İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

REROUTING ENHANCEMENTS FOR SINGLE-LAYER TRAFFIC GROOMING

M.Sc. Thesis by

Pınar KARAGÜLLE, B.S.
(504041530)

Date of Submission : 25 December 2006
Date of Defense Examination : 30 January 2007

Supervisor (Chairman): Assist. Prof. Dr. Feza BUZLUCA

Members of the Examining Committee Assoc. Prof. Dr. Sema OKTUĞ

Assoc. Prof. Dr. İbrahim ALTUNBAŞ

JANUARY 2007

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TEK KATMANLI TRAFİK "GROOMING" İÇİN YENİDEN YÖNLENDİRMEDE İYİLEŞTİRME

YÜKSEK LİSANS TEZİ

Müh. Pınar KARAGÜLLE
(504041530)

Tezin Enstitüye Verildiği Tarih : 25 Aralık 2006
Tezin Savunulduğu Tarih : 30 Ocak 2007

Tez Danışmanı : Yrd. Doç. Dr. Feza BUZLUCA

Diğer Jüri Üyeleri Doç. Dr. Sema OKTUĞ

Doç. Dr. İbrahim ALTUNBAŞ

OCAK 2007

FOREWORD

I would like to thank to Assist. Prof. Dr. Feza BUZLUCA for his guidance and support throughout my higher education. I very much appreciate the encouragement that I have received from Mr. BUZLUCA to complete this work successfully.

I should also thank to my colleagues and supervisors at Bizitek, for their tolerance.

Finally, to my mother, my father and my beloved sister: This work is dedicated to you, for your patience, understanding and endless support.

December 2006 Pınar KARAGÜLLE

TABLE OF CONTENTS

FOREWORD ii

TABLE OF CONTENTS iii

ABBREVIATIONS v

LIST OF TABLES vi

LIST OF FIGURES vii

LIST OF SYMBOLS viii

ÖZET ix

SUMMARY x

1. INTRODUCTION 1
1.1. Problem and Solution 2

2. GENERAL CONCEPTS AND PREVIOUS WORK 4
2.1. General Concepts 4
2.1.1. Dynamic Environment 4
2.1.2. Heterogeneous Network 4
2.1.3. Traffic Grooming 5
2.1.3.1 Grooming Policies 5
2.1.3.2 Traffic Grooming Problem 6
2.2. Previous Work 7
2.2.1. Single-Layered Route-Computation Algorithm 7
2.2.2. Auxiliary Graph Method 8
2.2.3. Grooming Policies and the Auxiliary Graph 12
2.2.4. Concentrating more on Dynamic Traffic 12
2.2.5. Link Bundled Auxiliary Graph Model 14
2.2.6. Grooming Decisions 17
2.2.7. Rerouting 17

3. IMPLEMENTATION OF CIGA 20
3.1 Initialization 20
3.1.1 Auxiliary Graph Initialization 20
3.1.2 Request Set Initialization 21
3.2 CIGA 22
3.2.1 Routing Requests 23
3.2.2 Removing Connections 24
3.2.3 CGSP 24

3.3 Wavelength Selection Schemes 25
3.4 LRB Grooming Policy 27

4. IMPLEMENTING REROUTING 28
4.1 Checking Cost for Deciding on Reconfiguration 29
4.2 Enhanced Reconfiguration 31
4.3 Selecting Lightpaths for Deletion 32

5. SIMULATION RESULTS 34
5.1 Wavelength Selection Scheme Comparison 36
5.2 LRB Grooming Policy Test 36
5.3 Cost-Check for Deciding on Reconfiguration Results 37
5.4 Enhanced Reconfiguration Results 38
5.5 IRD Results 40

6. CONCLUSION AND FUTURE WORK 43
6.1 Simulation Results 43
6.2 Future Work 44

REFERENCES 45

CURRICULUM VITAE 47

ABBREVIATIONS

WDM	: Wavelength Division Multiplexing
OXC	: Optical Cross Connect
OC-n	: 51.84 * n megabit per second
VTR	: Virtual Topology Reconfiguration
TDM	: Time Division Multiplexing
RWA	: Routing and Wavelength Assignment
ILP	: Integer Linear Program
TLRC	: Two-Layered Route-Computation Algorithm
SLRC	: Single-Layered Route-Computation Algorithm
RLNI	: Range-Limited with Neighbor Node Inclusion Algorithm
IGABAG	: Integrated Traffic-Grooming Algorithm
INGPROC	: Integrated Grooming Procedure
LCF	: Least Cost First
MUF	: Maximum Utilization First
MAF	: Maximum Amount First
AGP	: Adaptive Grooming Policy
MinTHV	: Minimize the Number of Traffic Hops on the Virtual Topology
MinTHP	: Minimize the Number of Traffic Hops on the Physical Topology
MinLP	: Minimize the Number of Lightpaths
MinWL	: Minimize the Number of Wavelength-Links
LBAG	: Link Bundled Auxiliary Graph
CIGA	: Constrained Integrated Grooming Algorithm
CGSP	: Constrained Grooming Shortest Path
GWC	: Generalized Wavelength Continuity
LR	: Least Resource Path First
LVH	: Least Virtual Hop Path First
LPH	: Least Physical Hop Path First
TGRR	: Traffic Grooming Ratio Reconfiguration
TLR	: Traffic Load Reconfiguration
LLR	: Maximum Lightpath Length Reconfiguration
IR	: Integrated Reconfiguration
LRB	: Least Resource Path First with Beta
IRD	: Integrated Reconfiguration with Deviation Check
LPDel	: Lightpath Deletion for Reconfiguration
LPDelE	: Lightpath Deletion Enhanced
NSFNET	: National Science Foundation Network
EUPAN	: European Public Administration Network
CBP	: Connection Blocking Probability

LIST OF TABLES

		Page No
Table 5.1	Detailed cost-check simulation results in terms of CBP……	38
Table 5.2	Detailed LPDelE results in terms of CBP …….…..	39
Table 5.3	Blocking probabilities for enhanced reconfiguration method under disconnected and periodical time-triggered policies……	40
Tablo 5.4	Detailed IR and IRD results in terms of CBP……..	41

LIST OF FIGURES

Page No

Figure 2.1 : Initial physical and virtual topologies of a simple network……... 9
Figure 2.2 : Corresponding auxiliary graph for topologies in Figure 2.1…….. 9
Figure 2.3 : Auxiliary graph state after lightpath establishment……... 11
Figure 2.4 : LBAG and layered auxiliary graph……... 14
Figure 3.1 : Sample network configuration file…… 20
Figure 3.2 : Auxiliary graph initialization…… 21
Figure 3.3 : Sample request file…… 22
Figure 3.4 : Run() method of class CIGA…….. 22
Figure 3.5 : RouteRequest method of class CIGA…… 23
Figure 4.1 : A sample case for grooming policy violation…… 29
Figure 4.2 : Method Reroute…… 29
Figure 4.3 : Method DeleteLightPath…… 30
Figure 4.4 : Method RouteDeletedConnectionsUsingExistingLPs…… 31
Figure 4.5 : A sample case for selecting lightpath with IR and IRD…… 33
Figure 4.6 : Method ComputeIdealCost…… 33
Figure 5.1 : Five-node mesh network…… 34
Figure 5.2 : 14-node NSF network…… 34
Figure 5.3 : 22-node EUPAN network…… 35
Figure 5.4 : Wavelength Selection Scheme comparison in the case of critical transceiver level…… 36
Figure 5.5 : Grooming Policy comparison in the case of critical transceiver level…….. 37
Figure 5.6 : Cost-check simulation results for five-node mesh network…..….. 38
Figure 5.7 : LPDelE results for five-node mesh network….…….. 39
Figure 5.8 : Rerouted connection counts for two triggering policies……. 40
Figure 5.9 : Simulation results for five-node mesh network comparing IR and IRD…….. 41
Figure 5.10 : Simulation results for EUPAN network comparing IR and IRD… 42
Figure 5.11 : Simulation results for NSFNET…….. 42

LIST OF SYMBOLS

G'	**:** Reachability graph in SLRC
λ	**:** Wavelength
φ(s, d, c)	**:** Connection request with source node s, destination node d and capacity c
φ	**:** Connection request
w(e)	**:** Weight of edge e
E_w	**:** Set of wavelength edges
E_r	**:** Set of receiver edges
E_l	**:** Set of lightpath edges
E_t	**:** Set of transmitter edges
hc(e)	**:** Number of physical hops within lightpath edge e
α	**:** Ratio of transceiver and wavelength weight
Q	**:** Queue of shortest paths discovered, in Dijkstra shortest path algorithm
n_1, n_2…	**:** Nodes on auxiliary graph
p_1, p_2...	**:** Paths on auxiliary graph
k	**:** Maximum number of shortest paths to find, for intermediate nodes in CGSP
β	**:** Receiver weight in LRB
hc(e)	**:** Number of physical hops within lightpath edge e
l_1, l_2...	**:** Lightpaths
Ψ	**:** Set of all connection requests
Ψ_b	**:** Set of blocked connection requests

TEK KATMANLI TRAFİK "GROOMING" İÇİN YENİDEN YÖNLENDİRMEDE İYİLEŞTİRME ÇALIŞMALARI

ÖZET

İnternetin hızla büyümesi, veri iletim kapasitesi ihtiyacını arttırmaktadır. Bu ihtiyacı karşılamak için, özellikle ana hatlarda, optik taşıyıcılar metal kabloların yerini almaktadır. Bir optik taşıyıcının yüksek bant genişliğinden, dalga boyu bölümlemeli çoğullama tekniği (WDM) ile yararlanılmaktadır.

