}(S,V) = P_{in}(S) * G * e^{-\alpha * SP(l/s)} \quad \textbf{(5)}$$

Figure 2.8 illustrates all the terms used in expression (5). *SP* is the span length between two successive amplifiers. An amplifier is required for a distance of *SP*. In other words the signal deteriorates completely after traveling a distance of *SP*. *G* is the gain of the amplifier being used for this span length. *L* is the length of the link from the node *S* to node *V*. 'α' is the fiber attenuation ratio. At 1550nm we have 4.34α=0.2 [2]

On simplifying expression (5):

$$P_{out}(S,V) = P_{in}(S) * (G^{1/SP} e^{-\alpha l}) = P_{in}(S) * Q^{L} \quad \textbf{(6)}$$

Thus the signal attenuation loss is exponentially dependent on distance.

2.6 General Expression for Power Loss

The power at any node *V* rooted at node *S* in the multi-cast tree is given by (7)

$$P_{out}(S,V) = P_{in}(S) * Loss_{atten}(S,V) * Loss_{splt}(S,V) \quad \textbf{(7)}$$

In expression (7), The loss due to attenuation is given by (8)

$$Loss_{atten}(S,V) = \Pi(Q^{D(l)}) \quad \textbf{(8)}$$

The product is taken over the entire path P (*S, V*), from the source *S* to the destination node *V* in the multicast tree. *l* is the length of each link in that path.

Similarly the loss due to splitting is is given by (9)

$$Loss_{split}(S,V) = \Pi(\frac{R}{F_T(u)}) < 1 \quad \textbf{(9)}$$

The product is taken over the entire path P (*S, V*), from the source *S* to the destination node *V* in the multicast tree. F_T is the fan-out of each node *u* (or the number of outgoing edges at each node u) on the path P(S, V) of the multi-cast tree. Now combining both losses, the final expression for P_{out} is given by (10)

$$P_{out}(S,V) = P_{in}(S) * \Pi(Q^{D(l)}) * \Pi(\frac{R}{F_T(u)}) \quad \textbf{(10)}$$

Where the first product is taken over the entire path, from S to V with V being the last destination in the path, and l being the length of each link in the path. The second product is taken over the entire path from S to V, with u being node number of each node on the path.

Consider the example in Figure 2.9. The fan-out at nodes 2, 3 and 5 are 1, 2 and 1 respectively. P_{out} at node 6, if node 1 is the source is calculated using the above expressions. It is:

$$P_{out}(S,6) = P_{in}(S) * (Q^{L1} * Q^{L2} * Q^{L4} * Q^{L5}) * (\frac{R}{1} * \frac{R}{2} * \frac{R}{1})$$

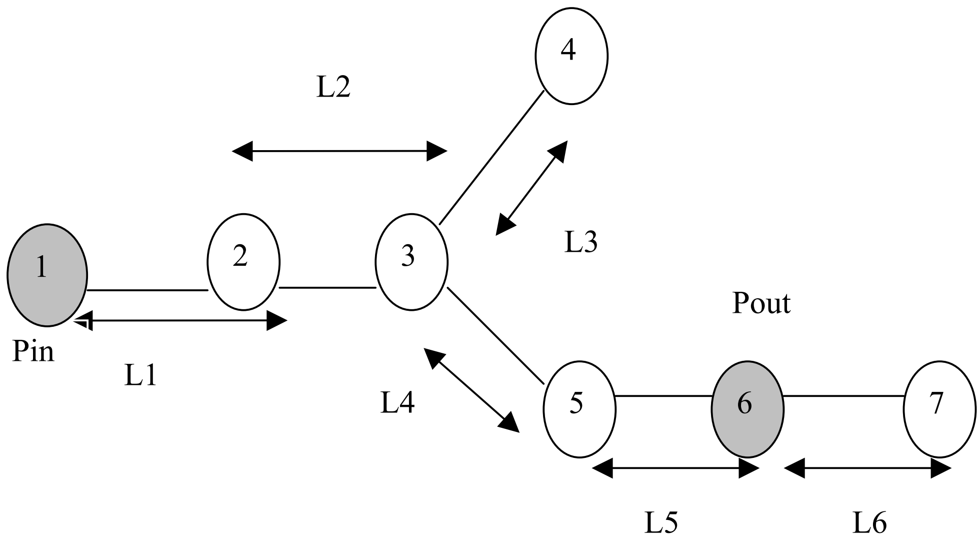


Figure 2.9. An example to illustrate the power distributions.

2.7 Thesis Problem Definition

From the above discussion it clear that power losses are unavoidable in a multicast tree. The splitting loss is usually acknowledged [3]. Attenuation loss however is usually not considered but can prove to be dangerous over long distances. It can also be deduced from the power loss expressions that we need to design algorithms that bring a balance between both types of losses. If a longer route is chosen then the attenuation losses can become larger and the power received by the last few nodes can be very small. This power will reduce even further if there are many splitters in the longer route. So one might want to have splits in the shorter branches of the tree instead of the longer branches. There are several other observations, which can be made from the power expression. These will be discussed in the next chapter.

Not much research was done in the area of reducing power losses, as none of the existing optical multicasting algorithms deal with this issue [1]. Previous work dealt mostly with building the tree, based on shortest paths. Some authors just acknowledge power losses but very few consider them when building the multicast tree.

The primary objective of this thesis is to reduce the power losses in the optical multicast trees. There was not much groundwork done in this area to compare this thesis with. One of the earliest works in this area is by Ali et.al. [7] who introduced an algorithm that attempts to find a Hamiltonian path including the source and the destinations. There are several shortcomings for this approach since it could result in very lengthy routes. The work by Rouskus et.al [2] focuses on building power balanced trees by pruning and adding new branches to an already existing multicast tree. The aim of that paper was to reduce the difference between the maximum powered node and the minimum powered node in the tree. The minimum powered node is discovered and the entire branch that contains that node is removed. All the removed nodes are then routed to the nearest MC node. They use various variants of the former algorithms like re-routing the removed nodes to the source or to another node, which has better power. The work in [10] uses a totally different approach that minimizes the average power loss rather than the individual power losses. It uses a similar approach to the one in [1]. However while re-routing the pruned branches, instead of routing them to an MC node, which is less loaded, they route them to the most highly loaded MC node. This increases the average power but at the same time, decreases the power at the individual destination nodes. Average power may not be a credible measure of power improvement as the final aim is always to ensure that all destination nodes in the multicast group receive signal power above the minimum required. The improvement in average power would be totally futile if the individual powers do not exceed the minimum required power.

The problem addressed in the thesis is different from the ones addressed in previous papers. We do not try to power balance the trees. Instead we attempt to build trees such that all destinations in the tree receive power greater than the minimum power required for the signal to be recovered reliably by the receiver at the destinations. Instead of just pruning the shortest path trees, the algorithms discussed in the next chapter take some measures while building the tree itself. The tree is pruned if it is still not satisfactory. That provides a twofold improvement. Three algorithms will be introduced and discussed. The algorithms will be compared with a known WDM multicasting algorithm. The three algorithms are different in nature but attempt to attain a common goal: that of ensuring that all the destinations receive a signal of acceptable strength.

Chapter 3

Power Efficient Multicast Algorithms

In designing power efficient algorithms our objective is that all destinations in the multicast group receive a reliably detectable signal above a fixed threshold power. Stated in other words, we are attempting to reduce the degradation of the quality of the signal as it travels through the optical network. As described in Chapter 2, there are two dominant sources for power losses (attenuation losses and splitting losses), which degrade the quality of the signal. There are essentially two choices, either to reduce splitting losses or to reduce attenuation losses. However the nature of the problem at hand is such that an attempt to reduce one source of losses may result in an increase in the other. In order to reduce the splitting losses, longer routes might have to be chosen, which leads to higher attenuation losses. In order to reduce the attenuation losses, routes through closer MC nodes have to be chosen which might increase the losses due to splitting. Therefore optimizing with respect to both types of losses is essential. While building the tree, none of the losses should be allowed to attain significant levels on any branch of the tree. If that happens then either one of the losses should be minimized. For example if the route to any destination is very long, then there should be fewer splits in the route and vice-versa. However a complete balance may never be achieved since the problem is an NP Complete problem [2].

The heuristics, which aim to improve the signal strength at the destinations, will employ certain techniques. Section 3.1 next discusses such techniques and some factors, which may affect signal strength. Section 3.2 will introduce and explain the three algorithms we developed for power efficient multicast in WDM networks.

3.1 Preliminaries

The following two subsections discuss some methods which may reduce power losses in optical multicast trees.

3.1.1 Non-Shortest Paths

An important feature of power efficient algorithms is that the chosen routes need not always be shortest paths. Algorithms based on shortest path heuristics add one path (from any node in the tree to any destination in the multicast group) at a time to the tree depending on which destination is closest to any member of the light tree. However sometimes it may be beneficial to select a slightly longer route in order to reduce the power losses due to splitting.

The problem addressed here is not just to find a multicast tree for a given multicast group, but to find a multicast tree within the boundaries of certain power constraints. Therefore it may be necessary to adopt non shortest paths for some destinations. However if the non-shortest paths are adopted for all destinations, the propagation delay may become significant. So non shortest paths should be used only when there is a significant power increase or when the delay does not exceed certain given limits. Unfortunately there exists no fixed rule in this case. One may choose either a longer route or a shorter route depending on the power received by the destination of that route in both cases.

3.1.2 Backtracking Technique

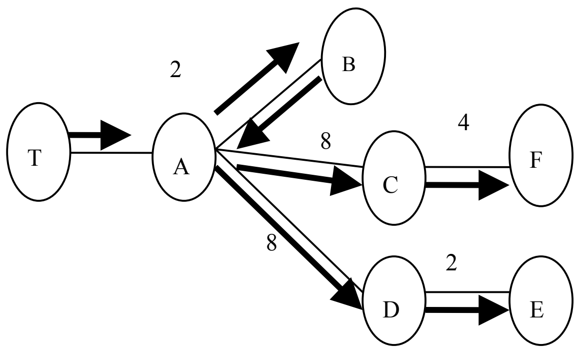
Backtracking is another technique that can be used to reduce signal degradation. It is a technique, which was not used previously to improve the signal strength at destination nodes. Backtracking was used to find a single trail for a multicast session. The trail originated at the source and included all the destinations [7]. In this work we utilize the backtracking technique to reduce power losses due to splitting by replacing splitting losses into lesser attenuation losses. To our knowledge this is the first time backtracking is used for that purpose.

Figure 3.1 depicts part of a tree. For normal routing the signal has to split at node *A* in order to reach destinations *B*, *E*, and *F*. Instead of splitting the signal 3-way, the signal may be routed as shown in the figure. The signal can be routed from *A* to *B* and then back from *B* to *A* and then it can be split 2 ways along *A* → C → F and A → D → E. So backtracking allows nodes and edges to be visited more than once in a path [the undirected edge (A, B) is visited more than once]. However the same wavelength can be used for the link from A to B and the link from B to A, since they are treated as two different channels. Backtracking is made feasible by the fact that a Tap and Continue node or a Splitter node can be visited more than once in a trail [7].

Figure 3.1(b) clearly depicts the physical implementation of backtracking. In this example it is assumed that the switches at the nodes have 3 input ports and 3 output ports. So the signal travels through two different input ports (in the same lightpath) at node A as it travels from T to A and from B to A. Thus node A is visited twice in the same lightpath.

There is a disadvantage associated with backtracking. Every time one back tracks the path length becomes longer by twice the length of the link(s) on which backtracking is implemented. This increases the delay and attenuation losses of that particular route. As the number of links (over which back-tracking is implemented) increases, so does the length of the path and the attenuation losses. Eventually attenuation losses may exceed the splitting losses we are trying to avoid, thereby offsetting the usefulness of the technique.

(a) A network representing back-tracking



(b) Physical representation of back-tracking at the switches at node A

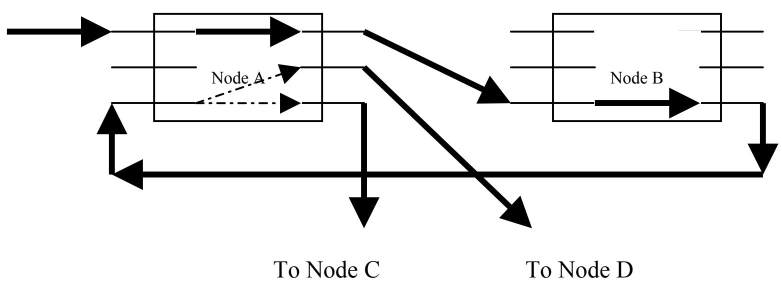


Figure 3.1 An example illustrating backtracking.

As an example consider Figure 3.1 which shows a light tree. Node F is the least powered node in that light tree. Let P be the power at node A. Using the expressions from Chapter 2 and the link lengths shown in Figure 3.1 (a), the power at node F is 0.198*P. Instead, if backtracking was implemented on all the links from A to B and the links from A to E, then the route from node A to node F will comprise of the links (A, B), (B, A), (A, D), (D , E), (E, D), (D, A), (A, C), and (C, F). In this case the power at node F would be 0.134*P (i.e.< 0.198*P). So, in this example complete back-tracking was not beneficial. This proves that backtracking may not be advisable for longer links or paths. So it may not be advisable to backtrack repeatedly every time there is a split. Instead it is desirable to use backtracking judiciously, only when it improves signal strength at the destinations.

Now that we have discussed a couple of techniques to reduce power losses, let us discuss some factors that influence signal strength in optical networks.

3.1.3 Number of MC nodes in the network

An important factor, which affects power at the destinations, is the number and location of nodes in the network that have splitting capability (MC). In many cases it may be required to prune and extensively re-route the pruned branches of the multicast tree to improve signal strength at the destinations. For such trees it is easier and more flexible to re-route the pruned branches if there are many multicast capable nodes in the network.

While building the light tree there will generally be more paths to choose from for every destination node when the number of multicast capable nodes in the network is larger. Thus an increase in the number of multicast capable nodes could result in more power efficient trees. It has been proved before in literature that including more than 50% MC nodes in the network may not result in any significant benefit. Thus it is adequate and desirable not to include more than 50% nodes. In this thesis it is assumed that around 50% of the nodes in the network have splitting capability.

3.1.4 Multiplexing & Demultiplexing Losses

A power loss due to the multiplexing and demultiplexing of switches at each node is usually not considered when building the multicast light trees. These losses are included in the splitting loss at each node as given by the expression (1) (see Chapter2).

$$P_{out} = \frac{R * P}{M} \tag{1}$$

Here the factor R takes care of the multiplexing and demultiplexing losses. A typical value of R is approximately 0.98 [2], [4]. These multiplexing and demultiplexing losses occur at every node in the light tree. The net splitting loss from the source to any destination will be the sum of the individual splitting losses at each node taken over the entire path from the source to that destination. The net power loss increases by around 2% (if R =0.98) at every node. Thus if there is a choice between two routes where route one is having a length L and P hops and route two is having a length L and Q hops (where $Q<P$), then its advisable to take the latter route as it has fewer visits to intermediate nodes.

So fewer hops would result in better power at the destination nodes insofar as multiplexing and demultiplexing losses are concerned. These losses in some cases may not be significant when compared to the two dominant losses (attenuation and splitting losses). However at the global level, these losses may not be significantly small to be neglected in all situations.

3.1.5 Balanced Light Trees

Once the light trees are formed for a particular multicast group, it is beneficial to make sure that these trees are balanced [2]. The difference between the powers at any two destinations must be minimized. It is not desirable to have a destination receive a weak signal while other destinations are receiving strong signals. This can be used as a deciding factor on the amount of pruning that a particular light tree requires. The pruning may be continued until the power difference (between the maximum powered node and the minimum powered node in the light tree) falls below a fixed level.

A threshold can also be set on signal strength at the destination nodes. The light tree can be pruned till signal strengths at all the destinations of the multicast group exceed this threshold. This can be a good criterion to decide on the amount of pruning that is required. The threshold can be chosen to reflect the sensitivity of photodetectors at the destinations. Care must be taken to make it always possible to construct such trees. Else some nodes will not receive reliable signals. In this thesis we assume that we can always construct such trees. In this thesis we consider situations where destinations can always be made to receive power greater than the threshold.

3.2 Power Efficient Multicast Algorithms

In this section we introduce three new algorithms for the construction of power efficient multicast trees. The problem of designing power efficient multicast algorithms is dealt with differently in this thesis. Current approaches use shortest path tree heuristics to build the tree and then prune it to reduce power losses [2], [10]. Our approach however reduces power losses while building the tree itself. If the power improvement is not within the acceptable limits, then additional pruning is utilized. Thus the set of algorithms described in this thesis use a two-tier power reduction technique.

Because we utilize power improving techniques while building the tree, our basic multicast tree will be different from the ones that would be obtained by the previous algorithms [1], [2]. Furthermore, the steps we take in pruning the trees are different from the ones used in earlier works [2], [10]. As we shall show in Chapter 4, our techniques give better results than those of the existing algorithms (Chapter 4).

The basic aim of these algorithms is to reduce the power losses in multicast optical networks. We begin by initializing the tree to contain only the source. The data structure TREE is initialized to the source. We iteratively add nodes to TREE as follows: whichever destination is "closest" to any member of the currently constructed tree (Algorithms 1 & Algorithm 2) is chosen and the path from that node of the tree to that destination is added to the tree. A priority scheme is used if any two destinations waiting to be included in the tree are equidistant from a particular selected member of the tree. The node that has the highest priority is incorporated into the tree. However even before that node is included, backtracking (Section 3.1.2) is used to check if any other route from a member of the tree to the selected destination can be chosen so that the power at the minimum powered destination can be improved. The node with the minimum power in the entire multicast light tree will be called the *minimum powered node*.

Once the tree is built, backtracking is used again to prune the tree. This is the basic procedure adopted for the first two algorithms. However these two algorithms are very different from each other in terms of the priorities and the method used for pruning the light trees. Algorithm 1 uses backtracking extensively and uses slightly longer routes while Algorithm 2 uses shortest paths and a milder version of the backtracking technique. The third algorithm (Algorithm 3) is very different from the first two algorithms. It uses the power of every destination node to decide on the node that is to be included in the tree. It should be noted here that Algorithm 2 has the smallest time complexity among our algorithms where as Algorithm 3 has the highest time complexity. This time complexity may be important in real time applications. The tree algorithms are discussed in detail in Section 3.2.2. Before that, we define the network and some of the sets that will be used in the algorithms.

3.2.1 Definitions

To facilitate the presentation, we represent a network of MC-OXCs by a simple graph G= (V, E). V denotes the set of nodes, and E denotes the set of edges of fiber links connecting the nodes. Let N= $|V|$ be the number of nodes in the network. D (*u, v*) is the distance of the path from node *u* to node *v* where *u* Є V and v Є V. In the lighttree that we are building the source of the multicast session is represented by *s* where *s* Є V. DEST_SET (which is a subset of V- {*s*}) is the set of destination nodes in the multicast group.

Let TREE be a data structure representing the multicast tree being constructed at any given point of time. F(s, DEST_SET) is a data structure representing a forest of trees containing the trees constructed thus far and rooted at node s, the source of the multicast session. CANDIDATES_SET is the set of nodes in the TREE to which the new destinations can be routed. This set consists of all the unexhausted multicast capable nodes and the leaf MI nodes in the TREE. An MC node is said to have been exhausted if its split capability has been fully used already. Else the MC node is said to be unexhausted. The nodes in this set are a subset of the nodes in TREE. The set USED_SET consists of the MC nodes which have been exhaustively used or the MI nodes which have already been used as non leaf nodes. The elements of this set cannot have any further branches emanating from them. A path from *u* to *v* will be denoted by P (*u, v*) where *u* Є V is the source of the path and *v* Є V is the end point of that path.