Bir WDM optik ağda, optik taşıyıcı bağlantıları fiziksel topolojiyi meydana getirir. Optik çapraz bağlarla (OXC) birbirine bağlanan dalga boylarından oluşan yollara ışık yolu denir. Bir optik ağda kurulmuş tüm ışık yolları, sanal topolojiyi oluşturur. Optik ağ üzerinde yönlendirme yapmak, ışık yollarının fiziksel topoloji üzerinden yönlendirilmesi ve bağlantı isteklerinin ışık yolları üzerinden yönlendirilmesi alt problemlerini içerir, dolayısıyla iki-katmanlı yönlendirme problemidir. Yönlendirme problemi iki katmanda ayrı ayrı çözülebileceği gibi (iki katmanlı çözüm), bütünleşik olarak da ele alınabilir (tek katmanlı çözüm).

Bir dalga boyu kanalının kapasitesi, bir bağlantı isteğinin bant genişliği ihtiyacına göre çok yüksektir. Ağ kaynaklarını verimli kullanabilmek için, yüksek hızlı ışık yollarının bant genişliğini, düşük hızlı bağlantı isteklerine paylaştırmak gerekmektedir. Bu yönteme trafik "grooming"[1] denir.

Bu çalışmada, dinamik bağlantı istekleri için tasarlanmış, "grooming" yeteneğini de hesaba katan mevcut tek katmanlı yönlendirme çözümlerinden biri seçilerek kullanılmıştır.

Bağlantıları tekrar yönlendirerek ağ kaynaklarının verimli kullanılması konusu, dinamik trafik için yaygın olarak işlenmiş bir konu değildir. Ayrıca, dinamik trafik için önerilen yeniden yönlendirme algoritmaları yeni yolların sanal katmanda aranmasına yoğunlaşmaktadır. Bu da yönlendirmenin tek katmanda ele alındığı çözümlere uygun bir yaklaşım değildir. Bu çalışmada, dinamik bağlantı istekleri için "grooming" sorununun tek katmanlı çözümüne uygun yeniden yönlendirme yöntemleri önerilmiştir. Simülasyon sonuçları, önerilen yöntemlerin kaynak kullanımını iyileştirdiğini ve bağlantıların engellenme olasılığını düşürdüğünü göstermiştir.

---

[1] Anlamı metinde açıklanan "grooming" sözcüğünün Türkçe karşılığı bulunamamıştır.

REROUTING ENHANCEMENTS FOR SINGLE-LAYER TRAFFIC GROOMING

SUMMARY

Fast growth of internet traffic increases the demand for data transmission capacity. Optical fibers are replacing the metal wires, especially at the backbone networks, to meet the increasing demand. An optical fiber's huge bandwidth capacity is exploited by Wavelength Division Multiplexing (WDM) technique.

In an optical WDM network, optical fiber links form the physical topology. Paths of wavelengths connected to each other by optical cross connects (OXCs) are called lightpaths. All the lightpaths established in an optical network constitute the virtual topology. Routing connections in an optical network involves sub problems of routing the lightpaths over physical topology and routing the connections over the lightpaths; thus is a two-layer routing problem. Routing problem can be solved in two layers separately (two-layer solutions) or jointly (single-layer solutions).

There exists a large a gap between the capacity of a WDM channel and the bandwidth requirement of a connection request. In order to use the network resources efficiently, low-speed traffic connections need to be multiplexed onto high-speed lightpaths. This method is referred to as traffic grooming.

In this study, an existing single-layer routing solution for dynamic traffic requests, which considers traffic grooming, was applied.

Rerouting connections to utilize network resources efficiently has not been widely considered for dynamic traffic conditions. Furthermore, rerouting algorithms proposed for dynamic traffic perform new path search on virtual layer. This approach is not suitable for single-layer routing solutions. In this study, rerouting enhancements for single-layer solution of traffic grooming under dynamic traffic conditions were proposed. Simulations proved that proposed methods improve resource utilization and decrease blocking probability.

1. INTRODUCTION

Fast growth of internet traffic increases data transmission capacity demand dramatically. Optical fibers are replacing the metal wires, especially at the backbone networks, to meet the increasing bandwidth demand.

The huge bandwidth of an optical fiber is exploited by Wavelength Division Multiplexing (WDM) technology [1]. Multiple wavelengths are carried on a single fiber link, with the use of WDM technology.

Optical fiber strands are laid in bundles (864 strands/ bundle) and bundles are laid in conduits (10 bundles/ conduit). [1] Multiplying the total capacity of multiplexed wavelengths per fiber with strand count in a conduit, the huge capacity of bandwidth an optical link provides can be calculated.

In an optical WDM network, optical fiber links form the physical topology. A wavelength path which spans several physical links and uses one wavelength on each link along its path is referred to as a lightpath. Wavelengths along a lightpath are connected to each other by Optical Cross Connects (OXCs). At the source node of a lightpath, a transmitter generates an optical signal from an electronic signal. At the destination node, a receiver converts the optical signal into an electronic signal.

All the lightpaths established in an optical network form the virtual topology. Connection requests are routed over the lightpaths, i.e. virtual topology. In other words, routing connections in an optical network involves sub problems of building the lightpaths and routing the connections over the lightpaths; thus is a two-layer (physical, virtual) routing problem.

There is a large gap between the capacity of a WDM channel (OC-48, OC-192, OC-768) and the bandwidth requirement of a typical connection request. If whole bandwidth of a wavelength channel is allocated for one single connection, a huge bandwidth capacity will remain unused, will be wasted. Multiplexing low-speed

traffic streams onto high-speed lightpaths, in order to use the network resources efficiently, is referred to as traffic grooming. [2] Routing solutions for WDM networks should consider traffic grooming ability, too.

Routing methods developed for WDM networks can be classified in two groups, depending on the type of traffic conditions they are built for. The first group of solutions is designed for the case when all traffic requests are known in advance (static traffic). The second group of solutions is designed for the case where connections arrive one at a time (dynamic traffic).

1.1 Problem and Solution

Various methods for solving routing problem in a groomable network, under dynamic traffic conditions were examined. An auxiliary graph method was selected and implemented. Since the auxiliary graph method handles routing problem in two layers jointly (single-layer solution), it performs better in terms utilizing network resources efficiently.

Under static traffic conditions, i.e. when the set of traffic requests are known in advance, it is possible to put the requests in the optimal order so that the resource utilization is minimized, when the connections are routed. Under dynamic traffic conditions, on the other hand, connections are routed as soon as they arrive; there is no chance of sorting requests for better resource utilization. A connection may be routed through a long route, because it is the shortest route available at the time of request's arrival. However, by attempting to reroute this type of connections, more efficient routes might be discovered, since the network state is continuously changing due to terminating and arriving connections.

Rerouting connections in dynamic traffic environment has not been widely considered. Virtual Topology Reconfiguration (VTR) concentrates on changing virtual topology for adapting changes in traffic patterns. Under dynamic traffic conditions, traffic patterns are not known in advance. For dynamic conditions, rerouting studies concentrate on "connection rerouting" which refers to rerouting connections through different existing lightpaths. This approach is not suitable when

routing is performed on a single-layer auxiliary graph, since it neglects better physical routes.

Starting with implementation of rerouting method used in [3]; in this study, various rerouting enhancements for dynamic traffic are proposed, on a single-layer traffic grooming architecture. The enhancements can be summarized as,

1. Selecting a lightpath for deletion and rerouting effected connections through existing lightpaths, only if more source-efficient new routes can be found.
2. Selecting a lightpath for deletion and searching for more source-efficient new routes on the whole auxiliary graph,
3. Checking cost deviation instead of cost, in the search of lightpath to delete.

2. GENERAL CONCEPTS AND PREVIOUS WORK

2.1 General Concepts

2.1.1 Dynamic Environment

In one type of routing environment, all connection requests are given in advance. This type of environment is called static environment. On the other hand, in real world the connections are usually not known in advance. They arrive one at a time and depart, dynamically. [4]

For dynamic environment conditions, the objective of routing should be minimizing the network resources used for each request, thus indirectly minimizing the blocking probability for future requests.

2.1.2 Heterogeneous Network

Some nodes can perform wavelength conversion, i.e., can convert the incoming wavelength into another wavelength, on demand. Conversion capabilities of nodes in a network may vary. Some may have full wavelength conversion capability (can convert all wavelengths into all others), while some can only convert from one subset of wavelengths to another. Nodes with no conversion capability require that the incoming and outgoing physical links in a lightpath passing through are on the same wavelength.

A network in which the nodes have different wavelength conversion capabilities is said to be heterogeneous, in terms of wavelength conversion capability.

Other heterogeneity components are grooming capabilities of nodes, transceivers counts of nodes and the number of wavelength edges on physical links.

Since real optical networks are heterogeneous, a realistic routing procedure should be applicable for heterogeneous networks.

2.1.3 Traffic Grooming

The capacity of a wavelength channel is very high (OC-48, OC-192, OC-768) in comparison with the bandwidth requirement of a single request (OC-1, OC-4, etc). If the entire bandwidth of a lightpath is allocated for one single connection, a large portion of the bandwidth will be wasted. This fact leads to the idea of packing multiple connections into a single lightpath, i.e. traffic grooming.

A node must employ access stations which can multiplex and demultiplex low-speed connections using techniques like time division multiplexing (TDM), in order to groom connections. Grooming capabilities of nodes in a network may vary. Some may provide full-grooming while some may have a limited number of transmitters and receivers for groomable wavelength channels.

2.1.3.1 Grooming Policies

Route selection in a grooming-enabled network is influenced by the answers of following questions [5]

1. Should the connection be routed over the existing lightpaths, if possible? In some cases, building a new lightpath might be more sensible, even though existing lightpaths can be used.

2. Between which nodes should the new lightpaths be added? A direct lightpath can be set up from source to destination, or existing lightpaths can be used along with new lightpaths set up between/to/from intermediate nodes.

The decisions on these two issues are referred to as grooming policies.

Grooming policies vary according to the network configuration and network state. Depending on the critical resource, the intension might be minimizing the number of wavelength-links, number of lightpaths, traffic hops on the virtual topology, etc.

2.1.3.2 Traffic Grooming Problem

Traffic grooming problem can simply be defined as routing problem in a network where all or some nodes have grooming capability.

The inputs of traffic grooming problem are:

1. Physical topology (Physical links, wavelengths available on each fiber and their capacities, number of transceivers at each node)
2. A set of connection requests with different bandwidth granularities

The objective is to find out how to set up lightpaths to satisfy connection requests. Multiple connections can be multiplexed on one lightpath. Since the connections are routed over the lightpaths and the lightpaths are routed over the physical links, grooming problem in an optical network is a two-layer routing problem.

Grooming problem is divided into four subproblems in [6]:

1. Determining the virtual topology (lightpaths)
2. Routing the lightpaths over the physical topology;
3. Assigning wavelengths to physical edges of the lightpaths
4. Routing connections on the virtual topology.

Since the subproblems of selecting virtual topology, routing and wavelength assignment (RWA) are NP-Hard, grooming problem is also NP-Hard.

These four subproblems can be solved separately or jointly. Disadvantage of separate solutions is that independent optimal solution of each step may not result in a near-optimal solution of the whole problem. Furthermore, this type of solution is not applicable to dynamic environment, since all the connection requests are not known in advance.

One joint solution method is building an Integer Linear Program (ILP). This solution is not applicable because of its high time complexity. It is not suitable for dynamic traffic, either, since it also requires that all the connections are known in advance.

For a more tractable joint solution, heuristic methods have been developed, which can apply to both static and dynamic traffic.

2.2 Previous Work

2.2.1 Single-Layered Route-Computation Algorithm

In [7], two joint grooming algorithms are proposed for dynamic traffic.

First solution, which is called Two-Layered Route-Computation Algorithm (TLRC), selects the route as follows, for each connection request with source s, destination d, and capacity c:

1. If there is a lightpath with enough available capacity, return path.
2. Route the request through existing lightpaths. If successful, return path found.
3. Try to establish a new lightpath between s and d. If successful, return the path.

Here, steps 1 and 2 run on the virtual layer and the third step runs on the physical layer. However, there might be cases that a request can be routed through existing and new lightpaths. It is not possible to discover this type of paths using TLRC, since it explores the two layers separately.

Second solution, which is called Single-Layered Route-Computation Algorithm (SLRC), selects the route as follows:

1. With the same nodes in the physical topology, generate a new graph G'. In G', add a link between each node pair (i,j), if there is a groomable lightpath between them or a new groomable lightpath can be set up between them.
2. Find the shortest path on graph G' from s to d. If successful, return the path.

The simulations showed that using SLRC decreases the blocking probability, since it can discover the paths consisting of both existing and new lightpaths, and uses the wavelength resources more efficiently, since it computes the shortest path. On the other hand, TLRC performs better when the critical resource is the transceivers, since it avoids setting up new lightpaths as much as possible.

So, if it is planned to use the methods proposed here for grooming, then one of the two computation schemes must be selected for different network configurations and for different network states, which is not very practical.

These two algorithms find a path if it exists, no matter how long it is. This might result in bad resource utilization especially for large networks; since some connections might be routed through very long paths. It might be better to block such resource-consuming requests, in favor of future requests. In [8], a new method called Range-Limited with Neighbor Node Inclusion Algorithm (RLNI) is proposed, to solve this problem. This algorithm modifies SLRC as follows:

1. Instead of all nodes, use only candidate nodes to draw the graph G'. The candidate nodes are the nodes on the physical shortest path from s to d.
2. Run shortest path alorithm to find a route. If successful, return path.
3. Add more candidate nodes to G', i.e. add neighbor nodes of the original candidate nodes.
4. Run shortest path algorithm again. If successful, return path. This procedure can be repeated for several times.

Here, the search starts locally and expands at every iteration. The length (and so the cost) of the routes discovered increases with iteration count. Optimal iteration count should be carefully selected so that short routes are not blocked and too-long routes are blocked.

This local approach not only prevents bad resource utilization, it also shortens the running time of the algorithm, by reducing the search space.

2.2.2 Auxiliary Graph Method

In [5], a heuristic joint method which employs a generic auxiliary graph for traffic grooming in a heterogeneous WDM mesh network is proposed. Instead of dealing with the routing problem separately in two layers (physical, virtual), the routing algorithm runs on the auxiliary graph which combines the two layers. Though the work concentrates on static traffic case, the method proposed can be applied for dynamic traffic, with little modification in procedure INGPROC, which will soon be explained.

Initial physical and virtual topologies of a simple network are shown in Figure 2.1. Each link on physical topology has two different wavelengths (λ_1, λ_2). There are no links on virtual topology, since no lightpaths have been established yet. Corresponding auxiliary graph that integrates these two graphs is constructed as in Figure 2.2.

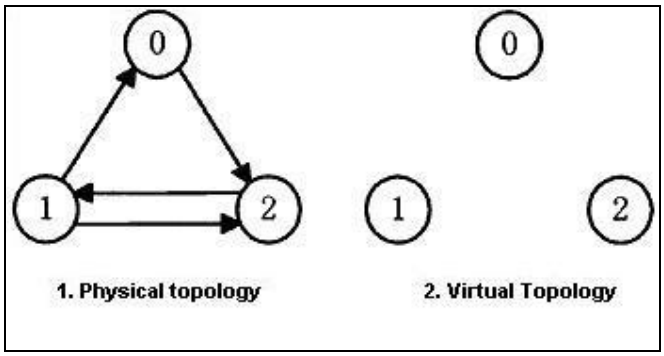

Figure 2.1: Initial physical and virtual topologies of a simple network

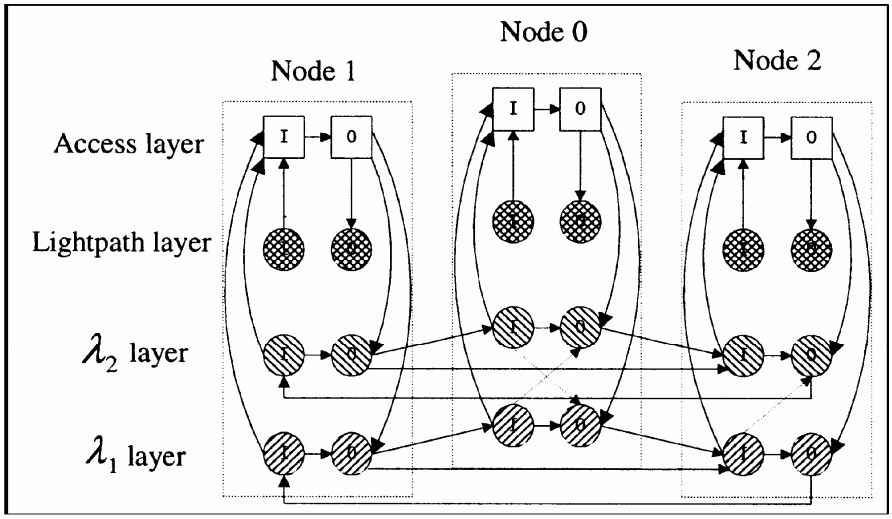

Figure 2.2: Corresponding auxiliary graph for topologies in Figure 2.1

For each node on the physical topology, there are two nodes in each layer of the auxiliary graph: input and output ports.