3.2.1.1 Algorithm 1

In the following subsection we introduce Algorithm1 which may be stated as follows

Begin

{

(1) F(s, DEST_SET) = ø; DEST_SET = All destinations in the multicast group.

(2) TREE = ø; USED_MI_SET = ø; CANDIDATE_SET= {s}.

(3) For each node pair (u`, $v$`) such that u` Є CANDIDATE_SET and $v$` Є DEST_SET, try to find the shortest physical length path P`($u$`, v`) such that no node on this path is in USED_SET. If no such path exists go to Step 8

Else

{

From the set of shortest paths found select the shortest path with minimum physical length. If more than one path is tied for being "shortest", then use the following set of priorities to select a pair (u, v). Node v will be the destination chosen for inclusion in TREE.

// Note that shortest length path is not necessarily the minimum hop path, //

1. A path with fewer intermediate nodes (or hops) in it is given priority over a path with larger number of intermediate nodes.

   // This step reduces the multiplexing and de-multiplexing losses, which depend on the number of intermediate nodes in a path. //

2. A pair (u, v) which has a multicast capable node at v is given priority over a pair which has a multicast incapable node at v

// Once a multicast incapable node is used, it cannot be used again. So if there is a choice, its better not to use the multicast incapable node till later when one is left with fewer choices (as the resources get exhausted towards the end). //

3. A pair (u_i, v_i), is given priority over a pair (u_j, v_j) if v_j has a higher node degree.

If the contention still remains, then choose a pair at random from among the contending pairs.

// The above priorities are used only when there is contention between destinations to be included into the tree. The contention occurs when more than one pair are "closest". //

Let the selected path be $P(u, v)$. v is the node selected for inclusion into TREE.

}

(4) Check the adjacent nodes of v.

If there exists more than one adjacent node q such that q Є TREE then choose that q which is "closest" to v.

// "closest" means shortest physical distance. //

{

Copy TREE into a temporary tree TEMP_TREE1

Copy TREE into a temporary tree TEMP_TREE2

Add path P (u, v) to tree TEMP_TREE1.

If q is a non-leaf node then add the links (q, v) and (v, q) to the tree TEMP_TREE2

Else add only the link (q, v) to the tree TEMP_TREE2.

Calculate P_{min}= Minimum power among all the destination nodes in TEMP_TREE1.

Calculate $P`_{min}$= minimum power among all the destination nodes in TEMP_TREE2

If ($P`_{min} > P_{min}$) then TREE = TEMP_TREE2

Else TREE = TEMP_TREE1

// We use two temporary trees to choose between two possible paths. One of which is a shortest path between *u* and *v* and the other uses backtracking over the link (*q, v*). We use backtracking only if it is beneficial. //

}

Else add the path P (*u, v*) to TREE and update it.

// See Figure 3.2 for a clearer view of this step.

//

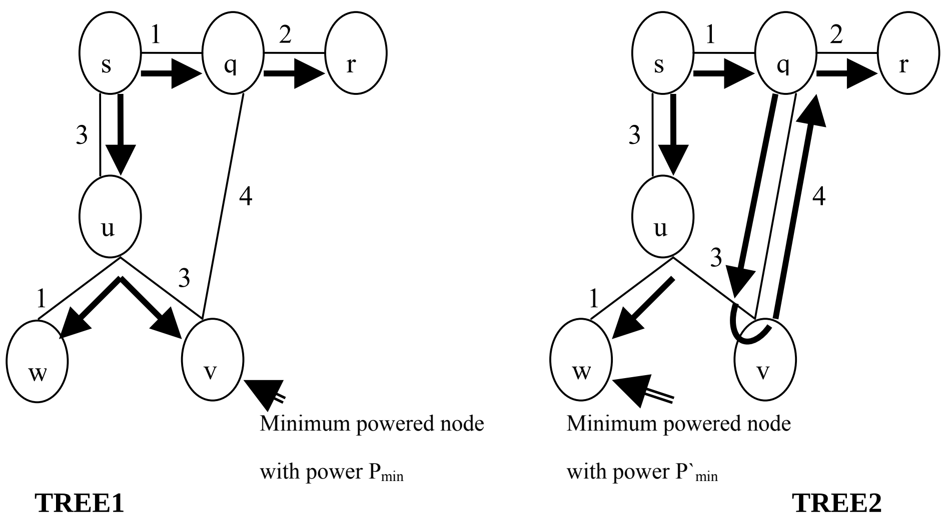


Figure 3.2 Illustration of Step 4 of Algorithm 1.

(5) Check the type (multicast capable or multicast incapable) of every node α which is a new addition to the tree.

{

If *α* is a non leaf MI node or an MC node whose splitting capability is exhausted then make sure that *α* is added to the set USED_SET.

Else add *α* to CANDIDATE_SET.

// If the Splitting capability is exhausted then no further branches can emerge from it and therefore is added to the USED_SET.//

}

(6) Remove *v* from the set DEST_SET.

(7) If DEST_SET $\neq \emptyset$ go to step (3).

// Pruning Starts here//

(8) Calculate the power at each destination node in TREE using the expressions in Chapter 2.

P_{min} = The minimum power among all the destinations in TREE

(9) For each leaf node *c* whose immediate predecessor *b* in TREE has a fanout >1.

{

Copy TREE into a temporary tree TEMP_TREE

Try backtracking on the link (*b, c*). The new route for *c* would be (*b*, *c)*, (*c*, *b)*. So add the link (*c, b)* to TEMP_TREE. This reduces the split degree at *b* by 1.

// The link (*b, c)* is already present in TEMP_TREE. So it is not added to TEMP_TREE //

Calculate $P`_{min}$ = minimum power among all destinations in TEMP_TREE

// $P`_{min}$ is the minimum power in TEMP_TREE if backtracking is used on the link from *b* to *c*. //

If ($P`_{min} > P_{min}$) then

{

TREE=TEMP_TREE

// We are checking to see if backtracking is improving the power at the minimum powered node in TREE. If $P`_{min} > P_{min}$ then backtracking is improving the power. //

}

}

// See Figure 3.3 for illustration.//

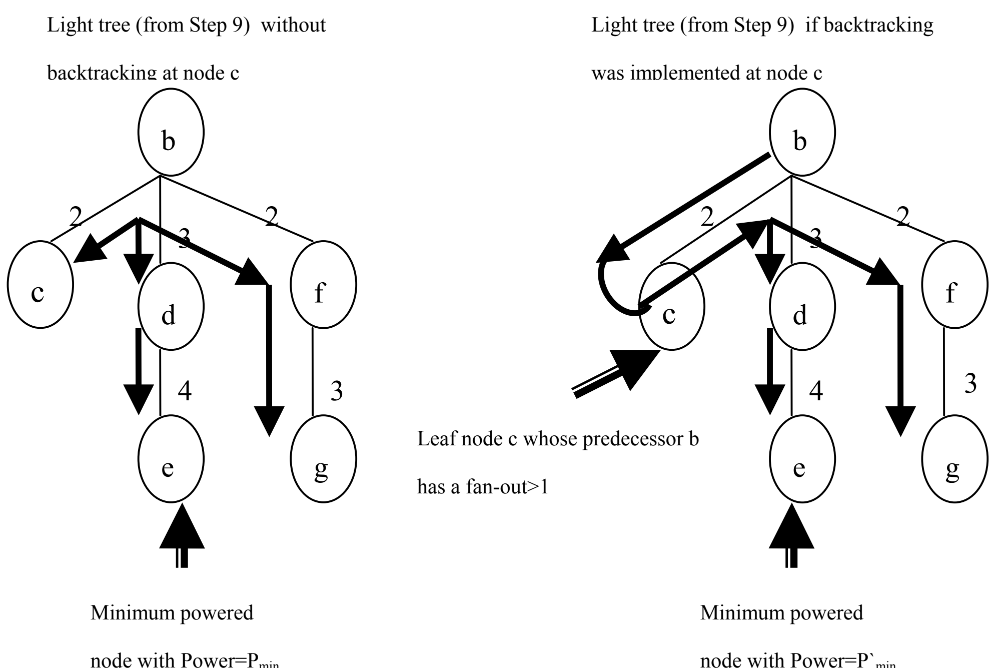


Figure 3.3 Illustration of the terms P_{min} and $P`_{min}$ in Step 9 of Algorithm 1.

// We introduce a second level of pruning next//

(10) Calculate the powers at all destinations in the current TREE.

Find the minimum powered node *z* in TREE.

Follow path P(*s, z*) in reverse order.

Find first node (*x*) in the path from *z* to *s* whose fan-out is greater than 1.

(11) Of all the outgoing branches from *x*, select that branch P (*x, n*) which has least number of destination nodes in it.

Remove all the nodes in the path P (*x, n*) other than *x* from TREE.

// We remove nodes from P (*x, n*) so that the destination(s) on P (*x, z*) get more power. //

Update appropriate sets.

// See Figure 3.4 for illustration. //

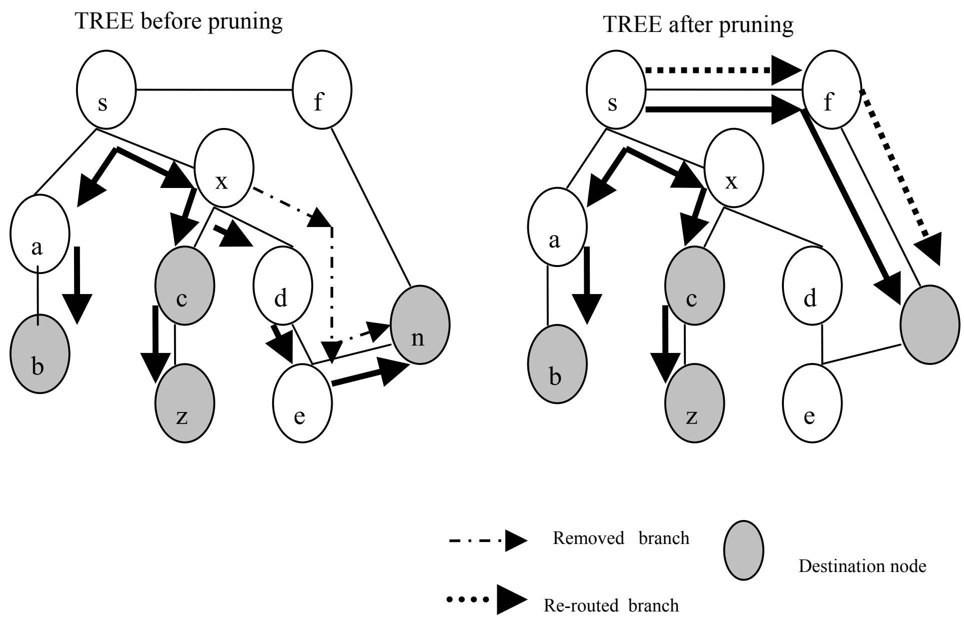


Figure 3.4 Illustration of the pruning adopted in Step 11 of Algorithm 1.

(12) Re-route all the removed destinations to leaf nodes in TREE or the source whichever is giving more power at the minimum powered node in each case. Update the relevant sets and TREE.

If not possible to re-route certain destinations(s) to source or a leaf node, then include it into TREE using Steps 3 to 5

If not possible to reroute any destination *des* then add *des* to DEST_SET and go to Step 14.

(13) Calculate the powers at the destinations in TREE.

P_{min} = The minimum power among all the destinations in TREE.

If the difference between P_{min} of this iteration and P_{min} of the previous iteration is greater than the *iteration threshold* then go to Step 10

Else

// A third level of pruning is tried here. //

{

Prune the backtracked nodes as well and re-route them using the rules in Step 12

// we are trying to find out if re-routing backtracked nodes will improve the power. If it is, then another level of pruning is tried. Otherwise another tree will have to be built.//

Calculate the powers at the destinations in TREE.

P_{min} = The minimum power among all the destinations in TREE.

If (P_{min} < *power threshold*) then

{

Find all the destinations with power< *power threshold.*

For each such destination *t*

{

Follow the path P (*t, s)* in reverse order.

Find first node (*r*) in the path from *t* to *s* whose fan-out is greater than 1.

Of all the outgoing branches from r, select that branch P (*r, n*) that has least number of destination nodes in it.

Remove all the nodes in the path P (*r, n*) other than *x* from TREE.

Add all the destinations on the path (*r, n*) to the set DEST_SET.

Update the appropriate sets

// we are removing all the destinations which are not receiving the adequate power from the tree and building another tree for them. //

}

}

}

// *power threshold* is the minimum power required for the signal to be detected reliably. I*teration threshold* is a value which is used to check the improvement, if any, in the power levels of the destination nodes from the power levels of the previous iteration. If there is no appreciable improvement then there is no sense in repeating the procedure. //

(14) If DEST_SET ≠ ø add TREE to F(*s,* DEST_SET) and go to step 2 to build another tree.

// the forest F(*s*, DEST_SET) is a union of all the already constructed trees. //

}

3.2.1.2 Algorithm 1 Description

The algorithm (which is illustrated through an example in Figure 3.8) basically maintains sets called DEST_SET, USED_SET, CANDIDATES_SET. These sets are defined in Section 3.2.1. F(s, DEST_SET) is the forest which is initialized to ø. The data structure TREE initially consists of just the source *s*, but otherwise consists of all the nodes and links in the multicast tree; CANDIDATE_SET consists of the MC nodes in the tree and the leaf MI nodes in the tree; USED_SET consists of the non leaf MI nodes in the tree. The nodes in this list cannot be used for further tree building as their splitting capacity is just one and it was used to the fullest.

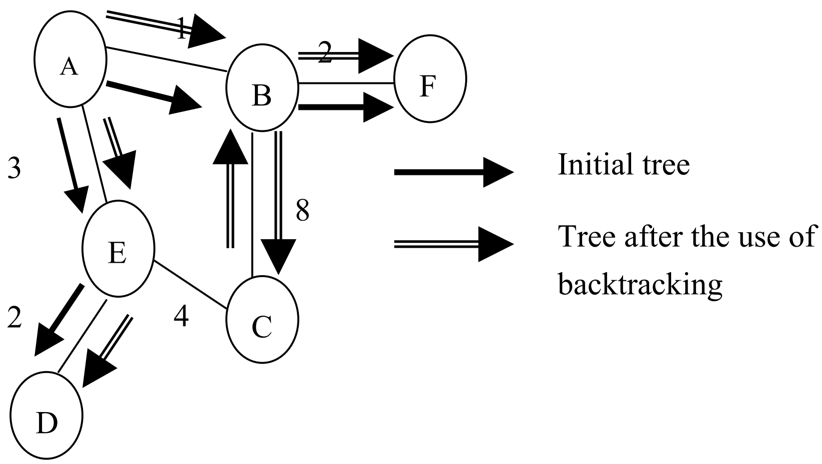


Figure 3.5 Example illustrating Step 3 of Algorithm 1.

In step 3 the closest destination to the tree (considering the fact that some of the nodes in the network are MI) is found as described in Step 2. Once such a destination is found, the path for that destination is checked in the sense that it is compared with the backtracking path (Figure 3.2) in terms of the power at the minimum powered node in each case. This comparison helps if that particular destination causes a split in the tree. For example consider Figure 3.5. In this example nodes *B, A, E, D* are added to the tree one after the other in that order. The last node to be included will be node *C***.** If shortest path is followed, then *C* would be added to the tree at node *E*. This causes a split at node E thereby reducing the power on the min-powered node (node *C* in this case).

However if *C* is back tracked to *B*, the path would be $B \rightarrow C \rightarrow B \rightarrow A \rightarrow E \rightarrow D$. The value of the power at the minimum powered node could be much larger than if C is attached to E. In such a case it would be better to back track. However, that may not always be the case as explained in Sub-section 3.1.2. If the distance between *B* and *C* is around 20 then backtracking may not be useful. In the former case splitting would produce a much better power distribution than backtracking.

So in Step (4) backtracking is always checked against non-backtracking case and is used only when it's beneficial. Then all the new MI non leaf nodes which are recently added to the tree are moved to the USED _SET and all the nodes in the tree which can accept further branches are added to the set CANDIDATE_SET. The destination node which was included in the tree in the previous step is removed from the set DEST_SET. This procedure is repeated till all the destinations are added to the tree. If it's not possible to include all the destinations in a single light tree, then more than one light tree is built to accommodate all the destinations. Before building another light tree the present tree is added to the light forest F. Then all the sets other than DEST_SET are re-initialized and another light tree is built. This process of building different light trees within the same light forest continues till all the destination nodes are included into the forest F.

Once the tree is built, the power at every destination node is calculated. Backtracking is used on all leaf nodes attached to nodes which have splits (Section 3.1.2, Figure 3.3). It should be noted that backtracking is not used for all leaf nodes but only for those whose predecessors have splits. The rationale for using backtracking only for certain leaf nodes may ensure that the attenuation losses (which increase with backtracking) are kept under control.

Pruning is implemented starting with Step 8 (Figure 3.4). The minimum powered node is determined and is traced back towards the source till one reaches the first split node on the path. Once a split node *x* is reached, all the branches that emerge from *x* are examined. Among such branches, the branch P *(x, n)* that has the smallest number of destination nodes in it is selected. In case of a tie a branch is chosen at random. The entire branch *P (x, n)* is re-routed using Step12.

This procedure is repeated iteratively till all the destinations get power above the minimum power level. If that is not possible, then the loop will terminate when there is no major difference (in terms of the power at the minimum powered node) between any two successive iterations. Another level of pruning is tried now to check if pruning and re-routing of the backtracked nodes will improve the power. If it is improving then the backtracked nodes are also pruned. Otherwise the algorithm stops pruning completely and checks for un-included destinations. If there are any un-included destinations then all the sets except the destination list are re-initialized and another tree is built. This process is repeated till all the destinations are included in the light forest.

3.2.2.1 Algorithm 2

In the following subsection we introduce Algorithm2 which may be stated as follows

Begin

{

(1) F(*s*, DEST_SET) = ø; DEST_SET = All destinations in the multicast group.

(2) TREE = ø; USED_MI_SET = ø; CANDIDATE_SET= {*s*}.

(3) For each node pair (*u*`, *v*`) such that *u*` Є CANDIDATE_SET and *v*` Є DEST_SET, try to find the shortest physical length path P`(*u*`, *v*`) such that no node on this path is in

USED_SET. If no such path exists go to Step 8

Else

{

From the set of shortest paths found select the shortest path with minimum physical length. If more than one path is tied for being "shortest", then use the following set of priorities to select a pair (*u, v*). Node *v* will be the destination chosen for inclusion in TREE.

// Note that shortest length path is not necessarily the minimum hop path, //

4. A path with fewer intermediate nodes (or hops) in it is given priority over a path with larger number of intermediate nodes.

   // This step reduces the multiplexing and de-multiplexing losses, which depend on the number of intermediate nodes in a path. //

5. A pair (*u, v*) which has a multicast capable node at *v* is given priority over a pair which has a multicast incapable node at *v*

   // Once a multicast incapable node is used, it cannot be used again. So if there is a choice, its better not to use the multicast incapable node till later when one is left with fewer choices (as the resources get exhausted towards the end). //

6. A pair (u_i, v_i), is given priority over a pair (u_j, v_j) if v_j has a higher node degree.

   If the contention still remains, then choose a pair at random from among the contending pairs.