Edges from input ports to output ports on each wavelength layer refer to OXCs and are called Wavelength Bypass Edges.

If node i has grooming capability, then there is an edge from input port to output port on the Access layer. These edges are called Grooming Edges.

At each node, there is a multiplexer edge from the output port on the access layer to the output port on the lightpath layer, and a demultiplexer edge from the input port on the lightpath layer to the input port on the access layer.

If there are transmitters available at node i on wavelength λ1, then there is a transmitter edge from output port on the Access layer to the output port on wavelength layer λ_1. If there are receivers available at node i on wavelength λ1, then there is a receiver edge from output port on wavelength layer λ1.to the Access layer.

Edges between wavelength layers (λ_1, λ_2) are Converter Edges.

If there is an unused wavelength of λ_1 on physical link i to j, then there is a wavelength-link edge from the output port on wavelength layer λ1 at node to the input port on wavelength layer λ_1 at node j.

There is a lightpath edge from the output port on the lightpath layer at node i to the input port on the lightpath layer at node j, if there is a lightpath from node i to j.

Based on the explained auxiliary graph structure, an integrated traffic-grooming algorithm (IGABAG) is proposed, which jointly solves the four traffic-grooming sub problems. The auxiliary graph reflects the state of the network, during the operation.

The grooming algorithm first constructs the auxiliary graph according to the network configuration. Then it simply finds the shortest path on the auxiliary graph for each request and updates the graph accordingly.

A traffic request φ(s, d, c) has a source, destination and capacity of the request. For each request, before running the shortest path algorithm, wavelength edges with capacity lower than the capacity of the request and lightpaths whose available (unused) capacity are lower than the capacity of the request are temporarily deleted. Then the shortest path from node s access layer output port to node d access layer input port is found. If no path is found, the request is blocked.

If the path found enters a wavelength layer, this implies that a new lightpath will be set up. The lightpath starts with a transmitter edge and is terminated by a receiver edge. While setting up a new lightpath on the auxiliary graph, transceiver and wavelength edges used are modified: count property of each edge is decreased by one, and the edge is deleted if its count reaches 0. Then the new lightpath is added to the lightpath layer.

To show how a new lightpath is established on the auxiliary graph, a simple case will be examined. Let each node have only one transmitter and one receiver for each wavelength and let each physical link have one λ_1 and one λ_2 wavelength. A lightpath will be set up from node 0 to node 1.

Starting from the output port of node 0 access layer, follow the path:

→ (via transmitter edge) node 0 λ_2 layer output port
→ (via wavelength link edge) node 2 λ_2 layer input port
→ (via wavelength bypass edge) node 2 λ_2 layer output port
→ (via wavelength link edge) node 1 λ_2 layer input port
→ (via receiver edge) node 1 access layer input port.

Figure 2.3 shows the auxiliary graph state after this lightpath is established and the auxiliary graph is updated as explained in the previous paragraph.

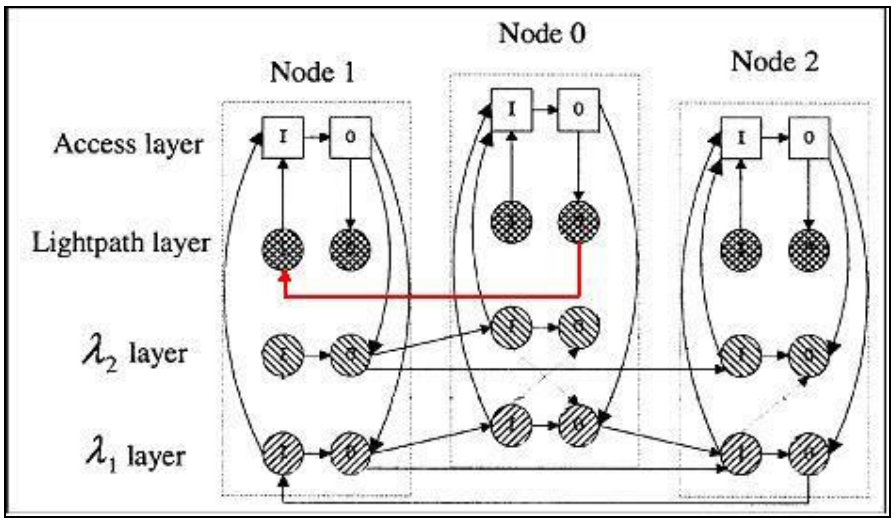

Figure 2.3: Auxiliary graph state after this lightpath establishment

Finally the connection is routed along the path found, which consists of old and new lightpaths. At this step, lightpath capacities are decreased by the capacity of the request.

The procedure calling IGABAG for each request in the connection request set is named INGPROC. INGPROC is developed for static traffic case and selects a connection request from the set, according to a traffic selection scheme. Traffic selection schemes proposed are Least Cost First (LCF), Maximum Utilization First (MUF), and Maximum Amount First (MAF).

LCF attempts to choose the most cost-effective traffic request under the current network state. The cost of a traffic request is computed by dividing the cost of the shortest path for routing the request by its capacity. MUF selects the request with highest utilization, which is computed by dividing the capacity of the request by the number of physical hops from source to destination. Finally, MAF simply selects the request with highest capacity.

2.2.3 Grooming Policies and the Auxiliary Graph

Auxiliary graph method is advantageous in that it simplifies applying different grooming policies.

Instead of designing a route-computation algorithm for each grooming policy like in [7], the same route computation method, the shortest path algorithm, is used in the auxiliary graph method. In the auxiliary graph method, a grooming policy is applied by assigning edges the proper weight functions. For example, if the intension of the Grooming Policy is to minimize the number of wavelength-links consumed, then the weight of the wavelength links must be assigned efficiently higher than the other edges. Details of assigning weights will be explained, later.

2.2.4 Concentrating more on Dynamic Traffic

The work in [9] concentrates more on dynamic traffic conditions.

The above auxiliary graph structure is used and the same connection routing algorithm finds the shortest path for each request on the auxiliary graph and updates

the graph to reflect the network state. The difference here is that connection requests are not selected from a given set of requests; they arrive dynamically and are routed as soon as they arrive.

Since the requests arrive and depart dynamically, the paper explains how the auxiliary graph is updated when a connection terminates. The operation is the reverse of the arriving connection request operation.

Step 1: Remove the traffic, i.e. free the lightpath capacities consumed for the request

Step 2: Delete all the lightpaths that do not carry any traffic.

Step 3: Add resources back to the auxiliary graph, which are freed by tearing down lightpaths, i.e. increase transmitter, receiver, wavelength counts etc.

Available network resources dynamically increase and decrease as the connections arrive and depart. Critical resources in the network might change during the operation. For example, setting up long lightpaths for successive requests may result in temporary shortage of wavelengths. Likewise, setting up too many short lightpaths for successive requests may result in temporary shortage of transceivers. A good grooming policy should adapt to the changing conditions, in the case of dynamic traffic.

In [9], an adaptive grooming policy (AGP) which dynamically changes edge weights is proposed.

Constant grooming policies were compared first: MinTHV (Minimize the Number of Traffic Hops on the Virtual Topology), MinTHP (Minimize the Number of Traffic Hops on the Physical Topology), MinLP (Minimize the Number of Lightpaths) and MinWL (Minimize the Number of Wavelength-Links). Since it was discovered that MinTHV performs the best when transceivers are the critical resources and MinTHP performs best when wavelength-links are critical, the advantages of these two policies are combined in the proposed AGP. AGP switches between MinTHV and MinTHP according to the current network state.

It is proved in the work that AGP performs better than MinTHV and MinTHP, which turn out to be useless in the case that the resource they try to minimize is not the critical one.

2.2.5 Link Bundled Auxiliary Graph Model

The disadvantage of Auxiliary Graph model proposed in [5] is that it gets harder to manage the graph, as the number of different wavelengths on a link increases. Since adding a new wavelength implies adding a layer to the graph, i.e adding (physical node count * 2) nodes, the search space grows dramatically as the number of wavelengths increase.

In [10], a "link bundling" method is proposed, for building a more tractable graph. Instead of showing each wavelength as a separate layer, there is one physical layer. On the physical layer, there is an edge between nodes (i, j), if there is an available wavelength between them. In other words, available wavelength set is an attribute of the physical layer link.

A simple physical topology, corresponding Link Bundled Auxiliary Graph (LBAG) and layered auxiliary graph are shown in Figure 2.4.

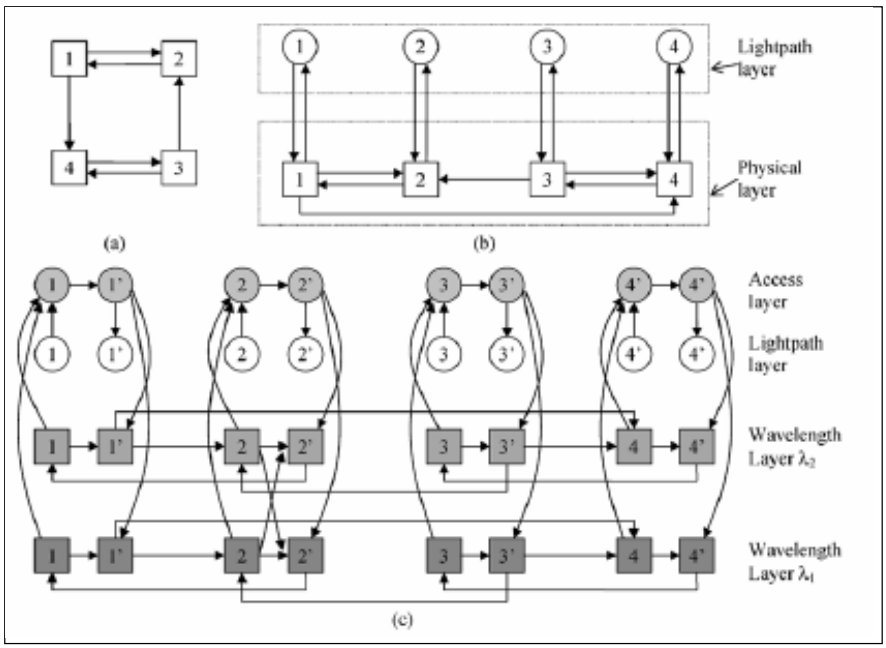

Figure 2.4: LBAG and layered auxiliary graph

For each node in the physical topology, there are two nodes in the auxiliary graph: one in physical layer and one in lightpath layer.

Transceivers are assumed to be tunable for every wavelength, i.e., a transceiver can be used to transmit/receive any wavelength. So, if there is an available transmitter at node i, then there is a transmitter edge from lightpath layer node i to physical layer node i. If there is an available receiver at node i, then there is a receiver edge from physical layer node i to lightpath layer node i. There is a lightpath edge from lightpath layer node i to lightpath layer node j, if a lightpath (i, j) is set up. Finally, there is a wavelength edge from physical layer node i to physical layer node j, if there is a physical link (i, j) and the available wavelength set is not empty for that link.

Finding the route on layered graph was simply accomplished by finding the shortest path, no constraints had to be checked, because all the constraints were represented by edges. Wavelength conversion capabilities were represented by wavelength converter edges, too. Here, along with the wavelength layers, wavelength converter edges are removed from the graph, too. Wavelength conversion capabilities are stored as node attributes. Storing available wavelength set as wavelength link property and wavelength conversion capabilities as node property implies that wavelength continuity constraint needs to be checked in the shortest path algorithm, i.e. a constrained shortest path algorithm should be applied on LBAG. Constrained Integrated Grooming Algorithm (CIGA) is proposed.

CIGA simply runs Constrained Grooming Shortest Path algorithm (CGSP) for each arriving request, to find the shortest available path from source to destination. If a path is found, the auxiliary graph is updated accordingly: Lightpath edges added, wavelength and transceiver edge properties updated, lightpath capacities decreased. If no path is found, the request is blocked. Likewise, after each disconnection request, the resources are freed and the graph is updated. CIGA will be examined in detail, later on.

CGSP is a modified version of k-shortest path Dijkstra. It checks Generalized Wavelength Continuity (GWC) constraint before inserting edges into the queue storing discovered paths.

Generalized Wavelength Continuity can be defined as follows: On a lightpath l, let incoming wavelength to node i be λ_1 and the outgoing wavelength be λ_2. Node i should have the capability of converting λ_1 to λ_2, for this wavelength assignment to be valid. In CGSP, if at least one feasible wavelength assignment is possible along the path from source s to intermediate node n, then the path is added to the queue of discovered paths.

In [10], a new grooming policy is proposed, too. The policy called Least Resource Path First (LR) attempts to combine policies Least Virtual Hop Path First (LVH) and Least Physical Hop Path First (LPH), which correspond to MinTHV and MinTHP in [5].

1. In order to apply grooming policy LVH, weight assignments for edges should be:

$w(e) = 0$, if $e \in E_w \cup E_r$ **(2.1a)**

1, if $e \in E_l \cup E_t$ **(2.1b)**

Where E_w is the set of wavelength edges, E_r is the set of receiver edges, E_l is the set of lightpath edges and E_t is the set of transmitter edges.

2. In order to apply grooming policy LPH, weight assignments for edges should be:

$w(e) = 0$, if $e \in E_t \cup E_r$ **(2.2a)**

1, if $e \in E_w$ **(2.2b)**

$hc(e)$, if $e \in E_l$ **(2.2c)**

Where hc(e) is the number of physical hops within lightpath edge e.

3. By simply multiplying LVH edge weights by a coefficient α and adding them to corresponding LPH edge weights, LR edge weights are obtained:

$w(e) = 0$, if $e \in E_r$ **(2.3a)**

α, if $e \in E_t$ **(2.3b)**

1, if $e \in E_w$ **(2.3c)**

$\alpha + hc(e)$, if $e \in E_l$ **(2.3d)**

LPH performs better in relatively sparse networks, like NSF where it is more difficult to find an alternate path and, thus, the wavelength resource is critical. LVH, performs better in denser networks, like EUPAN where plenty of alternate paths exist and the transceiver resource becomes critical. The LR grooming policy, which treats wavelength and transceiver resources equally (when α=1), has a good performance which is slightly below LPH in the NSF network and has the best performance in the EUPAN network.

Furthermore, by only changing α, the grooming policy can adapt to the network state, during operation.

2.2.6 Grooming Decisions

After the five papers [5, 7-10] explained up to now were examined, LBAG method with CIGA and LR grooming policy were selected and applied. Because:

1. Single-layer algorithms perform better in terms of blocking probability.
2. LBAG is suitable for reflecting a heterogeneous network.
3. CIGA is constructed for dynamic traffic.
4. Since CGSP is a constrained k-shortest path algorithm, high cost paths which lead to bad utilization of the resources are not found, unlike in methods of [5, 7, 9].
5. LBAG is smaller in size and thus CIGA is faster, compared to IGABAG that runs on layered auxiliary graph in [5, 9]
6. Different grooming policies can be simply applied by changing edge weights on LBAG, unlike in [7], where route computation scheme should be changed.
7. LR considers both transceiver and wavelength resources and it can simply be adaptive, by changing α.
8. CGSP search is global, i.e. explores the whole graph, but it still prevents finding too long paths. So it is preferable to local search proposed in [8].

2.2.7 Rerouting

Under dynamic traffic conditions, the connections are routed as soon as they arrive. There is no chance of delaying and sorting connections so that they are routed in the optimal order consuming minimum resource. A connection might be routed through

a long path, since it is the shortest path found at the time the request arrives. However in time, some connections are terminated and the connection might be rerouted through a shorter path, if a rerouting mechanism is employed.

Rerouting mechanism can be in the form of rerouting connections through new lightpaths, or rerouting lightpaths through new physical paths. The methods are called connection rerouting and lightpath rerouting, respectively. As it is clear from this categorization, rerouting studies concentrate on virtual and physical layer separately. There are not many works concentrating on rerouting for single-layered solutions of dynamic traffic routing. Considering grooming with rerouting has not been widely covered, either.

In [9, 11], it is asserted that under dynamic scenario, splitting or rerouting a lightpath is not acceptable.

In [11], a novel integrated grooming solution for dynamic traffic is explained and rerouting connections is considered, too. If a connection cannot be routed in the current network state, some connections are rerouted so that the arriving connection request can be satisfied.

In [3], reconfiguration of lightpaths in a groomable network is considered.

The routing mechanism in [3] searches existing lightpaths first, for routing a connection. Then it tries to establish a direct lightpath between source and destination. At the last step, a path consisting of existing and new lightpaths is searched. The routing method here is hybrid: first two-layered and then single-layered.

At each reconfiguration attempt, a lightpath is selected for deletion. If the connections the lightpath carries can be routed through existing lightpaths, then the lightpath is deleted and the connections are rerouted.

[3] concentrated more on when and how to reconfigure. Three reconfiguration triggering methodologies were examined: Disconnected, periodic time-triggered, and blocked. Comparing these three policies, it was discovered that periodic time-triggered policy has the best performance. Disconnected policy, which attempts to reconfigure at every disconnect request, has the best performance in decreasing

blocking probability, but disturbs network traffic too frequently. Blocked policy, which attempts to reconfigure whenever a connection is blocked, minimizes traffic disturbance, but has a bad performance in terms of blocking probability. The optimum solution can be applied by adjusting a period; which is possible in time-triggered policy.