// The above priorities are used only when there is contention between destinations to be included into the tree. The contention occurs when more than one pair are "closest". //

Let the selected path be *P* (*u, v*). *v* is the node selected for inclusion into TREE.

}

(4) Add the path P (*u, v*) to TREE and update it.

(5) Check the type (multicast capable or multicast incapable) of every node α which is a new addition to the tree.

{

If α is a non leaf MI node or an MC node whose splitting capability is exhausted then make sure that α is added to the set USED_SET.

Else add α to CANDIDATE_SET.

// If the Splitting capability is exhausted then no further branches can emerge from it and therefore is added to the USED_SET.//

}

(6) Remove *v* from the set DEST_SET.

(7) If DEST_SET ≠ ø go to step (3).

// Pruning Starts here//

(8) Calculate the power at each destination node in TREE using the expressions in Chapter 2.

P_{min} = The minimum power among all the destinations in TREE

(9) For each leaf node *c* whose immediate predecessor *b* in TREE has a fanout >1.

{

Check the adjacent nodes of *c* for a leaf node *d* Є TREE.

If *d* exists then

Attach *c* to *d* through the link (*d, c*) and remove the link (*b, c*) from TREE. This reduces a split by 1.

}

For each leaf node *c* whose immediate predecessor *b* in TREE has a fanout >1.

{

Copy TREE into a temporary tree TEMP_TREE

Try backtracking on the link (*b, c*). The new route for *c* would be (*b*, *c)*, (*c*, *b)*. So add the link (*c, b)* to TEMP_TREE. This reduces the split degree at *b* by 1.

// The link (*b, c)* is already present in TEMP_TREE. So it is not added to TEMP_TREE //

Calculate $P`_{min}$ = minimum power among all destinations in TEMP_TREE

// $P`_{min}$ is the minimum power in TEMP_TREE if backtracking is used on the link from *b* to *c*. //

If ($P`_{min} > P_{min}$) then

{

TREE=TEMP_TREE

// We are checking to see if backtracking is improving the power at the minimum powered node in TREE. If $P`_{min} > P_{min}$ then backtracking is improving the power. //

}

}

// See Figure 3.3 for illustration.//

// We introduce a second level of pruning next//

(10) Calculate the powers at all destinations in the current TREE.

Find the minimum powered node z in TREE.

Follow path P(s, z) in reverse order.

Find first node (x) in the path from z to s whose fan-out is greater than 1.

(11) Of all the outgoing branches from x, select that branch P (x, n) which has least number of destination nodes in it.

Remove all the nodes in the path P (x, n) other than x from TREE.

// We remove nodes from P (x, n) so that the destination(s) on P (x, z) get more power. //

Update appropriate sets.

// See Figure 3.4 for illustration. //

(12) Re-route all the removed destinations to leaf nodes in TREE or the source whichever is giving more power at the minimum powered node in each case. Update the relevant sets and TREE.

If not possible to re-route certain destinations(s) to source or a leaf node, then include it into TREE using Steps 3 to 5

If not possible to reroute any destination *des* then add *des* to DEST_SET and go to Step 14.

(13) Calculate the powers at the destinations in TREE.

P_{min} = The minimum power among all the destinations in TREE.

If the difference between P_{min} of this iteration and P_{min} of the previous iteration is greater than the *iteration threshold* then go to Step 10

Else

// A third level of pruning is tried here. //

{

Prune the backtracked nodes as well and re-route them using the rules in Step 12

// we are trying to find out if re-routing backtracked nodes will improve the power. If it is, then another level of pruning is tried. Otherwise another tree will have to be built.//

Calculate the powers at the destinations in TREE.

P_{min} = The minimum power among all the destinations in TREE.

If (P_{min} < *power threshold*) then

{

Find all the destinations with power< *power threshold.*

For each such destination *t*

{

Follow the path P (*t, s)* in reverse order.

Find first node (*r*) in the path from *t* to *s* whose fan-out is greater than 1.

Of all the outgoing branches from r, select that branch P (*r, n*) that has least number of destination nodes in it.

Remove all the nodes in the path P (*r, n*) other than *x* from TREE.

Add all the destinations on the path (*r, n*) to the set DEST_SET.

Update the appropriate sets

// we are removing all the destinations which are not receiving the adequate power from the tree and building another tree for them. //

}

}

}

// *power threshold* is the minimum power required for the signal to be detected reliably. I*teration threshold* is a value which is used to check the improvement, if any, in the power levels of the destination nodes from the power levels of the previous iteration. If there is no appreciable improvement then there is no sense in repeating the procedure. //

(14) If DEST_SET ≠ ø add TREE to F(*s,* DEST_SET) and go to step 2 to build another tree.

// the forest F(*s*, DEST_SET) is a union of all the already constructed trees. //

}

3.2.2.2 Algorithm 2 Description

This algorithm is very similar to Algorithm 1 except for a few steps. Unlike Algorithm1 this algorithm does not use the concept of backtracking while building the tree. It just uses the shortest path tree and then prunes it. Before the algorithm starts pruning, it checks all the leaf nodes *c* in TREE whose immediate predecessor *b* has a fanout >1. If there exists a node *d* Є TREE adjacent to *c*, then *c* is routed to *d* and the link (*b, c*) is removed from TREE. This step is unique to this algorithm. This step removes a split from the node *b* and routes it to a leaf node. So in this step we are essentially removing a split from TREE if possible.

A similar pruning procedure as in Algorithm 1 is adopted in this algorithm. However due to the lack of excessive backtracking, this algorithm is relatively less complex than Algorithm1. This algorithm performs better than Algorithm 1 in most cases as will be shown in Chapter 4.

3.2.3.1 Algorithm 3

In the following subsection we introduce Algorithm 3 which may be stated as follows

Begin

{

(1) F(*s*, DEST_SET) = ø; DEST_SET = All destinations in the multicast group.

(2) TREE = ø; USED_MI_SET = ø; CANDIDATE_SET= {*s*}.

(3) Selected path= ø

Copy TREE to a temporary tree TEMP_TREE

Assume the source s to be a leaf node till its fan-out capability is exhausted.

// This step ensures that the tree is spread in all directions from the source. In this algorithm we try to expand the tree in all possible directions from the source //

(4) For each node pair (*u*`, *v*`) such that *u*` is a leaf node in TREE and *v*` Є DEST_SET, try to find the shortest physical length path P` (*u*`, *v*`) such that no node on this path is in USED_SET or TREE. If no such path exists go to Step 4.

//The shortest path between a leaf node and an un-included destination node may not include a node, which is already in the tree.//

Else

{

From the set of found paths select that path *P* (*u*, *v*) which when added to TREE will result in largest power at the minimum powered node.

Selected path=P (*u, v*).

Add the selected path to TREE.

P_{min}= Minimum power in TREE

v is the destination that will be included into TREE.

}

(5) For each node pair (*u, v*) such that *u* Є TREE and *v* is the node selected in Step 4 (If a *v* was not selected in Step 4 then v is any node Є DEST_SET), try to find the shortest physical length path P (*u, v*) such that no node on this path is in USED_SET. If no such path exists go to Step 8.

Else

{

From the set of shortest paths found select the shortest path with minimum physical length.

Selected path = P (*q, v*).

Add the selected path to the temporary tree TEMP_TREE

$P``_{min}$=Minimum power among all destinations in TEMP_TREE

If $P``_{min}$>P_{min} or Selected path = ø

{

TREE=TEMP_TREE

// If Selected path = ø it means that no path was selected so far. That means that there is no path from any leaf node to any destination. So the only possible path for v would be the shortest path, which is P (q, v). //

// we are trying to decide between a path which *may have splits* [P (q, v) from TEMP_TREE] and a path which *will not have splits* [P (u, v) from TREE]. Between these two paths we select the path that will give better power at the minimum powered node in the tree in each case. //

}

}

(5) Check the type (multicast capable or multicast incapable) of every node α which is a new addition to the tree.

{

If α is a non leaf MI node or an MC node whose splitting capability is exhausted then make sure that α is added to the set USED_SET.

Else add α to CANDIDATE_SET.

// If the Splitting capability is exhausted then no further branches can emerge from it and therefore is added to the USED_SET.//

}

(6) Remove v from the set DEST_SET.

(7) If DEST_SET ≠ ø go to step (3).

// Pruning Starts here//

(8) Calculate the power at each destination node in TREE using the expressions in Chapter 2.

P_{min} = The minimum power among all the destinations in TREE

(9) For each leaf node *c* whose immediate predecessor *b* in TREE has a fanout >1.

{

Copy TREE into a temporary tree TEMP_TREE

Try backtracking on the link (*b, c*). The new route for *c* would be (*b*, *c)*, (*c*, *b)*. So add the link (*c, b)* to TEMP_TREE. This reduces the split degree at *b* by 1.

// The link (*b, c)* is already present in TEMP_TREE. So it is not added to TEMP_TREE //

Calculate $P`_{min}$ = minimum power among all destinations in TEMP_TREE

// $P`_{min}$ is the minimum power in TEMP_TREE if backtracking is used on the link from *b* to *c*. //

If ($P`_{min} > P_{min}$) then

{

TREE=TEMP_TREE

// We are checking to see if backtracking is improving the power at the minimum powered node in TREE. If $P`_{min} > P_{min}$ then backtracking is improving the power. //

}

}

// See Figure 3.3 for illustration.//

// We introduce a second level of pruning next//

(10) Calculate the powers at all destinations in the current TREE.

Find the minimum powered node *z* in TREE.

Follow path P(*s, z*) in reverse order.

Find first node (*x*) in the path from *z* to *s* whose fan-out is greater than 1.

(11) Of all the outgoing branches from *x*, select that branch P (*x, n*) which has least number of destination nodes in it.

Remove all the nodes in the path P (*x, n*) other than *x* from TREE.

// We remove nodes from P (*x, n*) so that the destination(s) on P (*x, z*) get more power. //

Update appropriate sets.

// See Figure 3.4 for illustration. //

(12) Re-route all the removed destinations to leaf nodes in TREE or the source whichever is giving more power at the minimum powered node in each case. Update the relevant sets and TREE.

If not possible to re-route certain destinations(s) to source or a leaf node, then include it into TREE using Steps 3 to 5

If not possible to reroute any destination *des* then add *des* to DEST_SET and go to Step 14.

(13) Calculate the powers at the destinations in TREE.

P_{min} = The minimum power among all the destinations in TREE.

If the difference between P_{min} of this iteration and P_{min} of the previous iteration is greater than the *iteration threshold* then go to Step 10

Else

// A third level of pruning is tried here. //

{

Prune the backtracked nodes as well and re-route them using the rules in Step 12

// we are trying to find out if re-routing backtracked nodes will improve the power. If it is, then another level of pruning is tried. Otherwise another tree will have to be built.//

Calculate the powers at the destinations in TREE.

P_{min} = The minimum power among all the destinations in TREE.

If (P_{min} < *power threshold*) then

{

Find all the destinations with power< *power threshold.*

For each such destination *t*

{

Follow the path P (*t, s)* in reverse order.

Find first node (*r*) in the path from *t* to *s* whose fan-out is greater than 1.

Of all the outgoing branches from r, select that branch P (*r, n*) that has least number of destination nodes in it.

Remove all the nodes in the path P (*r, n*) other than *x* from TREE.

Add all the destinations on the path (*r, n*) to the set DEST_SET.

Update the appropriate sets

// we are removing all the destinations which are not receiving the adequate power from the tree and building another tree for them. //

}

}

}

// *power threshold* is the minimum power required for the signal to be detected reliably. I*teration threshold* is a value which is used to check the improvement, if any, in the power levels of the destination nodes from the power levels of the previous iteration. If there is no appreciable improvement then there is no sense in repeating the procedure. //

(14) If DEST_SET ≠ ø add TREE to F(*s,* DEST_SET) and go to step 2 to build another tree.

// the forest F(*s*, DEST_SET) is a union of all the already constructed trees. //

}

3.2.3.2 Algorithm 3 Description

This algorithm is very different from the two previous algorithms. It differs only in Steps 3-4. Otherwise it is similar to the other two algorithms. However these 2 steps completely change the light tree as they are the basic steps used to build the tree.

Algorithm 3 associates a cost called power for each link that needs to be included into TREE. This algorithm tries to spread the tree in all possible directions in such a way that cost is minimized. Algorithm 1 considers two cases before any destination node is included into TREE and chooses the better one. The two choices are:

1. Finding the destination node which is closest to any of the leaf nodes in the tree and attaching it to the tree through the path from that leaf node (This path will always be through a leaf node.) The source is assumed to be a leaf node till its fan-out capability is exhausted. This step was taken to ensure that the tree spreads in all directions.
2. Attaching the destination node selected in the previous case through its shortest path to the tree (which may or not be through a leaf node).

While the first case will not cause splits the second case can cause splits. Thus this algorithm essentially decides between split losses and attenuation losses. It selects a path that contributes the least loss among the two. This procedure is repeated till all the destinations are included into TREE.

Then pruning is used if the power levels are not acceptable. The pruning technique used by this algorithm is the same as the one used in the previous algorithms (Algorithm1 & Algorithm 2). However it is observed that Algorithm 3 requires minimal pruning and produces better results as far as power levels at the destinations are concerned. On the other hand this algorithm is much more complex than the other two algorithms. Moreover the algorithm may consume more network resources. An illustration of this algorithm on an irregular network is given in Figure 3.10.

3.3 Examples illustrating the performance of the designed algorithms

In this section we try out the Member only algorithm [1], the algorithm by Rouskas et.al. [2] and our algorithms on the sample 20 node network Shown in Figure 3.6. In this network the link lengths correspond to weights listed beside the links. In the network the gray colored nodes are the destinations and the dark colored node is the source. Different arrows represent different levels of pruning. The dark arrows represent the tree that was built initially. Remaining arrows show the pruned tree. The power at the source is assumed to be 1. The minimum powered node is found for each algorithm and the value of that minimum power is calculated and shown in the in the figures (beside the minimum powered node). These figures clearly indicate the performance of the three algorithms. While the first two algorithms require pruning the third algorithm doesn’t require any pruning whatsoever in this example. The number of iterations of the Rouskas algorithm is unknown so we just try it out for two iterations (the maximum number of iterations required for our algorithms in this example). The actual comparisons will be done in the next chapter (Chapter 4).

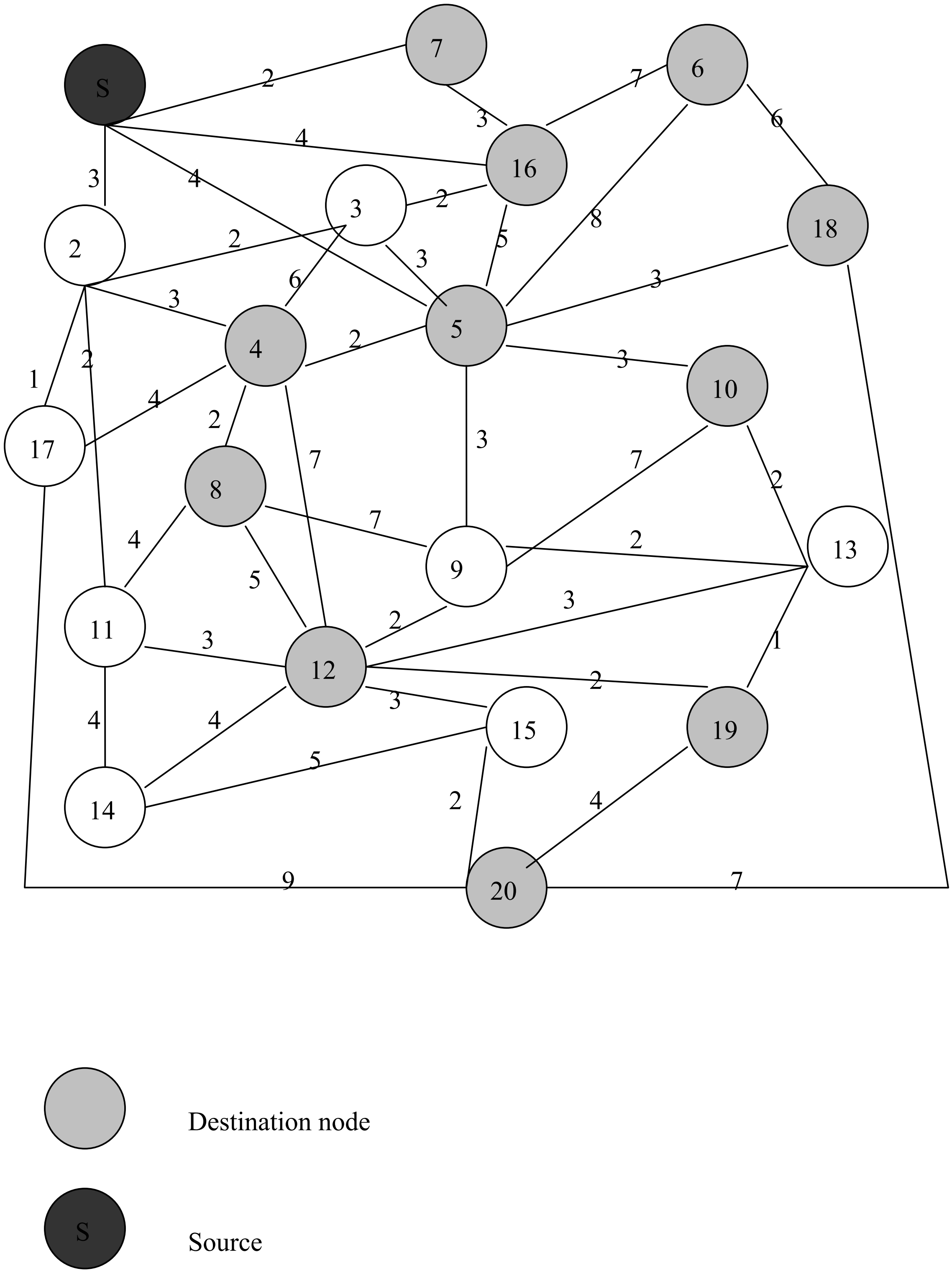


Figure 3.6 A sample network of size 20.