In reconfiguration algorithm analysis, four algorithms were examined: traffic grooming ratio reconfiguration (TGRR), traffic load reconfiguration (TLR), maximum lightpath length reconfiguration (LLR) and the proposed integrated reconfiguration (IR). TGRR selects the lightpath to delete, by considering the number of low speed connections carried by lightpaths. TLR considers traffic load and LLR considers the physical hop count. IR considers all these factors. The tests showed that IR performs the best, when both network disruption and resource utilization are taken into account.

3. IMPLEMENTATION OF CIGA

3.1 Initialization

CIGA takes connection request set and initial auxiliary graph as input. Auxiliary graph and the connection request set initialization will be covered in this part.

3.1.1 Auxiliary Graph Initialization

Network configuration is read from network configuration file. A sample network configuration file is shown in Figure 3.1. Format of the file is a modified version of format used in [12].

```
//<TOPOLOGY>     N_NODE  N_TRANSRCVR
<TOPOLOGY>       5       100

//WAVELENGTH_CONVERSION_MATRIX          OC3     OC12    OC48    OC192
<WAVELENGTH_CONVERSION_MATRIX>  OC3     1       0       0       0
<WAVELENGTH_CONVERSION_MATRIX>  OC12    0       1       0       0
<WAVELENGTH_CONVERSION_MATRIX>  OC48    0       0       1       0
<WAVELENGTH_CONVERSION_MATRIX>  OC192   0       0       0       1

//WAVELENGTH_SET        OC3     OC12    OC48    OC192
<WAVELENGTH_SET>        0       0       0       5


//NODE  NAME
<NODE>  1
<NODE>  2
<NODE>  3
<NODE>  4
<NODE>  5

//LINK  src     dst
<LINK>  1       2
<LINK>  2       1
<LINK>  1       3
<LINK>  3       1
<LINK>  2       3
<LINK>  3       2
<LINK>  2       4
<LINK>  4       2
<LINK>  3       4
<LINK>  4       3
<LINK>  4       5
<LINK>  5       4
```

Figure 3.1: Sample network configuration file

First, the physical topology is created, using node count (N_NODE), node names and link information. Transmitter and receiver count property of each physical node is initialized by N_TRANSRCVR. Each physical node's wavelength conversion matrix property is initialized by WAVELENGTH_CONVERSION_MATRIX entries and each physical edge's available wavelength set is initialized by WAVELENGTH_SET entries.

Then, the auxiliary graph is initialized by physical topology, as shown in Figure 3.2.

```
AuxiliaryGraph(PhysicalTopology)

For every PhysicalNode in physicalTopology.PhysicalNodes
        AuxiliaryGraphNodes.Add(PhysicalNode)
End For

For every PhysicalEdge  in physicalTopology.PhysicalEdges
        AuxiliaryGraphNodes.Add(PhysicalEdge)
End For

For every PhysicalNode in physicalTopology.PhysicalNodes
        AuxiliaryGraphNodes.AddVirtualNodeForNode(PhysicalNode)
End For

For every PhysicalNode in physicalTopology.PhysicalNodes
        AuxiliaryGraphNodes.AddTransmitterEdgeFromVirtualToPhysical
                                (PhysicalNode.TransmitterCount)
        AuxiliaryGraphNodes.AddReceiverEdgeFromPhysicalToVirtual
                                (PhysicalNode.ReceiverCount)
End For
```

Figure 3.2: Auxiliary graph initialization

Note that wavelength conversion matrix properties are stored in the copied physical edges and available wavelength set properties are stored in copied physical edges.

3.1.2 Request Set Initialization

Connection requests and disconnection requests are read from request file. A sample request file is shown in Figure 3.3.

```
//CONNECTION     conid   time     src     dst     cap     duration
<CONNECTION>     1       1.06     5       1       1       630.859
<DISCONNECTION>  1       631.919
<CONNECTION>     2       1.989    3       4       1       232.506
<DISCONNECTION>  2       234.495
<CONNECTION>     3       2.944    2       4       1       278.579
<DISCONNECTION>  3       281.523
<CONNECTION>     4       3.72     3       5       16      336.415
<DISCONNECTION>  4       340.135
```

Figure 3.3: Sample request file

Capacity parameter can take on values from a set (1, 4, 16), where 1 corresponds to OC-3, 4 corresponds to OC-12 and 16 corresponds to OC-48.

In addition to initializing request set and auxiliary graph, CGSP parameter k, Grooming Policy and Lightpath Selection Scheme decisions must be carried out, too, before running CIGA.

3.2 CIGA

CIGA is initialized with parameters AuxiliaryGraph and RequestSet. After initialization, the method Run, of which pseudo code is shown in Figure 3.4, is called.

```
Run()

while RequestSet is not empty
        if nextReuqest is ConnectionRequest
                routed ← RouteRequest(nextRequest,AuxiliaryGraph)
                if (routed is false)
                        AuxiliaryGraph.BlockedRequests.Add(nextRequest)
                end if
        else
                AuxiliaryGraph.RemoveConnectionByID(nextRequest.ID)
        end if
        RequestSet.Remove(nextRequest)
end while
```

Figure 3.4: Run() method of class CIGA

Request set contains both connection and disconnection requests. The first part of the outer "if" clause processes connection requests and the second part processes disconnection requests.

For connection requests, the algorithm attempts to route the connection first. If the connection cannot be routed, it is inserted into the blocked request list.

For disconnection requests, corresponding connection is found by the id of disconnection request and is removed.

3.2.1 Routing Requests

RouteRequest function is described in Figure 3.5.

```
RouteRequest(currentRequest, AuxiliaryGraph)
AuxiliaryGraph.RemoveEdgesUnderCapacity(currentRequest.Capacity)
Path ← CGSP.FindShortestPath(currentRequest)
if (Path is found)
        For every Edge in Path
                If Edge is a Lightpath
                        ConnectionLPs.Add(Edge)
                End if
                If Edge is a Transmitter
                        TransmitterEdge ← Edge
                        AuxiliaryGraph.useTransmitter(TransmitterEdge)
                        Edge ← Get next edge from Path
                        while (Edge is not a Receiver)
                                PhysicalSubpath.Add(Edge)
                                Edge ← Get next edge from Path
                        end while
                        ReceiverEdge ← Edge
                        AuxiliaryGraph.useReceiver(ReceiverEdge)
                        For every PhysicalEdge in PhysicalSubpath
                                selectedWavelength ← Assign Wavelength
                                PhysicalEdge.UseWavelength(selectedWavelength)
                        End for
                        Lightpath ← new LightPath from TransmitterEdge.Source to
                                        ReceiverEdge.Destination
                        ConnectionLPs.Add(Lightpath)
                End if
        End for
        For every Lightpath in ConnectionLPs
                Lightpath.UseCapacity(currentRequest.Capacity)
                AuxiliaryGraph.CurrentConnections.Add(currentRequest)
        AuxiliaryGraph.SatisfiedRequests.Add(currentRequest)
        End For
        Routed ← True
Else
        Routed ← False
Return Routed
```

Figure 3.5: RouteRequest method of class CIGA

RouteRequest first removes the edges with insufficient capacity, temporarily. Lightpaths which have remaining capacity less than the capacity of the request, or which have reached upper limit for grooming count are removed. Wavelengths with insufficient capacity are also removed, by modifying wavelength set property of wavelength edges.

CGSP is called to find the shortest available path from current request's source to its destination. If no path is found, then RouteRequest returns false and it implies that the connection is going to be blocked.

If a path is found, existing lightpaths in the path are directly inserted into a list of connection lightpaths. Transmitters on the path imply the start of a new lightpath and when encountered, succeeding physical edges are followed until the receiver is reached. Wavelength assignment of edges on physical subpath is carried out and the new lightpath is created. Transmitter, receiver and wavelength resources are updated during the operation.

After the lightpaths are established and the graph is updated, connection is routed; i.e. capacity of the lightpaths (new and existing) used for satisfying the request are decreased by the capacity of the request.

3.2.2 Removing Connections

While removing a connection, first the capacity on the lightpaths it uses is increased by the capacity of the request, i.e. bandwidth is freed. Then, if there are some lightpaths that no longer carry any traffic, they are removed from the graph and the resources used by these lightpaths, i.e. wavelengths, transmitters and receivers are given back.

3.2.3 CGSP

CGSP, which is proposed in [10] will not be covered in detail here. CGSP is a constrained Dijkstra Shortest Path Algorithm version.

In Dijkstra, shortest paths to intermediate nodes are discovered, during the operation. A queue of shortest paths discovered, Q is maintained. At every step, shortest path p

in Q is extracted. Let the destination node of p be n_1. The cost of paths from source node s to the neighbor nodes of n_1, which pass through path p are calculated. If a lower cost path p_2 to a neighbor node n_2 is discovered by passing through p, then p_2 replaces previously found path to n_2 Q, or if there are no paths to n_2 in Q, then it is simply inserted.

The difference made by adding a constraint check is that, not all discovered paths can be inserted into Q. Here, the constraint is GWC. If for a path p, there is no possible valid wavelength assignment, then p cannot be inserted into Q.