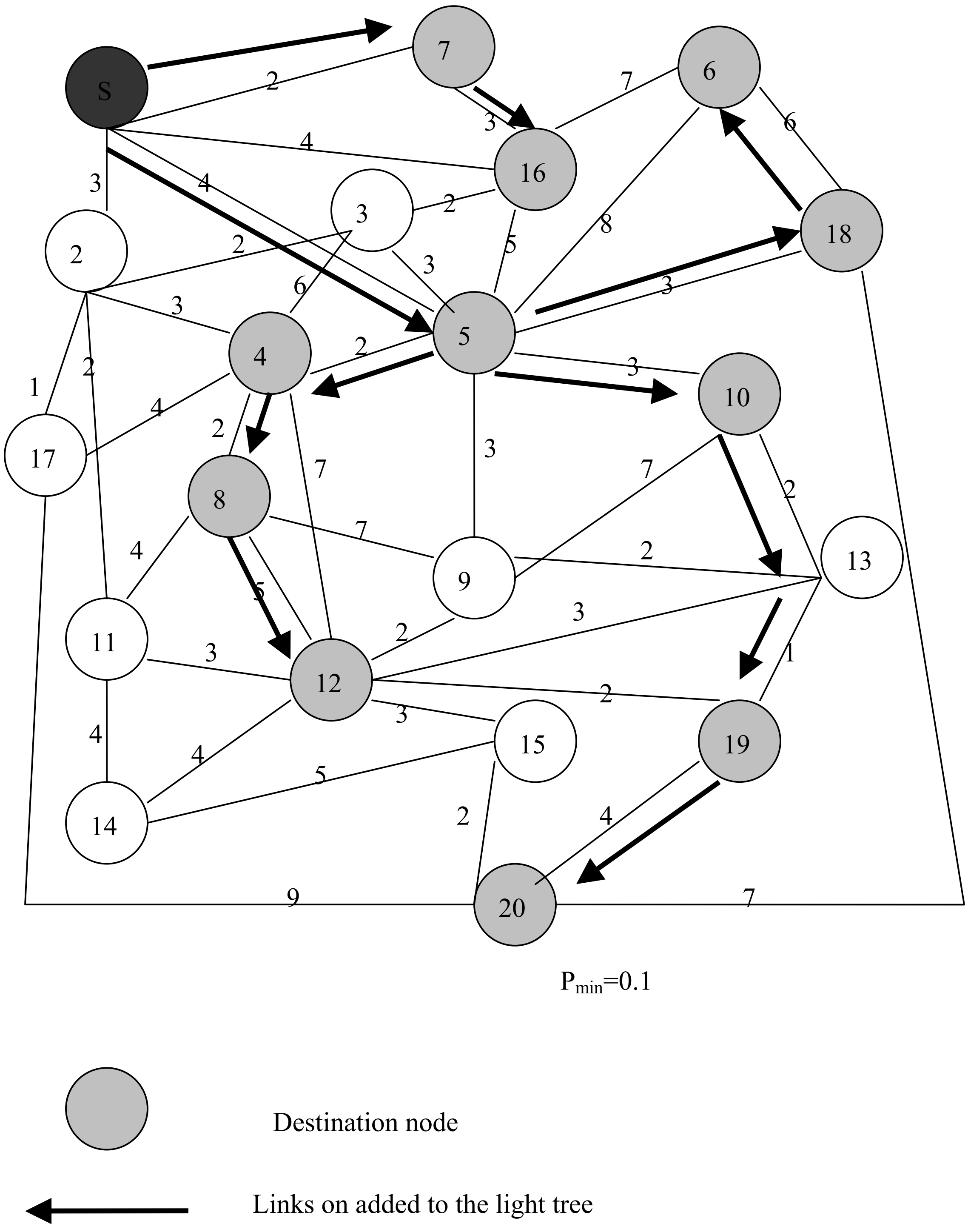


Figure 3.7 The light tree that was built by the Member only algorithm [1] for the network in Figure 3.6.

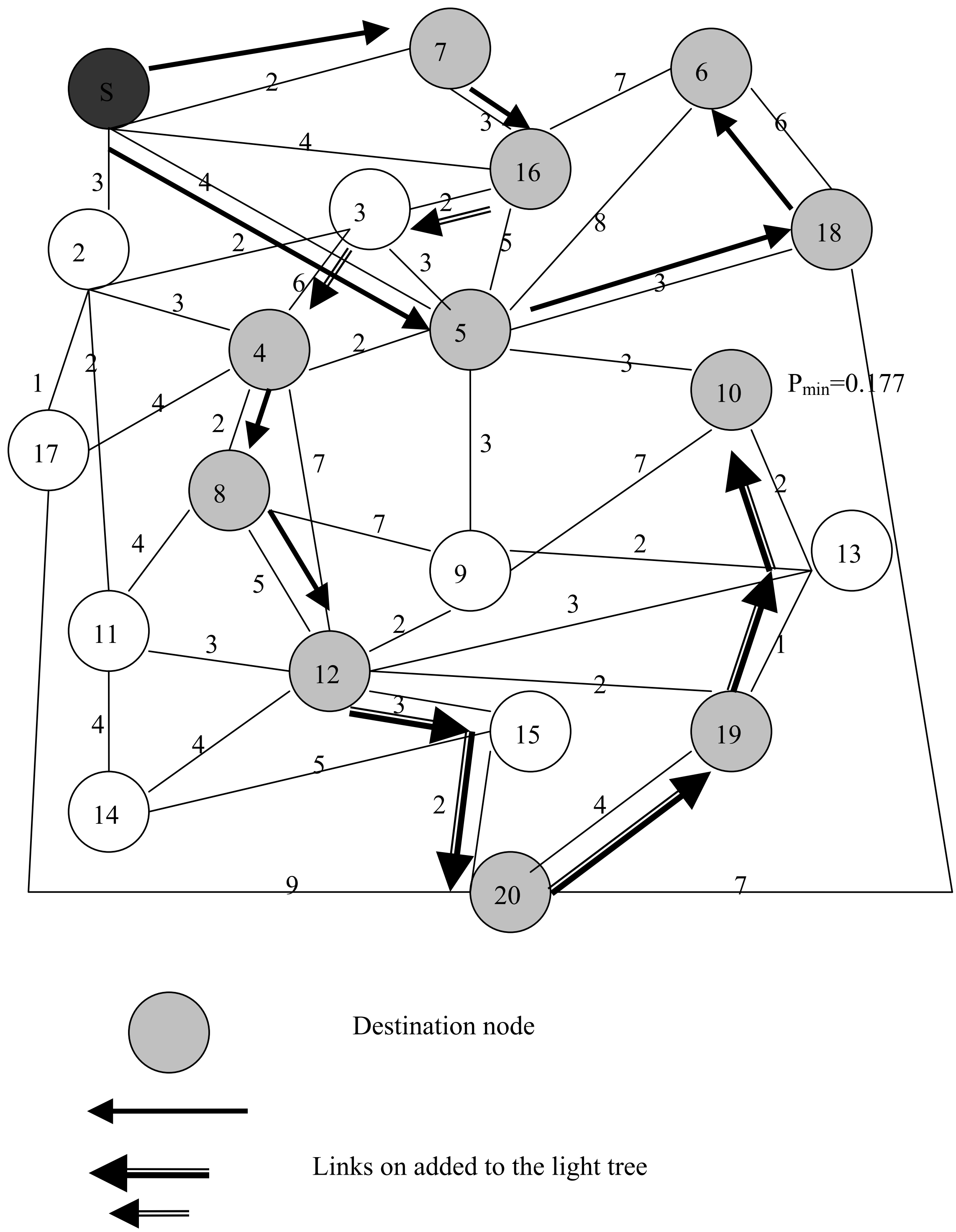


Figure 3.8 The light tree that was built by the algorithm by Rouskas [2] for the network in Figure 3.6.

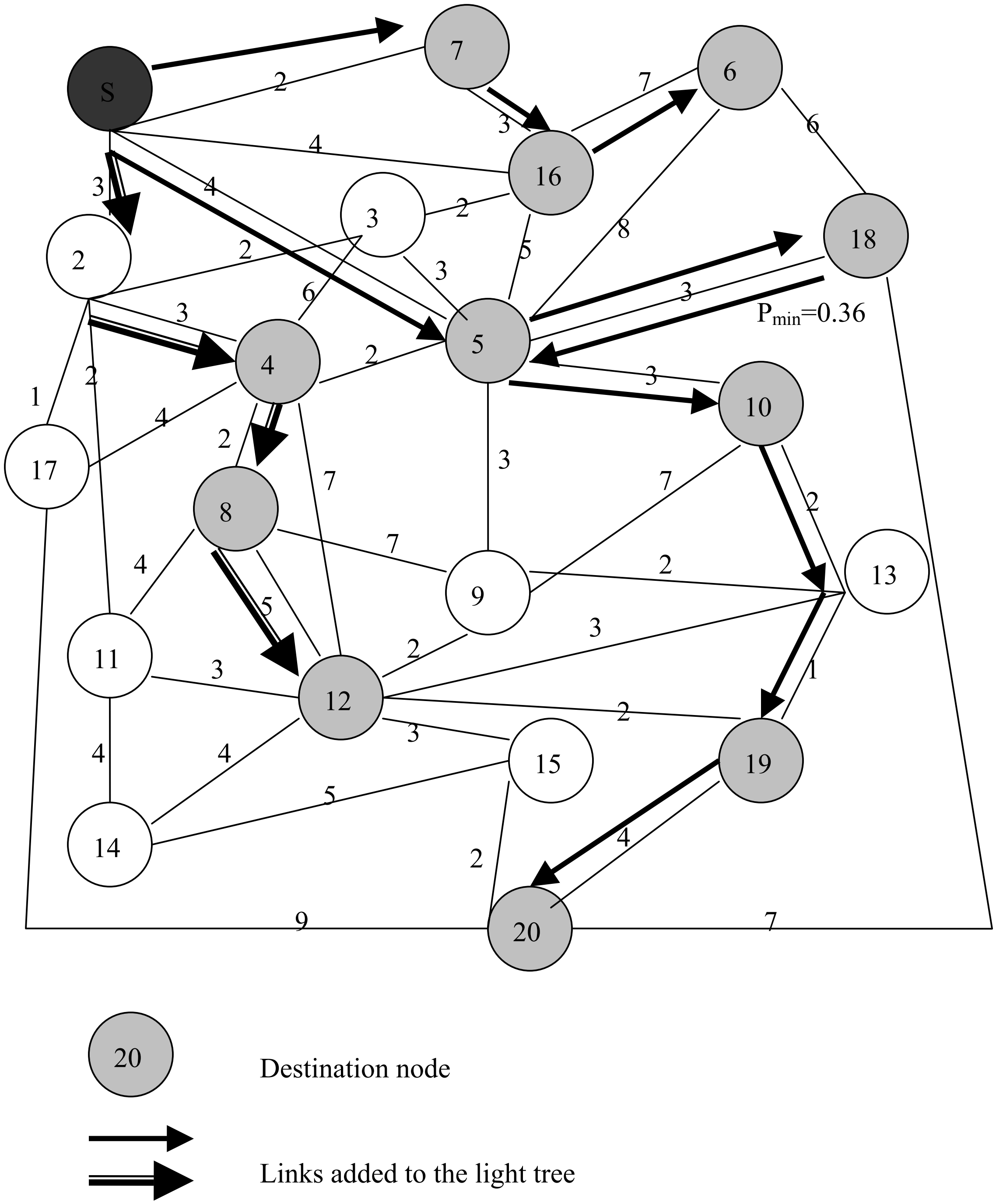


Figure 3.9 The light tree that is built by Algorithm 1 for the network in Figure 3.6.

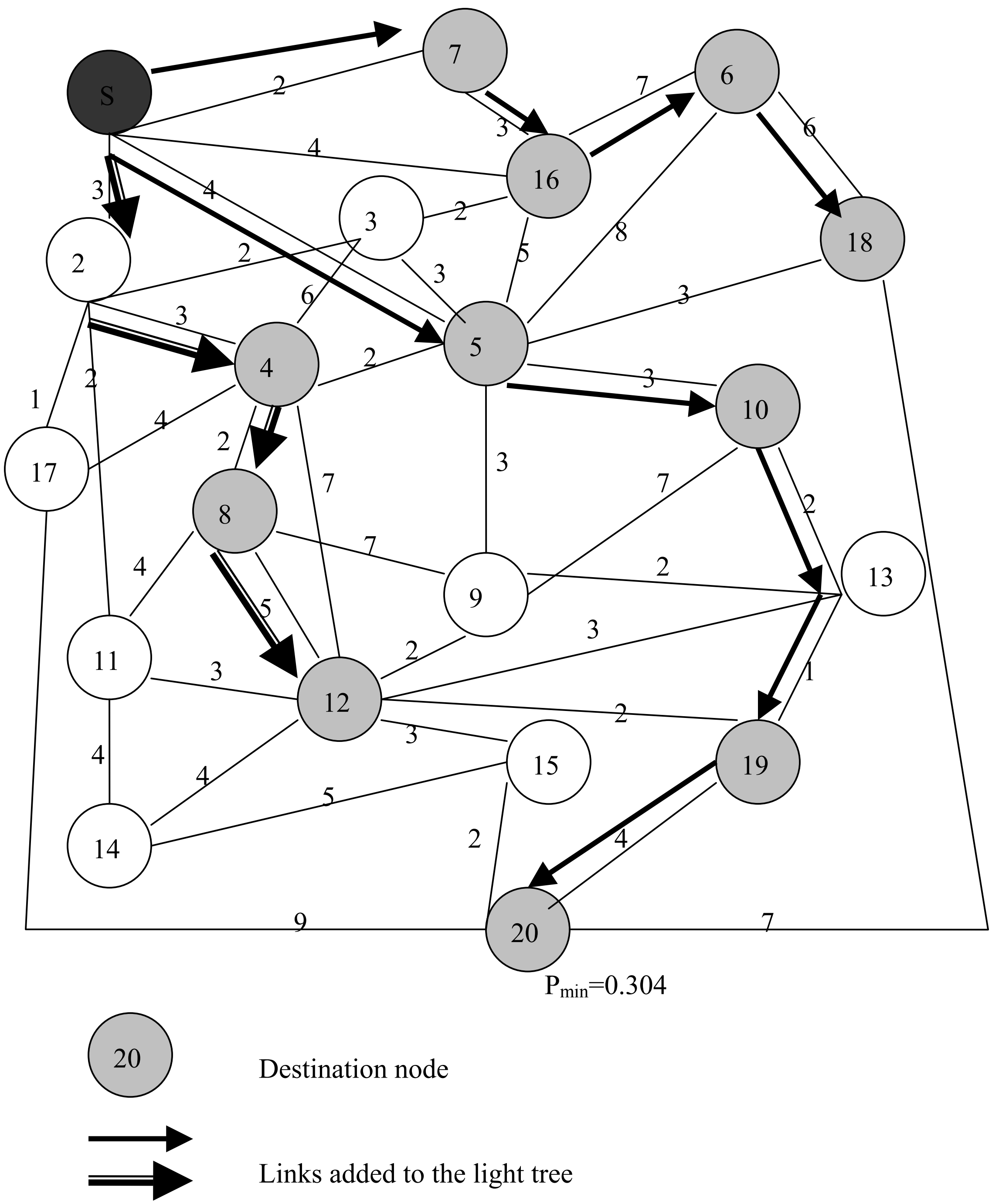


Figure 3.10 The light tree that is built by Algorithm 2 for the network in Figure 3.6.

\\

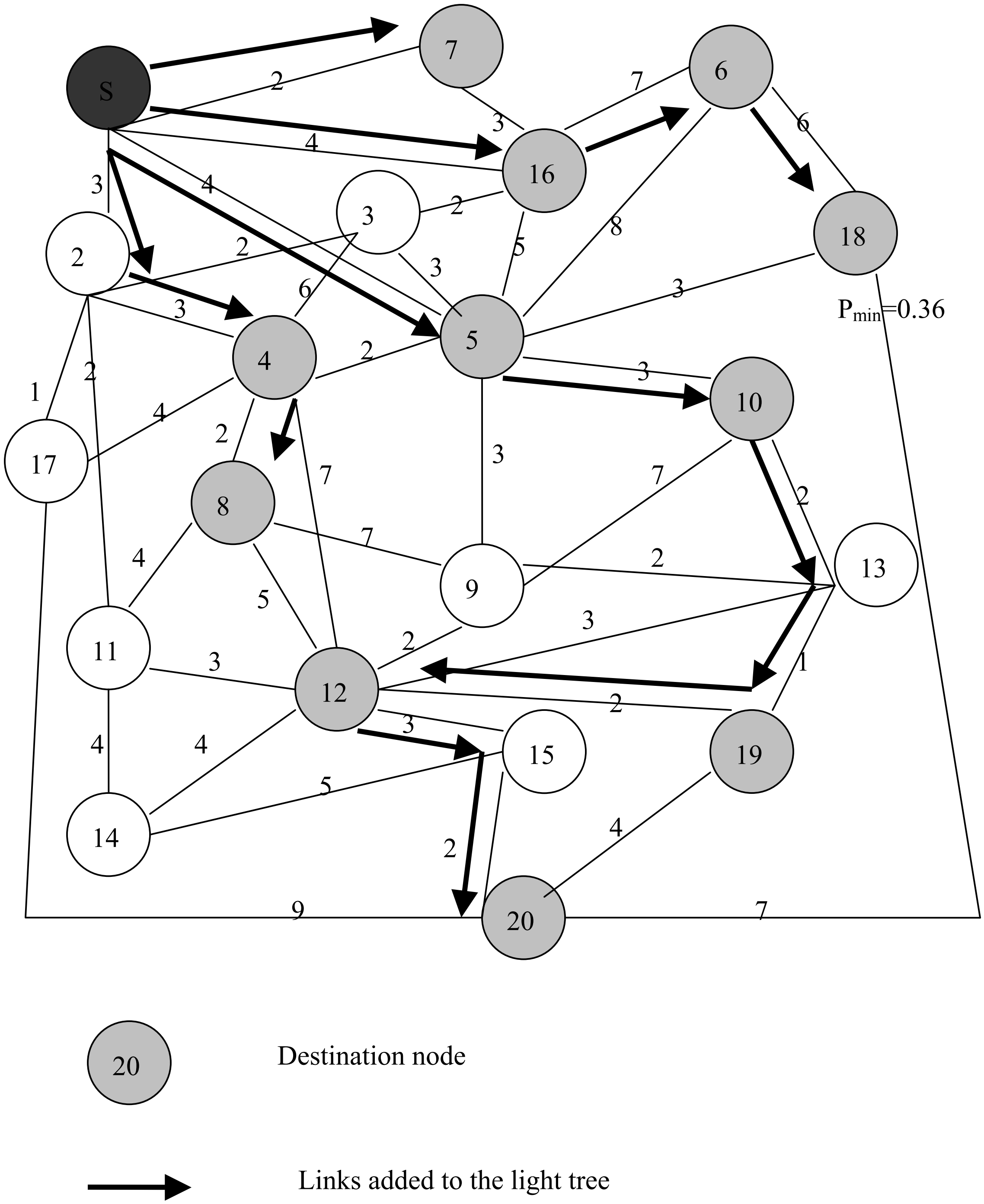


Figure 3.11 The light tree that is built by Algorithm 3 for the network in Figure 3.6.

Chapter 4

Results

In this chapter the three algorithms described in Chapter 3 will be analyzed and compared with one of the already existing algorithms. One of the known algorithms in the literature is due to Rouskas et.al [2]. This algorithm uses the shortest path heuristic to build the light tree. The branch with the minimum powered node is removed from the tree and routed to that branch which has the maximum powered node. However in this process they may not re-route through the shortest path. Thus the algorithm does not take into account the attenuation losses that may be incurred in re-routing the minimum powered node in the tree.

Our algorithms do not have this shortcoming as we always consider the length of the re-routed path. In other words we consider both split losses and attenuation losses. Besides, our re-routing technique is also different. We include a re-routed path only if it improves the minimum power received by a destination in the tree. We try to use the shortest paths as long as much as possible. Moreover, the results presented in [2] clearly indicate that the final tree depends on the chosen number of iterations. This number is not fixed. The authors experiment with it to come up with an optimal number for different networks. The fact that in [2] the results depend strongly on the number of iterations (which is not fixed), makes it impossible to compare our work with the work in [2]. So we compare our results with another algorithm, which comes closest to our work. That algorithm is the Member Only Algorithm reported in [1].