Checking a constraint in the shortest path algorithm decreases the probability of finding a path. In order to increase the chances of finding a path, a parameter k is employed. Instead of inserting only shortest paths to intermediate physical nodes into Q, shortest paths up to an upper limit of k can be inserted.

3.3 Wavelength Selection Schemes

In function RouteRequest, wavelengths are assigned to selected wavelength edges, during lightpath establishment.

In grooming works examined, wavelength selection schemes used were not mentioned.

Random, First-Fit, Most-used and Least-Used are the most widely known wavelength assignment schemes. [1]

Random wavelength assignment is simply selecting a wavelength randomly.

In First-Fit, the wavelengths are numbered and the lowest-numbered wavelength available is selected. The idea behind this is that higher-numbered wavelengths will remain available, for setting up longer lightpaths.

Least-Used selects the wavelength that is least used in the network, in order to balance load between wavelengths. However, setting up long lightpaths gets harder unless conversion is possible.

Most-Used selects the wavelength that is most used in the network, trying to pack the connections into fewer wavelengths.

Most-Used performs the best and is followed by First-Fit, then Random and Least-Used, in terms of blocking probability of connections. However, First-Fit is easier to apply and has a lower time complexity.

In this study, two First-Fit variations are proposed: Highest-Capacity First and Lowest-Capacity First. Highest-Capacity First attempts to increase grooming ratio, by prioritizing setting up high-capacity lightpaths. This scheme is expected to decrease blocking probability in the case that transceiver resource is the critical resource, since fewer lightpaths will be established.

When transceiver count of nodes is high compared to wavelength count in a network, Highest-Capacity First Wavelength selection may lead to bad resource-utilization. Setting up a high-capacity lightpath for low-capacity requests, which can also be routed through lower-capacity lightpaths, may lead to blocking of higher-capacity requests. However, the performance of selection schemes vary in the case of critical wavelength resource, according to the order low and high requests arrive. When lower-capacity requests arrive earlier than higher-capacity requests, the case explained above arises and Lowest-Capacity First performs better. On the other hand, when higher-capacity requests arrive earlier, they can be routed and Highest-Capacity First results in the same or lower blocking ratio than Lowest-Capacity First.

3.4 LRB Grooming Policy

LR grooming policy explained in part 2.2.5 compares the total resource, i.e transmitter + wavelength usage of paths, when selecting routes. Setting up new lightpaths and using existing lightpaths is not distinguished by this method, as long as they consume the same amount of total resource. On the other hand, a mechanism which would tend to choose existing lightpaths, instead of setting up new ones when there is no difference in total resource consumption, could perform better, by increasing grooming ratio. Note that the method should have no impact when there is a difference between resource consumption of two paths.

Such a mechanism, called "LR with Beta" (LRB) is proposed here. Edge weight assignments are shown below, for implementing LRB.

$w(e) = \beta$, if $e \in E_r$ **(3.1a)**

α, if $e \in E_t$ **(3.1b)**

1, if $e \in E_w$ **(3.1c)**

$\alpha + hc(e)$, if $e \in E_l$ **(3.1d)**

Note that the only difference made to LR is the receiver weight, β. β should be very small compared to 1 and α, so that it does not have an impact when there is a difference between total transmitter and wavelength consumption of two paths. It is set to 0.001, in our study.

4. IMPLEMENTING REROUTING

As mentioned in part 2.2.7, rerouting has not been considered with single-layer solutions of routing in WDM networks. This study focused on developing rerouting methods suitable for single-layer routing on a WDM mesh network, where nodes have grooming capabilities.

In [3], on each reconfiguration step, a lightpath is selected and deleted, if the connections it carries can be rerouted through existing lightpaths. Grooming ratio of lightpaths is increased and blocking probability is dropped, by packing connections onto existing lightpaths.

This method may not work with single-layer solutions, like CIGA.

CIGA finds the shortest path on the auxiliary graph (LBAG) for each connection request. Edge weights are determined by a grooming policy. Deleting a lightpath and rerouting the connections through existing lightpaths may result in more costly paths to be selected, violating grooming policy. As a result, rerouting may cause even higher blocking probabilities.

A sample case is shown in Figure 4.1. Solid lines are physical links. The dashed lines are lightpaths. Assume that lightpaths l_2 and l_3 have enough remaining capacity to carry the connection between 1 and 2. Let l_1 be selected for deletion and the connection between 1 and 2 is rerouted through lightpaths l_2 and l_3. Compare the old and new route costs using LR:

Old route cost = $\alpha + 1$
New route cost = $2\alpha + 2$

In this case, the old route has a lower cost. If the connection is rerouted, it will consume more resource and may lead to blocking of connections from 1 to 3 and from 3 to 2.

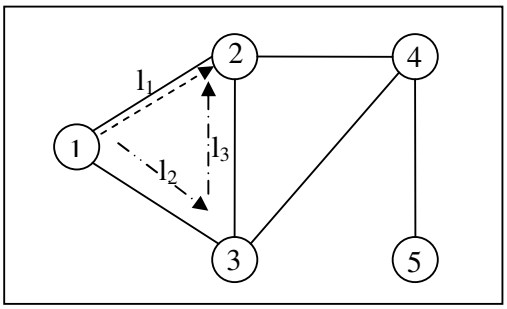

Figure 4.1: A sample case for grooming policy violation

In order to prevent such cases, a check mechanism is proposed.

4.1 Checking Cost for Deciding on Reconfiguration

At each reconfiguration attempt, before deleting the selected lightpath, it is proposed to compare the current total cost of connections to be rerouted; with calculated new total cost in the case rerouting is performed. This way, we can make sure that rerouting connections will not lead to bad resource utilization and at least will not increase the blocking probability.

Pseudo code for rerouting, with or without checking cost, is shown in Figure 4.2. A lightpath is selected for deletion at first step, and then it is deleted by calling procedure DeleteLightPath, which deletes the connections carried by lightpath, too. Then the procedure call RouteDeletedConnectionsUsingExistingLightPaths attempts to route deleted connections through existing lightpaths. If this procedure succeeds, i.e. returns true, rerouting is committed, rolled back otherwise.

```
Reroute()

lightpath ← FindLightPathToDelete()
DeleteLightPath(lightpath)

routed ← RouteDeletedConnectionsUsingExistingLPs()

if (routed)
        CommitRerouting()
else
        RollBackRerouting()
```

Figure 4.2: Method Reroute

DeleteLightPath procedure is shown in Figure 4.3. It simply removes connections hosted by lightpath from the graph, inserting them into a temporary list of removed connections. These connections might be restored with rollback, if the attempt of rerouting fails.

It might be interesting that there is no line in DeleteLightPath that removes lightpath from auxiliary graph. However, since all connections hosted by the lightpath are removed, inside the procedure call RemoveConnectionByID for the last connection, the lightpath will be removed since it carries no other connection.

```
DeleteLightPath(lightpath)

While(lightpath.ConnectionsHosted is not empty)

        connectionID ← LP.ConnectionsHosted[0]
        TemporarilyRemovedConnections.Add(GetConnectionByID(connectionID))
        RemoveConnectionByID(connectionID)

End While
```

Figure 4.3: Method DeleteLightPath

Finally, pseudo code for function RouteDeletedConnectionsUsingExistingLPs is shown in Figure 4.4. For each temporarily deleted connection, the function tries to find a path on the auxiliary graph. Before searching for a path, all physical edges are removed, since the path found should consist of only lightpath edges. If for any connection, the path search fails, then RoutedAllRequests is set to false and returned, leading to rollback of reconfiguration attempt. In the case of successfully finding new paths for all removed connections, the cost check mechanism explained earlier is applied. If total cost of new routes is not lower than the cost of old routed, RoutedAllRequests is set to false, again leading to rollback.

```
RouteDeletedConnectionsUsingExistingLPs()

routedAllRequests ← true

For Every ConnectionRequest cr in TemporarilyRemovedConnections

        currentRequest ← cr.CopyConnectionRequest
        TemporaryNewConnections.Add(currentRequest)

        RemoveAllWavelengthEdges()
        RemoveEdgesUnderCapacity(currentRequest.Capacity)

        Path ← CGSP.FindShortestPath(currentRequest)

        If (Path is null)
                routedAllRequests ← false
                EmptyTemporarilyRemovedEdges()
        Else
                RouteConnectionThroughPath(Path);
                EmptyTemporarilyRemovedEdges();
        Enf If
End For

If (routedAllRequests)
        If (TemporarilyRemovedConnections.TotalCost <= TemporaryNewConnections.TotalCost)
                routedAllRequests←false;
        End If
End If

return routedAllRequests
```

Figure 4.4: Method RouteDeletedConnectionsUsingExistingLPs

Cost check part is not implemented, in the version of RouteDeletedConnectionsUsingExistingLPs without check, and the reconfiguration attempt is rolled back only if a connection can not be rerouted.