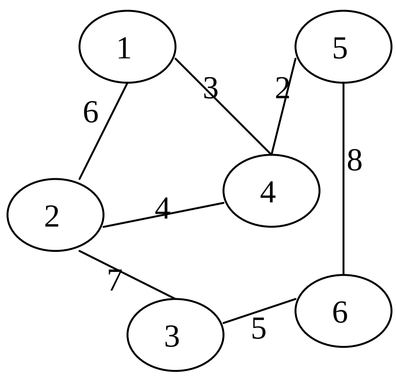
This algorithm initializes the Steiner tree with the shortest path from the source to an arbitrary multicast member. It then repeatedly includes a new member by adding the shortest path between this new destination node to the current partial tree without exhausting the splitting capacity at any node. This entire procedure is repeated until all members of the multicast group have joined the tree. This algorithm was proven to be one of the most efficient multicast algorithms for sparse split networks. Before we report the comparisons, the simulator setup and performance metrics are described in the next section.

4.1 Simulator Setup

We used simulations to evaluate the performance of the light tree routing algorithms on randomly generated irregular networks. Many well known practical networks have a network size of around 20 nodes. Thus the algorithms designed in Chapter 3 are tested on various 20-node networks. These networks are randomly generated. The link lengths of these random networks are scaled down to vary between 1 and 10 units for computational simplicity. The node powers reported in this chapter are normalized. The power at the source is assumed to be 1 and the minimum power in the tree is calculated.

It is assumed that approximately 50% of the nodes in the network have splitting capability. The MC nodes are placed on the most highly connected nodes in the network. A random placement is chosen if there is a contention for the placement of MC nodes.

The software for implementing the algorithms was written in the C language. The program takes the network as an input and gives a light tree as output. The pruning, which involves backtracking, was done manually. The input is given in the form of an adjacency list. Each node on this list is associated with a list of neighbors and the degree of the node. The weight (Normalized link length) of each outgoing edge is also listed. The link lengths are modeled as edge weights. The node type (MC or MI) is also listed in the adjacency list. The output is also given in the form of a forest of trees. The forest will be expressed as a set of adjacency lists, one per tree. A sample adjacency list is given in Figure 4.1.



6 // Number of nodes in the network//
1 2 2 6 4 3 **1**
2 3 1 6 3 7 4 4 **0**
3 2 2 7 6 5 **1**
4 3 1 3 2 4 5 2 **1**
5 2 4 2 6 8 **0**
6 2 3 5 5 8 **0**

The numbers in bold specify the node type.
The first number on each line specifies the node number.
The second number on each line specifies the degree of the node.

A Sample network **Adjacency List**

Figure 4.1 An example of the input to the simulator.

4.2 Performance Metrics

We study the performance of the three algorithms in this section. We will be considering the following three performance metrics.

1. Minimum power in the tree: This measure gives a fair idea of how effective the algorithms are. The power threshold is set to be 10% of the signal strength at the source. If the power of all the nodes is above this threshold, then we will stop the pruning process. If it is not possible for any node(s) to get power above this

threshold we try to bring in the maximum improvement in power at those nodes. So once the final trees are built, we compare the minimum power in each case with the minimum power in the tree produced by the Member Only Algorithm.

2. Percentage increase in minimum power in the tree: This measures the improvement in minimum power (in the tree) when compared to the Member Only Algorithm.
3. Number of links in the light tree: this measure captures the amount of resources consumed by the multicast session as used by each algorithm.

We study the behavior of all algorithms in terms of the three performance metrics as a function of the multicast group size and connectivity. The connectivity is varied from 30 links to 62 links for all 20-node networks. The multicast group size is varied from 4 to 18 for all networks.

The networks being considered are irregular networks. Therefore the results will depend on the network being considered. In order to increase the effectiveness of the results, each metric is tested on 5 different 20 node irregular networks and the results averaged over all 5 networks.

4.3 Minimum Power Metric

The basic aim of this work is to ensure that all the nodes in the network receive a reliably detectable signal. Our algorithms try to improve the power at the minimum powered node. The threshold that we set is 10% i.e. all the destinations should receive a signal which has at least 10% of the signal power produced at the source of the multicast session. Pruning is stopped when the signal strength at the minimum powered node exceeds the threshold of 10%. Normally for a drop-and-continue node, 2-10% of the signal power is tapped at the local node. The remaining signal power is then forwarded to the next nodes(s) in the tree.

4.3.1 Minimum Power vs Multicast Group Size

The results reported here give the power (at the minimum powered node) as a fraction of the power at the source. The group size of the multicast group is varied from 4- 18 nodes and the minimum power in each case (for all the four algorithms) are calculated and plotted. The plots are given in Figures 4.2 through 4.7

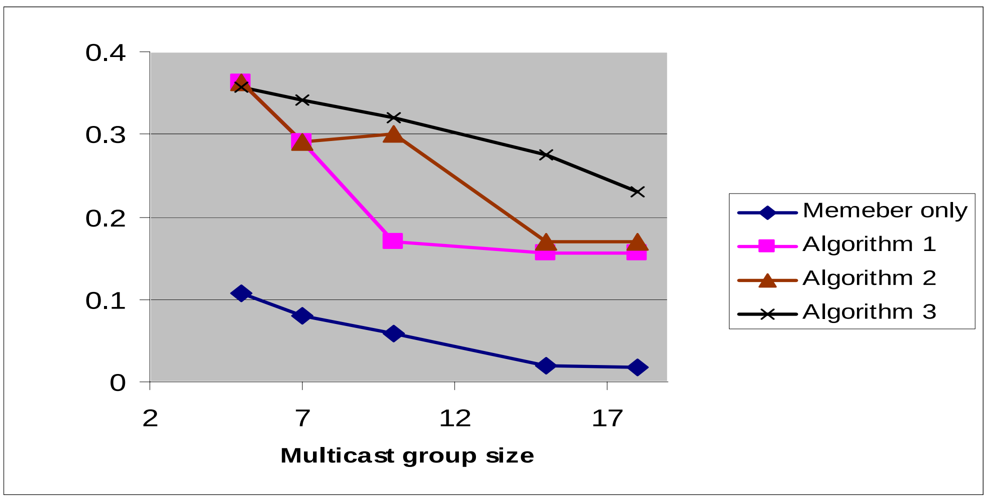


Figure 4.2 Minimum power in the tree vs multicast group size for network 1.

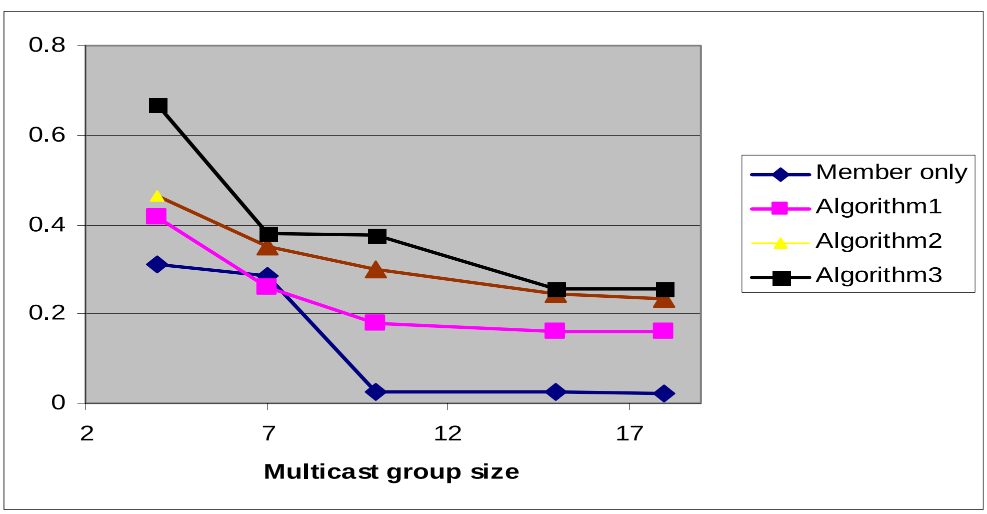


Figure 4.3 Minimum power in the tree vs multicast group size for network 2.

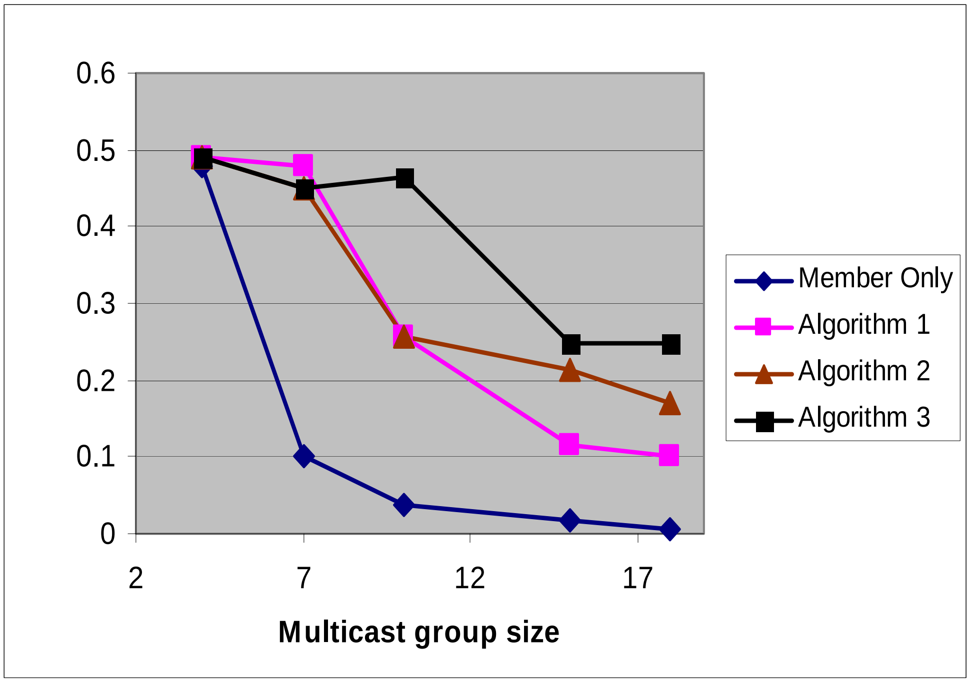


Figure 4.4 Minimum power in the tree vs multicast group size for network 3.

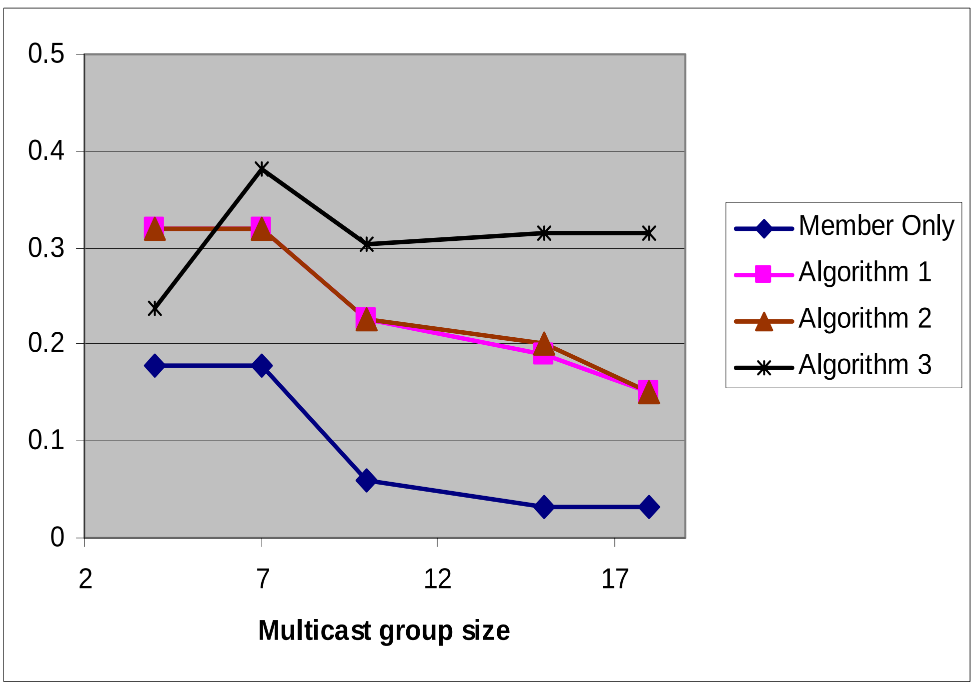


Figure 4.5 Minimum power in the tree vs multicast group size for network 4.

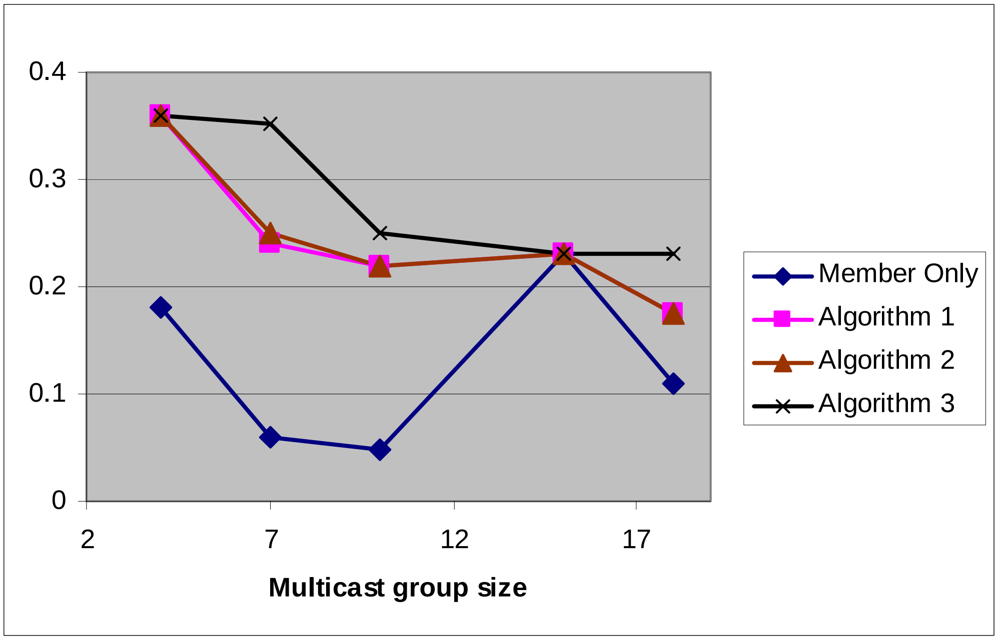


Figure 4.6 Minimum power in the tree vs multicast group size for network 5.

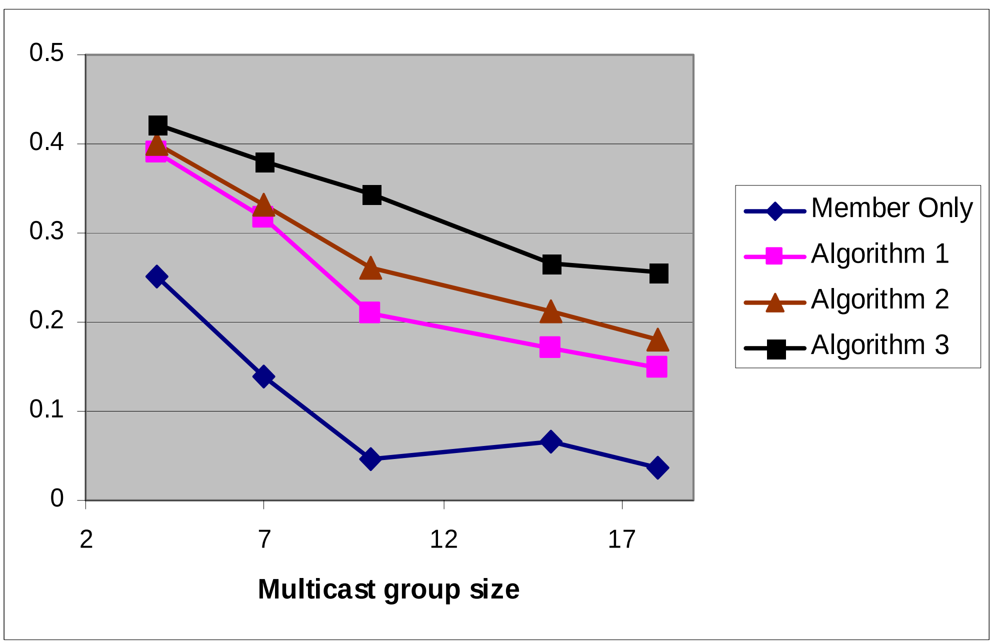


Figure 4.7 Averages of the results in Figures 4.2 through 4.6.

The plots in Figures 4.2 through 4.7 show that all of our algorithms outperformed the Member Only Algorithm. The plots also show that for this metric Algorithm3 works much better than Algorithm2 and Algorithm1. Algorithm3 considers power before including any node into the tree. It includes only those paths that give maximum power from among all the possible paths. This explains why Algorithm 3 gives much better performance over the other three algorithms, which do not consider power (at any node) while building the tree. All algorithms demonstrate good improvement (in terms of power) over the Member Only Algorithm. However there is not much difference between the performance of Algorithm1 and Algorithm 2 in many networks. These two algorithms are very similar and vary only in the amount of backtracking used. Algorithm1 uses excessive backtracking and hence incurs some extra attenuation losses in return. Thus, overall Algorithm 2 outperformed Algorithm 1 when it came to minimum power. Although Algorithm 3 outperforms Algorithm 1 and Algorithm 2, it must be expressed that Algorithms 1 and 2 have much lower complexity and may be preferred in real time situations.

As the group size increases, some additional attenuation losses and split losses are un-avoidable. Therefore, all the graphs show a steady decrease in minimum power with increase in group size. There may be slight anomalies in some graphs. But they are acceptable when one considers the fact that we are dealing with light tree routing in irregular networks. In order to generalize the trends described above, we take the average of the results for 5 networks. The average graph is given in Figure 4.6. Each point on this graph is an average of 5 simultaneous points on 5 different (20 node) networks. The average plot confirms the conclusions made from the previous graphs. The average graph is much smoother and shows the trends described before in a much clearer way. So in summary for this metric our three algorithms show significant improvement over the Member Only Algorithm. Algorithm 3 gives the best results among all four algorithms.

4.3.2 Minimum Power vs Connectivity

Connectivity is another parameter, which affects signal strength. A highly connected network is expected to have much better power distribution than a less connected network. The number of options (paths) for any destination node are higher when connectivity is higher. Pruning will tend to give better results in a highly connected network since as the pruned branches can be re-routed more easily.

The following graphs verify the above assertations. Minimum power in the tree is given as a function of connectivity in the plots we present. It should be stressed that the power in the figures is normalized power. The power at the source is assumed to be 1 and the minimum power in the tree is calculated.

Connectivity is given in terms of the number of links in the network. Like the previous case, we vary the connectivity of the network from 30 links to 62 links for different 20 node networks and plot the individual results. We then take an average of the individual graphs. The plots are given in Figure 4.8 through 4.12.

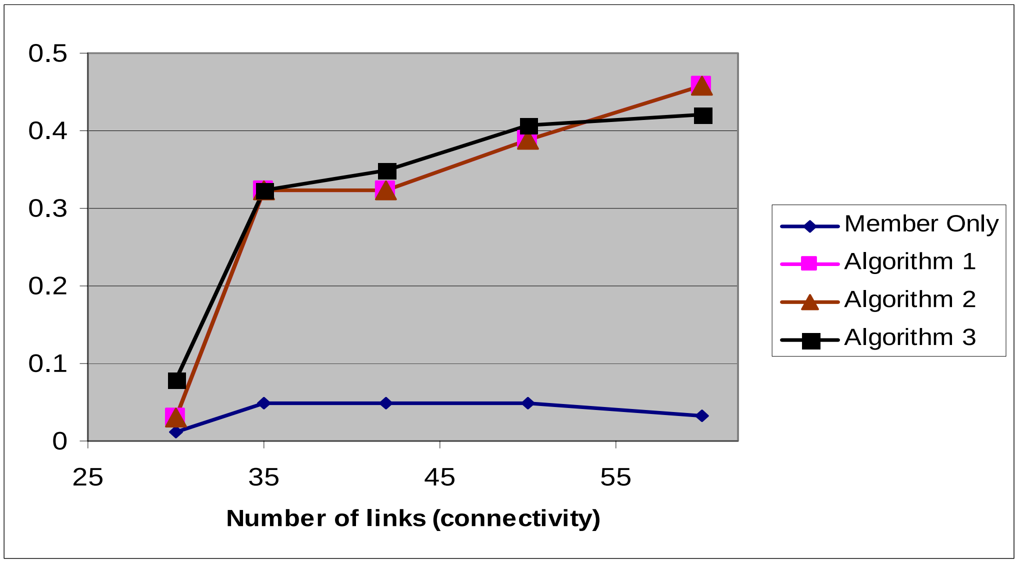


Figure 4.8 Plot of normalized minimum power in the tree vs. Connectivity for network 1.

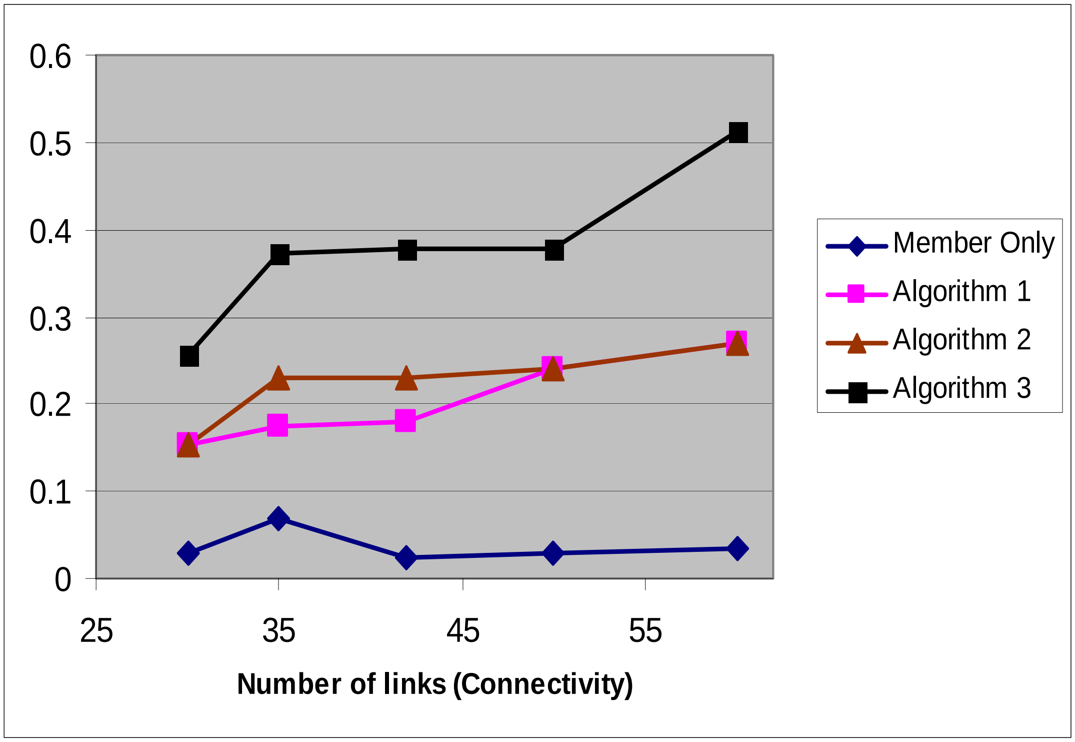


Figure 4.9 Plot of normalized minimum power in the tree vs. Connectivity for network 2.

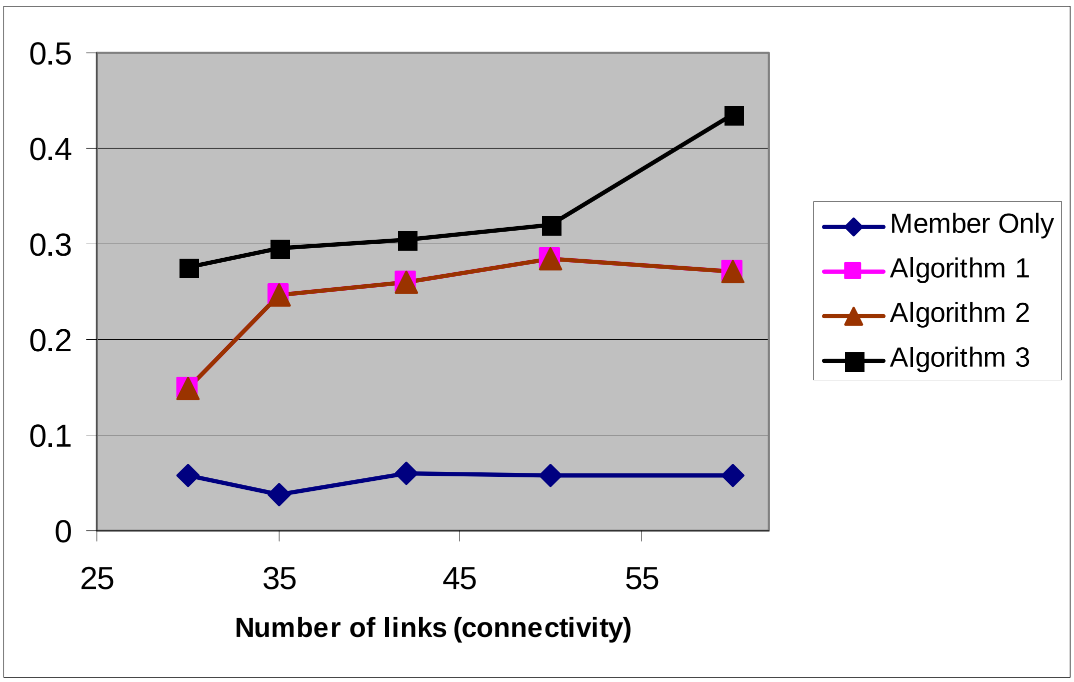


Figure 4.10 Plot of normalized minimum power in the tree vs. Connectivity for network 3.

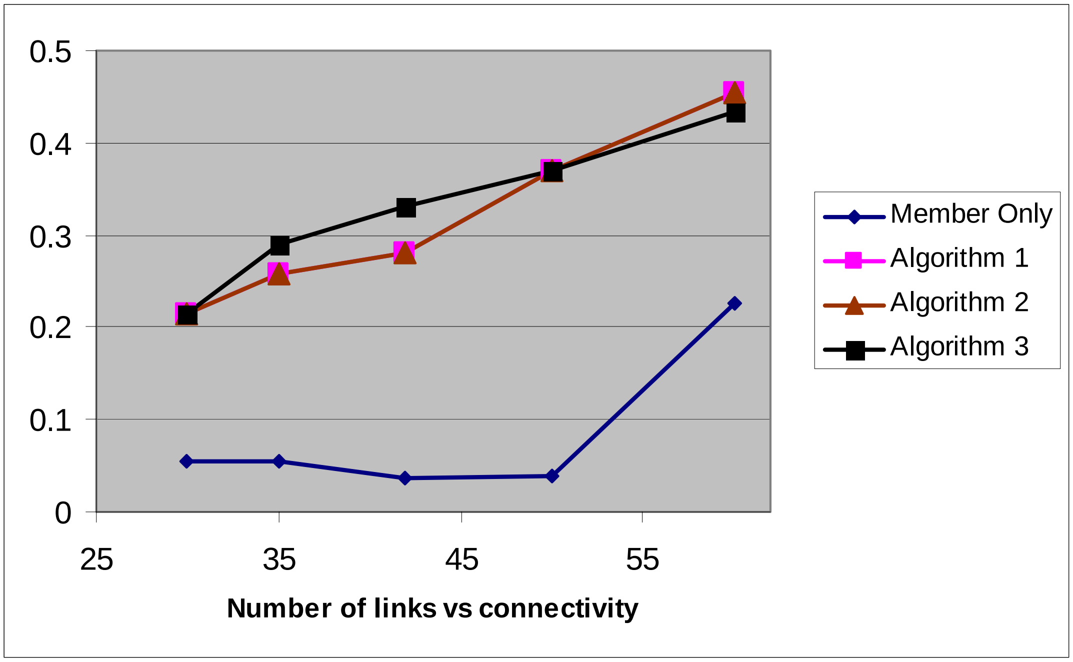


Figure 4.11 Plot of normalized minimum power in the tree vs. Connectivity for network 4.

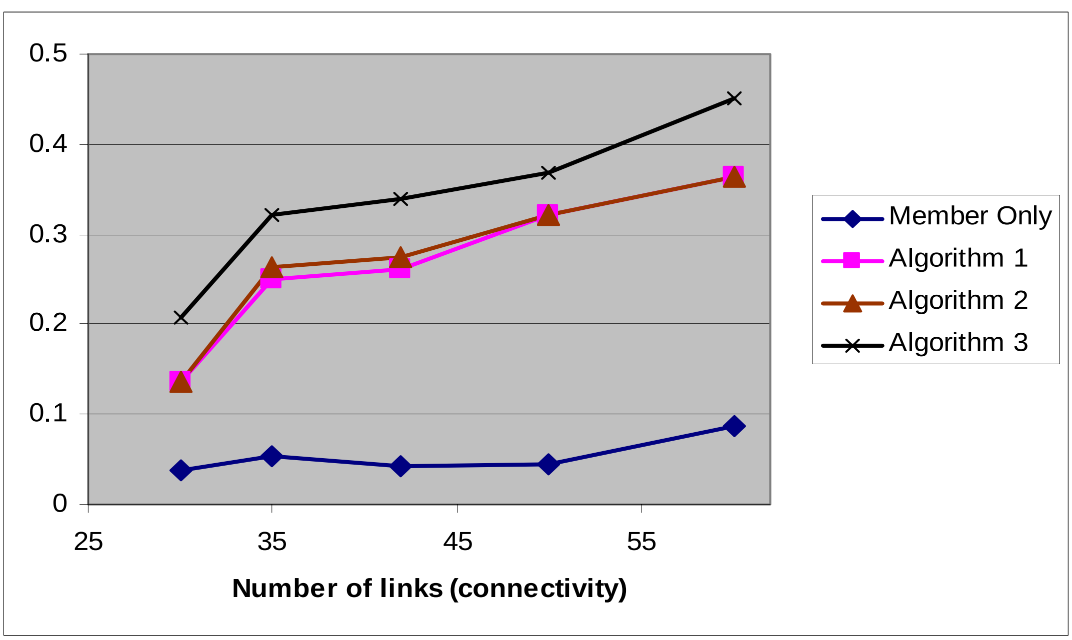


Figure 4.12 Average of the results in Figures 4.8 through 4.11.

The plots show that an increase in connectivity increases the minimum power delivered to the destinations. All three algorithms give much better power performance than the Member Only Algorithm. Algorithm 3 works best in most cases. Algorithm1 and Algorithm2 give similar results in most cases. These two algorithms produce the same light trees in many cases and that explains why some points coincide in the plots.

The improvement gained by Algorithm3 is higher in this case compared to the previous set of plots when the group sizes were varied. The reason for this could be in the fact that re-routing becomes very efficient and effective as the number of links increases. With lower connectivities it is not possible to re-route some of the meagerly powered nodes. Thereby limiting the minimum power. The average of the data in the Figures 4.8 through 4.11 is calculated and plotted in Figure 4.12. This figure further verifies the assertations made thus far. The increase in power is much steadier in the average plot. This plot proves that on an average, our three new algorithms give much better performance than the Member Only Algorithm. The improvement ranges from 100% to 500%.

4.4 Percentage Increase in Minimum Power

In this subsection, we compare the performance of the algorithms with respect to multicast group size and connectivity. We measure the improvement as a percentage increase over the minimum power in the Member Only Algorithm.

4.4.1 Percentage Increase in Minimum Power vs Multicast Group Size

Here the group size is varied from 4 –18 for different 20 node networks and the percentage increase in power over the Member Only Algorithm is calculated for each case and plotted in Figures 4.13 through 4.18.

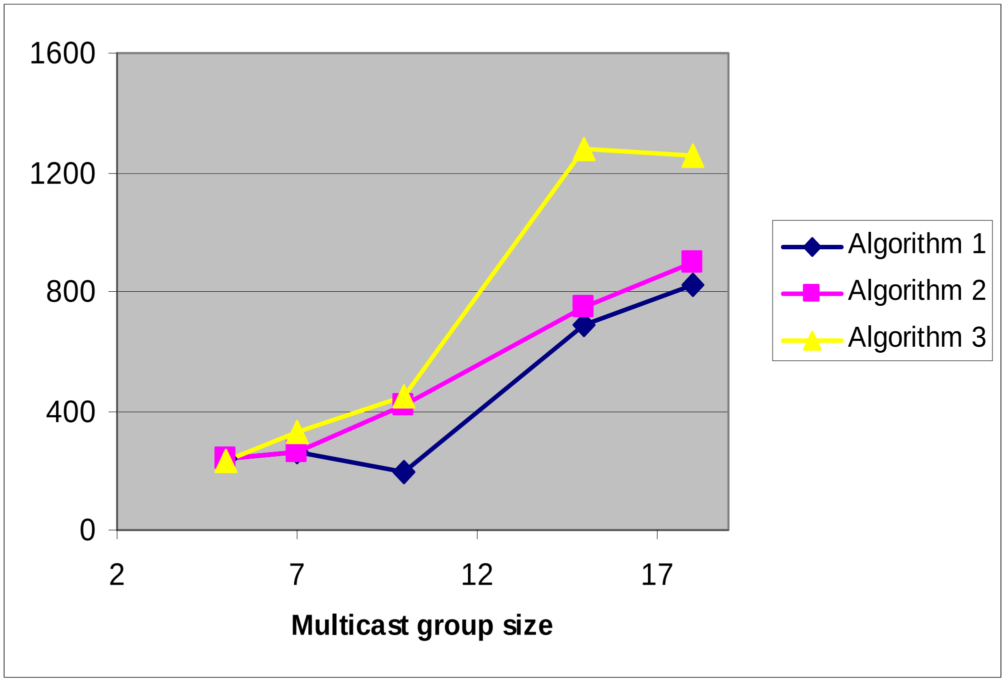


Figure 4.13 Percentage increase in minimum power vs. Multicast group size for network 1.

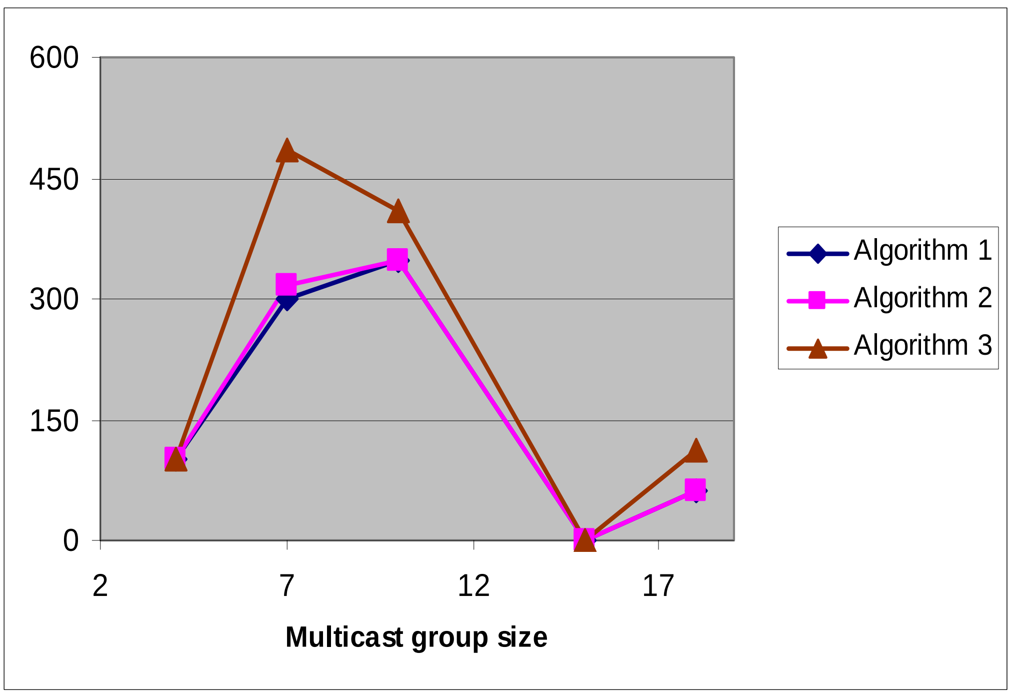


Figure 4.14 Percentage increase in minimum power vs. Multicast group size for network 2.

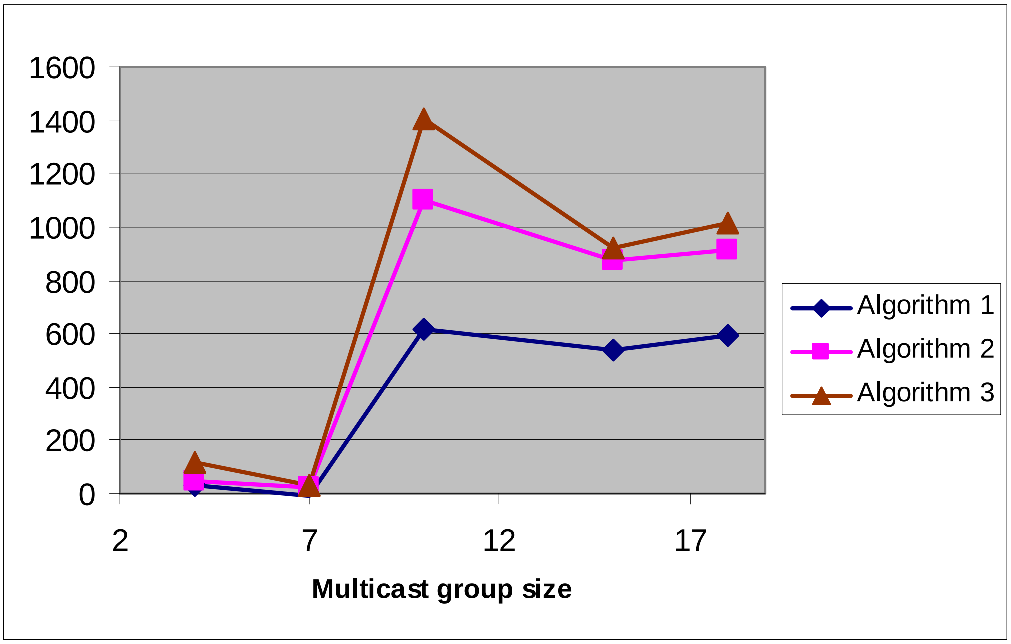


Figure 4.15 Percentage increase in minimum power vs. Multicast group size for network 3.

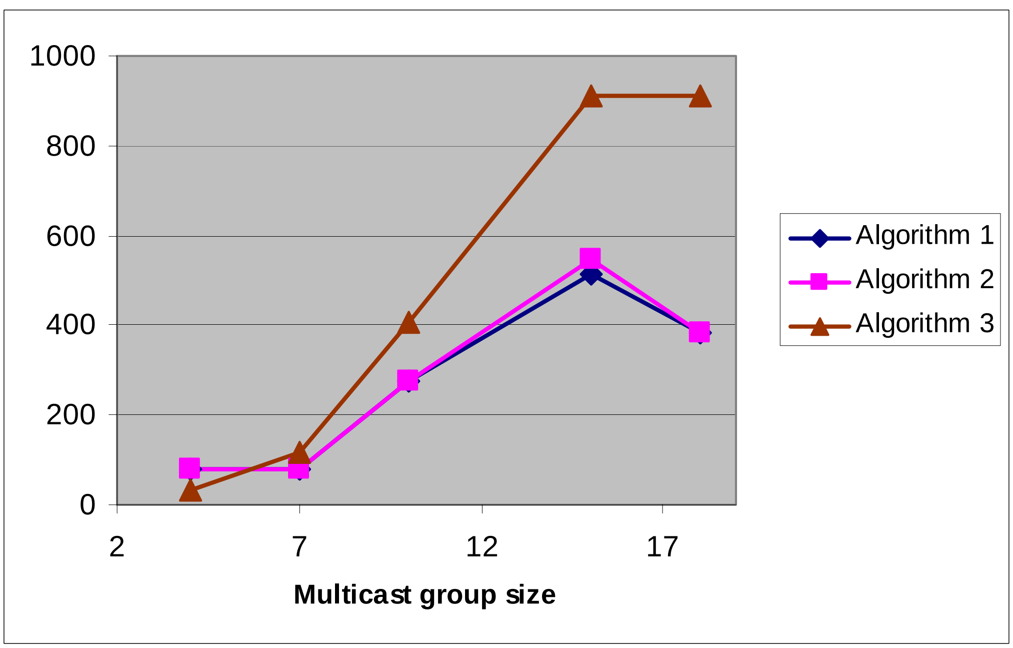


Figure 4.16 Percentage increase in minimum power vs. Multicast group size for network 4.

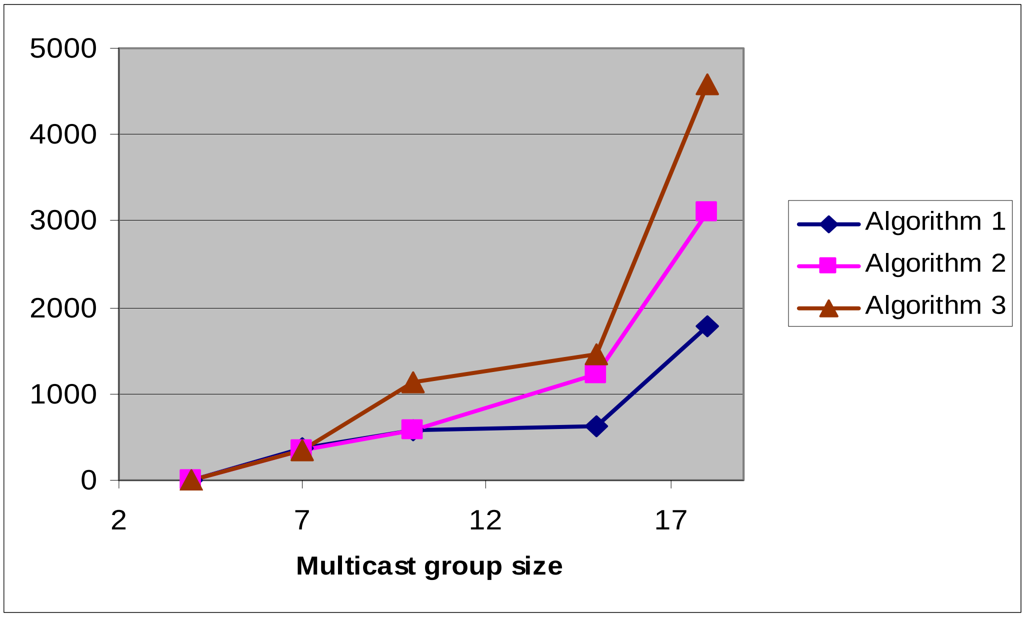


Figure 4.17 Percentage increase in minimum power vs. Multicast group size for network 5.

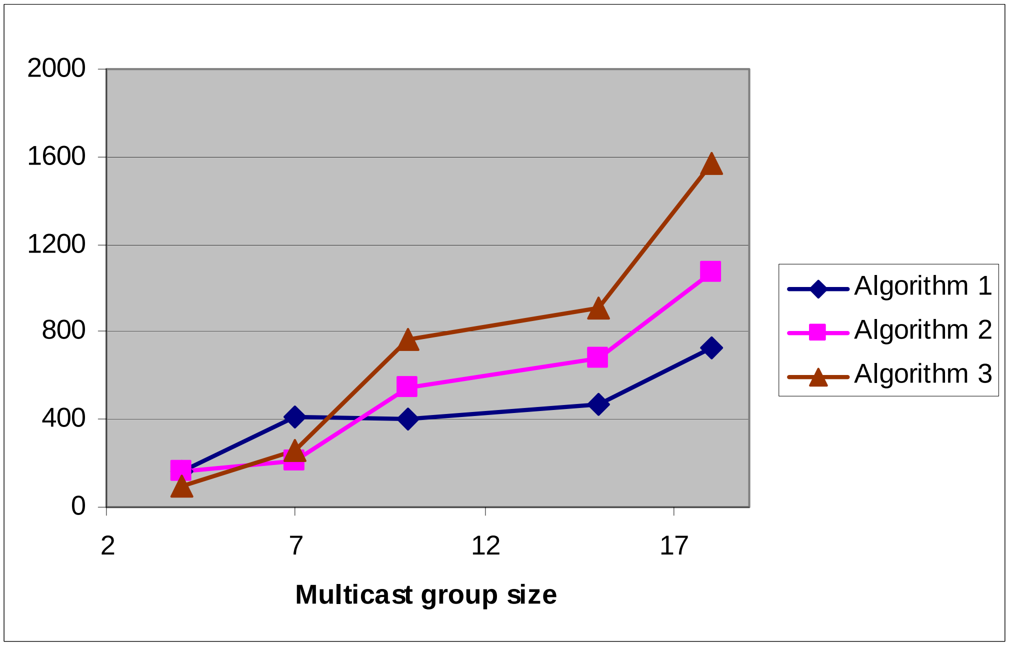


Figure 4.18 Average of the plots in Figures 4.13 through 4.17.

The percentage increase in minimum power plots show that Algorithm 3 works the best for higher multicast group sizes. For smaller group sizes this Algorithm is not distinguishable when compared to Algorithm1 and Algorithm 2. We explain this as follows: If the group size is small then the number of splits in the light tree is minimal. Removal of those splits will not bring in major change in the minimum power. A split is not very disadvantageous in such cases. Algorithm 3 tries to avoid as many splits as possible by choosing a slightly longer route. So, for smaller group sizes the percentage increase in minimum power is not that significant.

From Figures 4.13 through 4.17, it is evident that the percentage increase in power in almost all networks increases rapidly up to a particular point and stops increasing beyond that point. As the group size increases so do the splits in the light tree and the attenuation losses. But attempts to increase the power at the minimum powered nodes will not be very successful as there may be very few free nodes in the network to re-route to.

When group size is small, then number of nodes in the tree will be small compared to the total number of nodes in the network. Thus there will be many nodes not included in the tree. The pruned nodes can be routed to these unincluded nodes, thereby increasing the minimum power in the network. However as group size increases, the number of unincluded nodes decreases. At the same time the number of splits also increases thereby increasing the necessity for pruning and re-routing. However with the unavailability of many unincluded nodes, it may not be possible to re-route the pruned nodes. One is only left with the option of routing these to some other node which is already in the tree, thereby increasing the number of splits. So there may not be a significant improvement in power as we are removing a split from one branch and placing it on another. This conclusion is evident from the percentage increase plots. The average percentage increase plot takes an average of the plots shown in Figures 4.13 through 4.17.

4.4.2 Percentage Increase in Minimum Power vs Connectivity

Here the percentage increase in minimum power in the tree is expressed as a function of the connectivity of the network. The connectivity is represented by the number of links in the network. The number of links is varied from 30 links to 60 links for different 20 node networks and the percentage increase is calculated in each case. The percentage increase is calculated by comparing the minimum power as obtained using Algorithm1, Algorithm2, and Algorithm3 with the Member Only Algorithm. The plots are given in Figures 4.19 through 4.23.

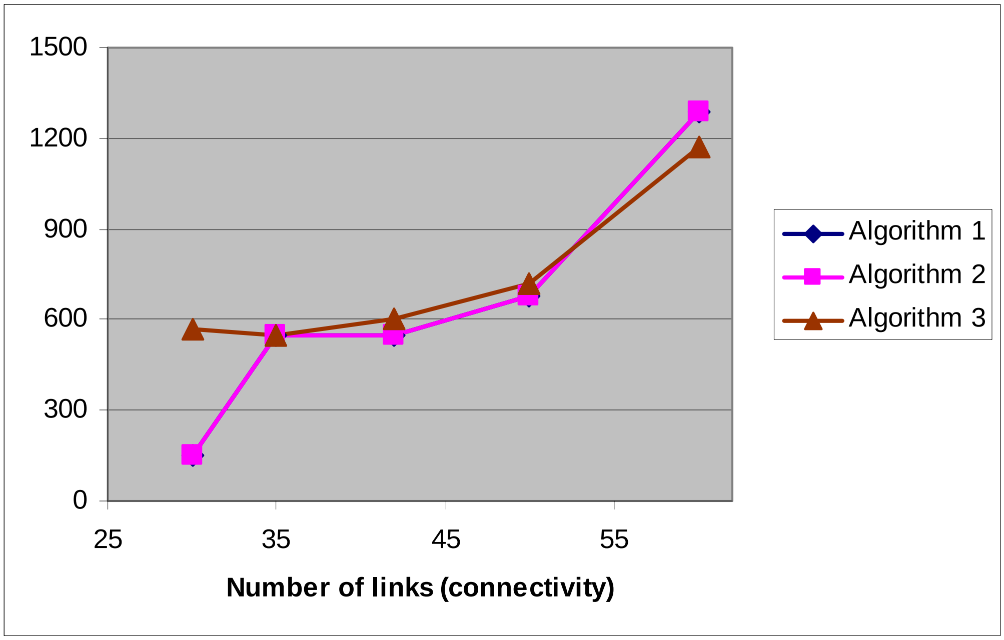


Figure 4.19 Percentage increase in minimum power vs Connectivity for network 1.

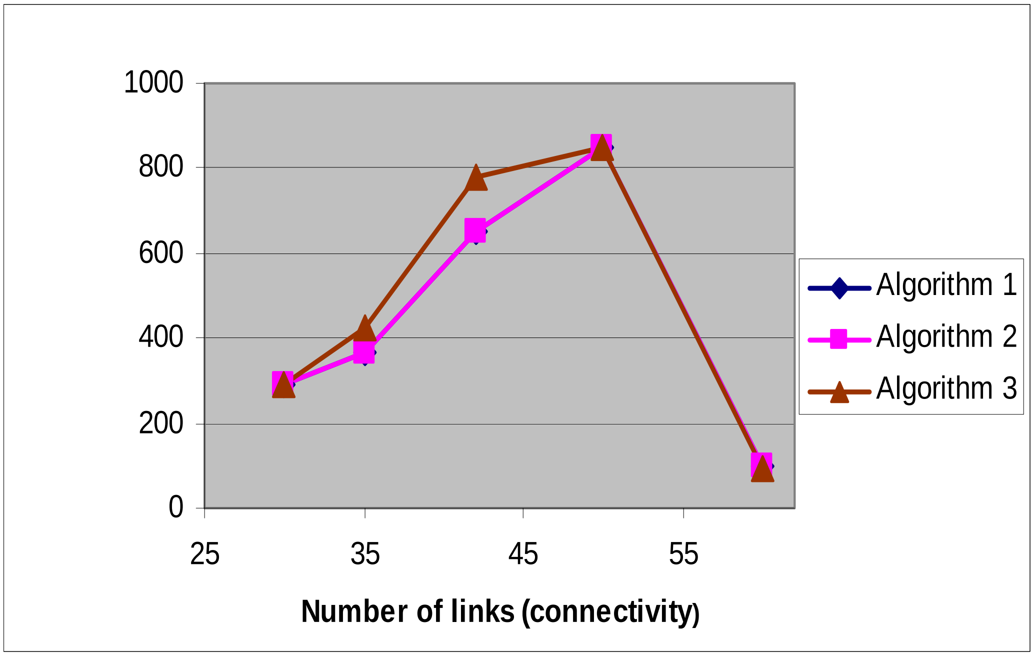


Figure 4.20 Percentage increase in minimum power vs Connectivity for network 2.

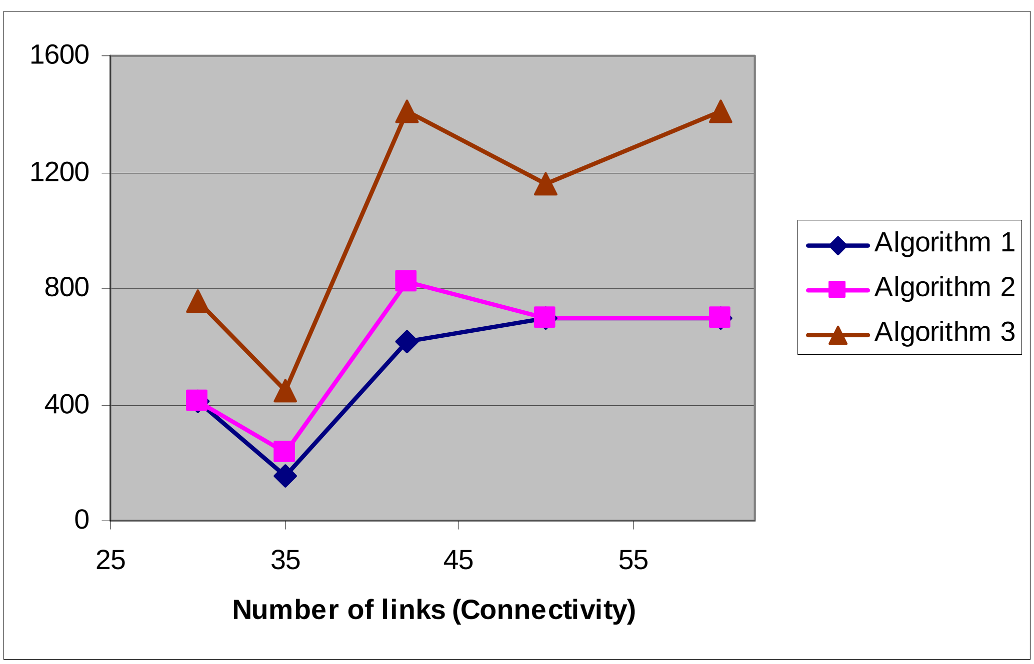


Figure 4.21 Percentage increase in minimum power vs Connectivity for network 3.

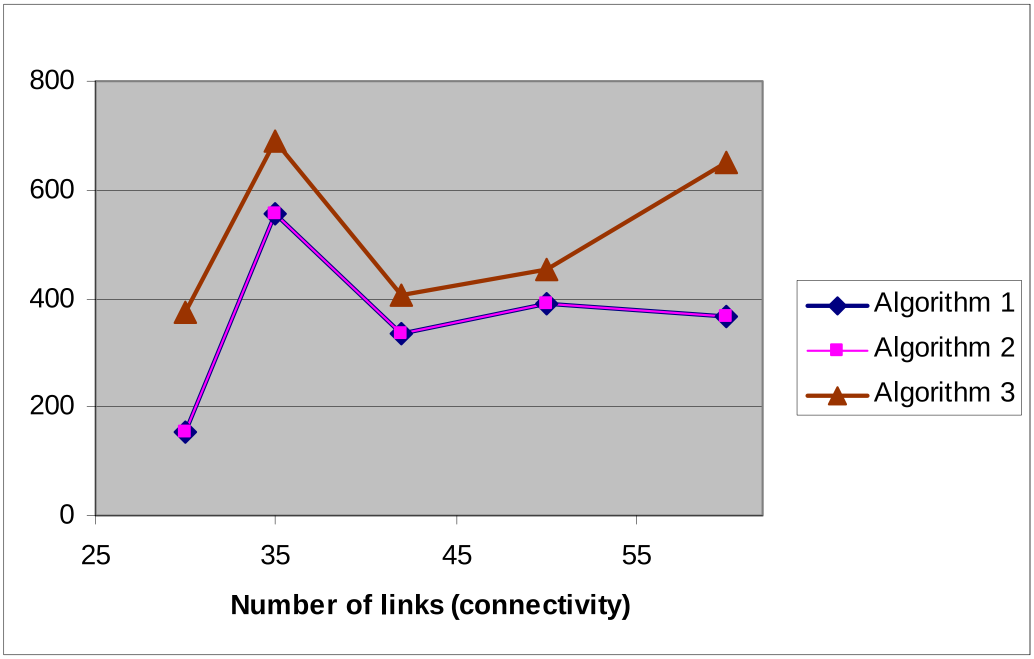


Figure 4.22 Percentage increase in minimum power vs Connectivity for network 4.

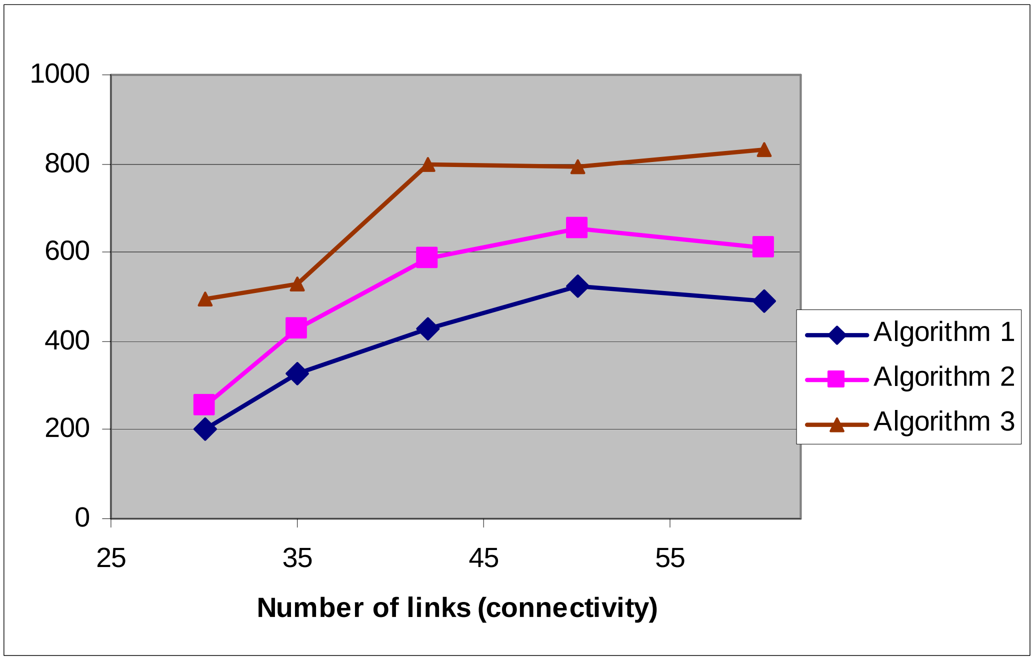


Figure 4.23 Average of the plots in Figure 4.19 through 4.22.

The plots show that as connectivity increases Algorithm 3 outperforms all other algorithms. Algorithm1 and Algorithm2 still perform well when compared to the Member Only Algorithm. Although Algorithm 3 outperforms Algorithm 1 and Algorithm 2, it must be expressed that Algorithms 1 and 2 have much lower complexity and may be preferred in real time situations.

The plots show that there is a steady increase in percentage power till a certain point. However after a certain point the percentage increase decreases. This may be explained as follows: With increase in connectivity, re-routing increases the minimum power in the tree when compared to the Member Only Algorithm. However if the connectivity is increased beyond a certain limit the Member Only Algorithm also starts performing well thereby giving stronger signals at the destinations. Thus the increase in power levels in our algorithm compared to the Member Only Algorithms drops. That explains why most of the plots show an increase initially and a decrease after a certain point. Thus the effectiveness of our algorithms is more profound with low to moderate connectivity's.

4.5 Network Usage Metric

This metric gives a measure of the amount of network resources that will be utilized to perform the multicast. For each multicast session the links in the tree are reserved on one particular wavelength. That wavelength cannot be used on those links for other traffic. So the number of links used is a measure of the amount of network resources used.

We vary the multicast group size for different 20 node networks and measure the number of links used in each case in order to get a clearer view of network usage. It should be noted that each link that underwent backtracking is treated as two links. Then an average graph for results over five networks is again plotted to generalize the results. The multicast group sizes are varied from 4 to 18 and the number of links used by tree is measured. The plots for the results are given in Figures 4.24 through 4.28.

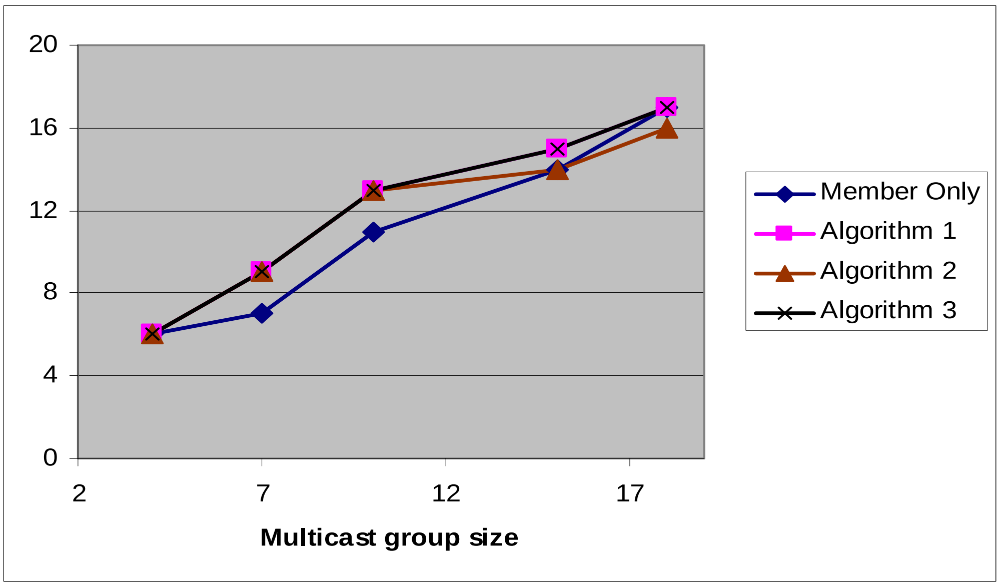


Figure 4.24 No. of links in the tree vs. Multicast group size for Network 1.

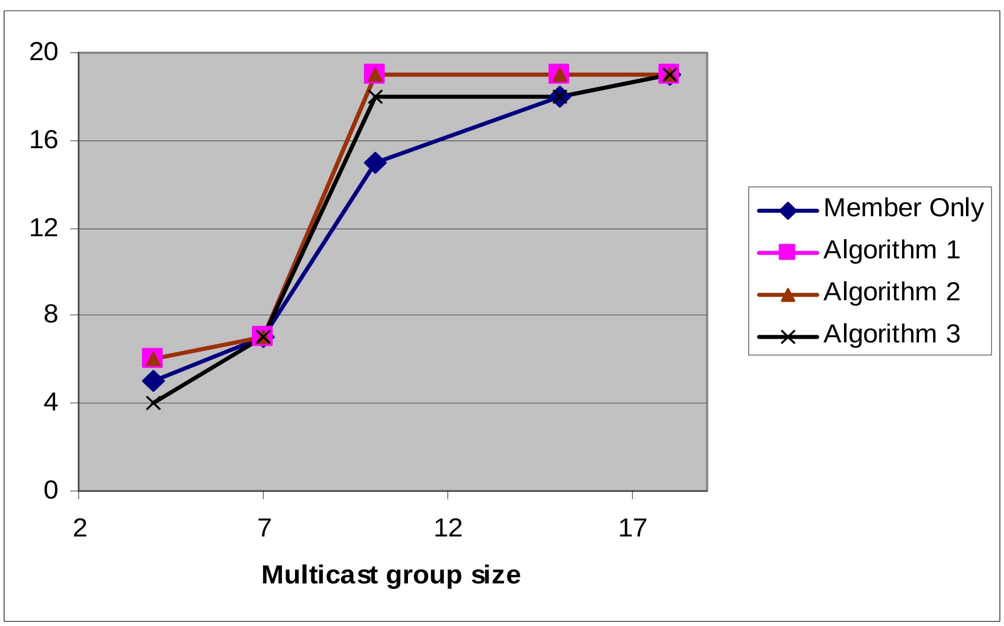


Figure 4.25 No. of links in the tree vs. Multicast group size for network 2.

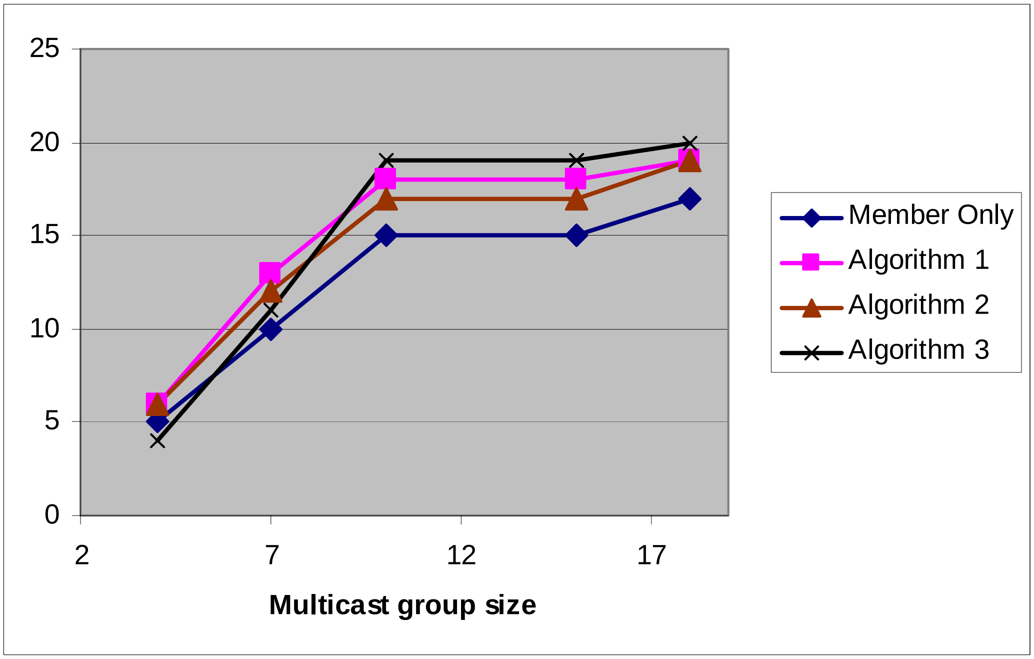


Figure 4.26 No. of links in the tree vs. Multicast group size for network 3.

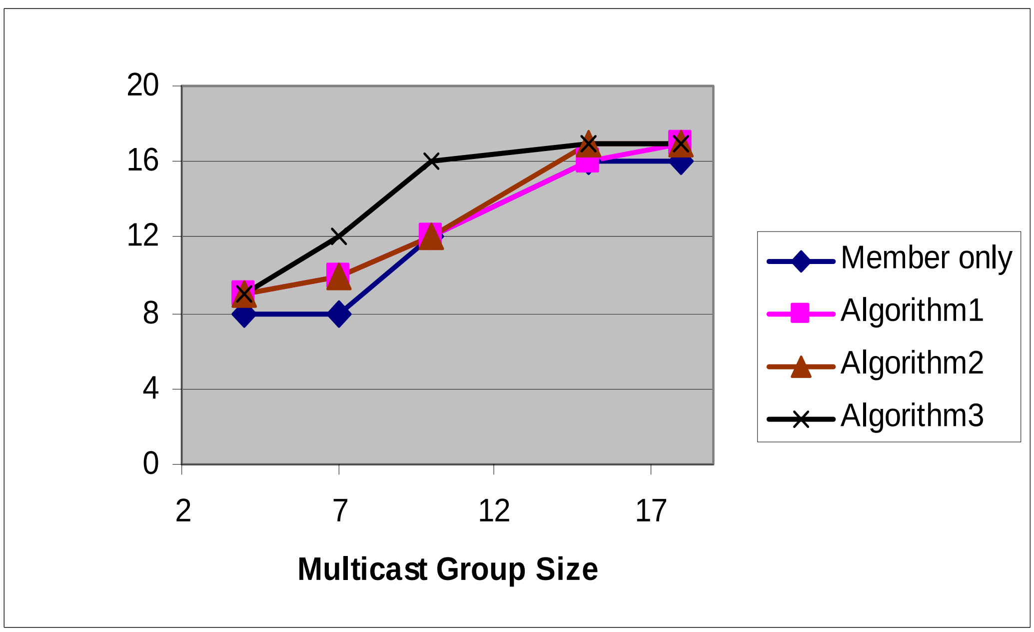


Figure 4.27 No. of links in the tree vs. Multicast group size for network 4.

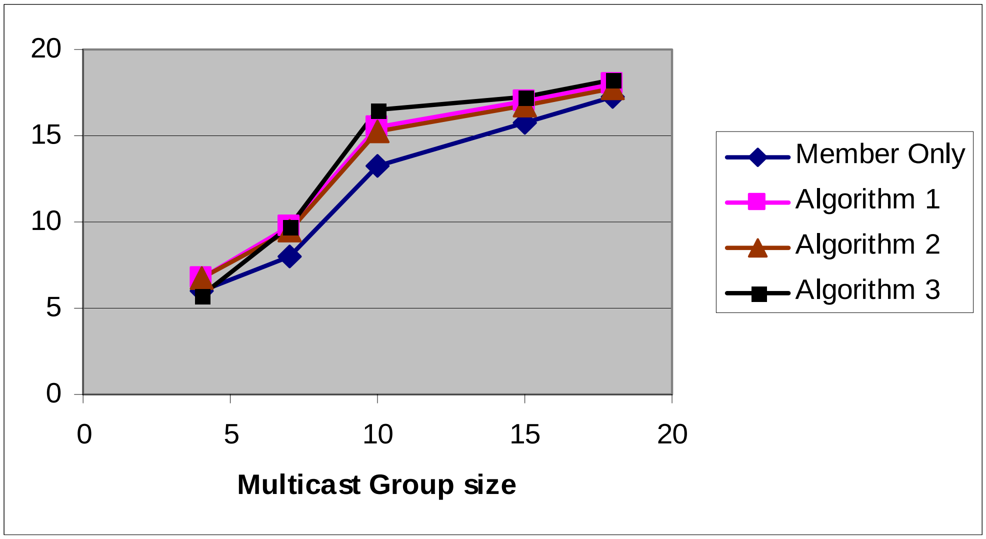


Figure 4.28 Average of the plots in Figures 4.24 through 4.27.

The plots show that Algorithm 3 uses slightly more resources when compared to the other three Algorithms. The Member Only Algorithm has the least network usage. Algorithm 3 as explained earlier does favor more links over a split if that provides better power at the destination. So it tends to use more links than the other three algorithms, which use the shortest paths for almost all destinations. Algorithm 1 and Algorithm 2 have similar network usage owing to the similarity in the algorithms. However since Algorithm 1 uses backtracking excessively it has slightly more network usage over Algorithm2.

Thus if power improvement is the major criterion then Algorithm 3 is the most suitable algorithm. But if Network usage is also a constraint then Algorithm 1 or 2 should be used. Overall the three algorithms designed in this work produce much stronger signals at the destinations compared to Member Only Algorithm.

Chapter 5

Conclusion and Future scope

This thesis dealt with the important issue of power efficient multicast in WDM optical networks. We reported three new and efficient algorithms for constructing multicast trees (or forests as the case may be) that attempted to maximize power delivered to the nodes. More precisely the algorithms try to make certain that every destination in the multicast set received power greater than a certain minimum threshold. That threshold is chosen such that the received signal can be detected reliably.

The algorithms we reported here are the first to consider both major types of power losses; attenuation and split losses, while constructing the trees. We have used both backtracking and tree pruning to achieve the power efficiency of the algorithms. The first two algorithms are enhanced versions of the shortest path heuristics for building multicast trees. The third algorithm, however, used a novel concept and considered power delivery at every step of tree construction. . In this algorithm, the order of inclusion of destination nodes into the tree was based on the power distribution in the tree and not distance. All three algorithms pruned the trees if the power levels at the destinations were not acceptable.

The performance of these three algorithms under several constraints was studied on several irregular topologies. All three algorithms reported in this work produced significant improvements and signal strengths at the set of destinations over existing multicast algorithms. Numerical results showed that our third algorithm outperformed the first two algorithms as well as the existing multicast algorithms. While we considered only power losses in this work, future algorithms can be easily modified to accommodate physical network impairments. Our algorithms in conjunction with the placement of optical amplifiers and split nodes can produce some *strong* multicast light trees.

Bibliography

[1] Zhang. X, Wei. J and Quiao. C, "Constrained Multicast Routing in WDM Networks with Sparse Light Splitting."

[2] Xin.Y and Rouskas.G, "Light Tree Routing Under Optical Power Budget Constraints," Proceedings of IEEE INFOCOM 2003.

[3] Ali. M and Deogun. J, "Allocation of Splitting Nodes in Wavelength Routed Networks," 2(3): 245-263, August 2000.

[4] Ali. M and Deogun. J, "Power Efficient Design of Multicast Wavelength Routed Networks," IEEE Journal on Selected Areas in Communications, 18(10): 1852-1862, 2000.

[5] Hu. W. S and Zeng.Q. J, "Multicasting Optical Cross Connects Employing Splitter-And-Delivery Switch," IEEE Photonics Technology Letters, 10:970-972, July 1998.

[6] Wu. K. D, Wei. J.Y and Qiao. C, "Multicast Routing with Power Consideration in Sparse Splitting WDM networks," Proceedings of IEEE ICC, Pg 513-517, 2001.

[7] Ali. M and Deogun. J. S, "Cost Effective Implementation of Multicast Wavelength Routed Networks," 0733-8724 2000 IEEE.

[8] Malli. R, Zhang. R, Qiao. C, "Benefit of Multicasting in All-Optical Networks," in SPIE Proc. Conf. All Optical Networking, Vol 38, Pg. 68-73, Feb 2000.

[9] Takahashi. H and Matsuyama. A, "An Approximate Solution for the Steiner Problem in Graphs," Math. Japanica, Vol. 24, No. 6, Pg.573-577, 1980.

[10] Wu. K. D, Wu. J. C, and Yang. C. S, "Multicast Routing with Power Consideration in Sparse Splitting WDM Networks."

[11] Murthy. C and Guru Swamy. M, "WDM Optical Networks".

[12] Mukherjee. B and Sahasrabuddhe. B. H, "Light Trees: Optical Multicasting for Improved Performance in Wavelength Routed Networks," IEEE Communication. Mag. Vol 37, pp 67-73, Feb 1999.

[13] Zhang. X, Wei. J and Qiao. C, "On Fundemental Issues in IP Over WDM Multicast," in Int. Conf. Computer Communications and Networks, Pg 84-90, Oct 1999.

[14] Bharath Kumar. K and Jaffe. J. M, "Routing to Multiple Destinations in Computer Networks," IEEE Transactions Commun., Vol.COM-31, no. 3, Pg. 343-351, Mar. 1983.

[15] Ali. M, "Optimizing the Splitting Node Placement in Wavelength Routed Optical Networks," IEEE Journal on Selected areas in Communications.

[16] Giles. C. R and Desurvire. E, "Modelling Erbuim Doped Fiber Amplifiers," IEEE/OSA J. Light Tech., Vol. 9, No.2, Pg.271-283, Feb. 1991.

[17] Ramamurthy. B, Iness. J and Mukherjee. B, "Optimizing Amplifier Placements in a Multi-Wavelength Optical Network," Dept. of Comp.Science., UC Davis, T.R. No. CSE-96-13, Sept 1996.

[18] http://www.arcelect.com/fibercable.htm.

[19] http://www.cis.ohio-state.edu/~durresi/cis777-03/sonet.pdf

[20] Rouskas. G. N and Perros. H. G, " A Tutorial on Optical Networks," Networking 2002 Tutorials, LNCS 2497, Pg. 155–193, 2002.

[21] http://www.iec.org/online/tutorials/sonet

[22] http://www.cisco.com/univercd/cc/td/doc/product/mels/cm1500/dwdm

[23] http://www.cis.ohio-state.edu/~jain/cis788-99/ftp/dwdm/index.html

[24] http://www2.rad.com/networks/1999/wdm/wdm.htm

[25] http://www.telebyteusa.com/foprimer/foch2.htm

[26] Lin. H. C and Wang. C. H, "A Hybrid Multicast Scheduling Algorithm for Single Hop WDM Networks."

[27] Oritz. Z, Rouskas. G. N, and Perros. H. G, "Scheduling of Multicast traffic in tunable receiver WDM networks with non negligible tuning latencies," In Proceedings of SIGCOMM `97, Pg 301-310. ACM, September 1997.

[28] Oritz. Z, Rouskas. G. N and Perros. H. G, "Maximizing Multicast throughput in WDM networks with tuning latencies using the virtual receiver concept," European Transactions on Telecommunications, 11(1): 63-72, January/February 2000.

[29] Doar. M and Leslie.I. "How bad is naive multicast routing," In *Proceedings of INFOCOM '93*, Pg 82–89. IEEE, April 1993.

[30] Kou. L, Markowsky. G and Berman. L, "A fast algorithm for Steiner trees."*Acta Informatica*, 15: 141–145, 1981.

[31] Ribeiro. C. C and De Souza. M. C, " Tabu search for the Steiner problem in graphs," *Networks*, 36(2): 138–146,September 2000.

[32] Bauer. F and Varma. A, "Degree Constrained Multicasting in point-to-point Networks," in Proc. IEEE INFOCOM `95, Boston, 1995, Pg. 369-376.

[33] Mukherjee. B, "Optical Communication Networks," New York: Mc-Graw-Hill, 1997.

.

Vita

Kavitha Devi Buddharaju was born in the city of Visakhapatnam in India. She completed her high school at Timpany School in the same city. Then she went on to do her bachelor's in engineering in electrical engineering at GITAM. She graduated with first class in august 2001. Zeal for higher studies made her start her master's program in Louisiana State University. She is expected to graduate with a master's degree in the fall of 2003.