4.2 Enhanced Reconfiguration

Reconfiguration mechanism can be enhanced further, for auxiliary graph. Since there is a grooming policy applied and the shortest path search depends on the weight costs determined by grooming policy, limiting a better route search with only lightpaths, i.e. virtual topology, still is not the ideal solution. Lower cost paths employing physical edges are ignored.

Rerouting algorithm is enhanced, by expanding the new path search with physical paths. The only change made in the algorithm

RouteDeletedConnectionsUsingExistingLPs is removing the line RemoveAllWavelegthEdges. The resulting method can be simply summarized as: Deleting a selected lightpath and rerouting each hosted connection, just like a new connection request. Note that the deleted lightpath might be established again, but the connections will be routed through different paths.

This method can be applied with or without cost check. However, checking cost is more sensible, since using more costly paths is not serving the goal of reconfiguration.

4.3 Selecting Lightpaths for Deletion

In [3], different factors were considered for selecting the lightpath to delete, at every reconfiguration step, as explained in part 2.2.7, Methods TGRR, TLR, LLR and IR were compared and the tests showed that IR performs the best, when both network disruption and resource utilization are taken into account.

IR was implemented in our solution, sorting lightpaths by checking the factors in the following order: ascending load, descending cost, ascending number of connections hosted. Thus, the most effective factor is load, followed by cost and connection count.

An enhancement was considered for this lightpath selection scheme. Checking cost deviation is proposed, instead of cost. This method is called Integrated Reconfiguration with Deviation Check (IRD).

The idea behind IRD is that, a lightpath spanning a long path of physical edges might not necessarily be using resources inefficiently; the path might be the shortest possible, between its endpoints. On the other hand, a shorter lightpath might be using a lot more physical edges then the shortest physical path possible. Thus, comparing lightpath cost deviation, which is calculated by subtracting ideal route length of lightpath from current route length, is more sensible than comparing lightpath cost.

A sample case for selecting lightpath with IR and IRD is illustrated in Figure 4.5, on 14-node NSF network. The dashed lines are routes of lightpaths l_1 and l_2. Let the consumed capacity of these lightpaths be equal. If IR is used, l_2 will be selected,

since it is the most costly lightpath. On the other hand, if IRD is used, l_1 will be selected, since its deviation is 1 (ideal route for l1 is the direct physical edge between nodes1 and 2), where l_2 has 0 deviation.

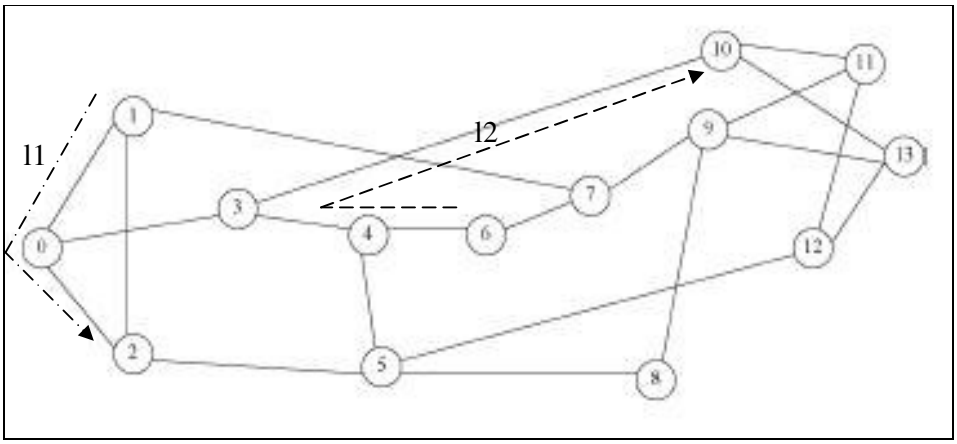

Figure 4.5: A sample case for selecting lightpath with IR and IRD

Ideal cost is computed by maintaining a copy of the initial auxiliary graph unmodified. Every time a lightpath's ideal cost is to be calculated, the shortest path is found for the lightpath on the initial path, using CGSP. This is done by creating a dummy request from lightpath source to lightpath destination and routing the request on initial graph. Procedure ComputeIdealCost is shown in Figure 4.6.

```
ComputeIdealCost(currentRequest, initialAuxiliaryGraph)

CIGA.RouteRequest(currentRequest, initialAuxiliaryGraph)
idealCost ← initialAuxiliaryGraph.LightPaths[0].PhysicalEdgeCount
initialAuxiliaryGraph.RemoveConnectionByID(currentRequest.ID)
```

Figure 4.6: Method ComputeIdealCost

5. SIMULATION RESULTS

In order to evaluate the performance of rerouting enhancements offered, they were simulated on a Five-Node Mesh network, NSF network and EUPAN network. Five-node network is shown in Figure 5.1. There are five nodes and 12 unidirectional links. NSF Network is shown in Figure 5.2. It has 14 nodes and 42 unidirectional links. EUPAN Network is shown in Figure 5.3. It has 22 nodes and 90 unidirectional links

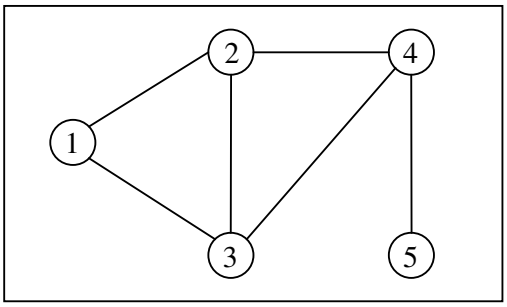

Figure 5.1: Five-node mesh network

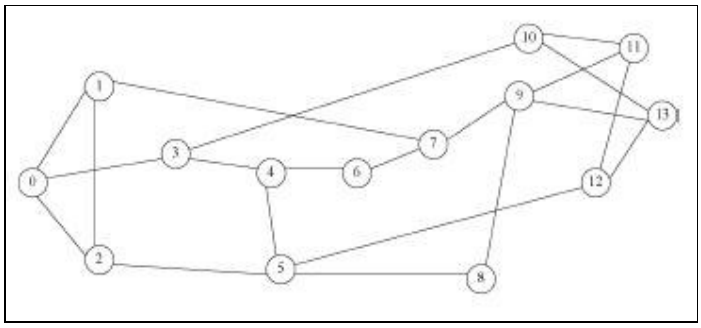

Figure 5.2: 14-node NSF network

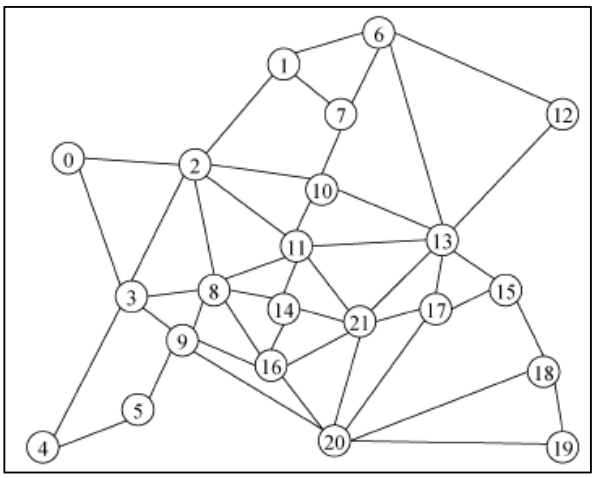

Figure 5.3: 22-node EUPAN network

Connection requests produced constitute a Poisson process with rate λ. Service time of the requests is exponentially distributed with unit mean. Grooming factor, i.e. maximum number of connections that can be carried on a lightpath is 16. Normalized connection rates are {1, 4, 16}, i.e. OC-3, OC-12 and OC-48. Total capacity of requests for each connection rate is the same.

As long as the transceiver and wavelength counts are not mentioned: Transceiver count at each node in all networks is 100. For five-node and NSF networks, on each physical link there are five OC-192 wavelengths. For EUPAN network, on each physical link there are three OC-192 wavelengths.

100.000 connection requests were produced for each simulation. The connection requests produced are written into files and then loaded from files during simulation, enabling to run different methods on the same set of connection requests, which provided clarity in comparisons.

LR grooming policy in [10] was employed for all simulations, unless otherwise is mentioned.

Performances of proposed methods were evaluated in terms of connection blocking probability (CBP). CBP is calculated as follows:

$$CBP(\Psi) = n(\varphi \in \Psi_b) / n(\varphi \in \Psi) \quad \textbf{(5.1)}$$

Where Ψ is the set of all connection requests φ and Ψ_b is the set of blocked requests.

5.1 Wavelength Selection Scheme Comparison

In part 3.3, two Wavelength Selection Schemes were proposed. It was mentioned that under the case of critical transceiver resource level, Highest-Capacity First is expected to perform better, since it attempts to establish fewer lightpaths. In order to create the case of critical transceiver resource level in NSF Network, 12 wavelengths (3*(OC-3, OC-12, OC-48, OC-192)) on each link and 20 transceivers on each node were used. The simulation results are consistent with the expectation, as shown in Figure 5.4.

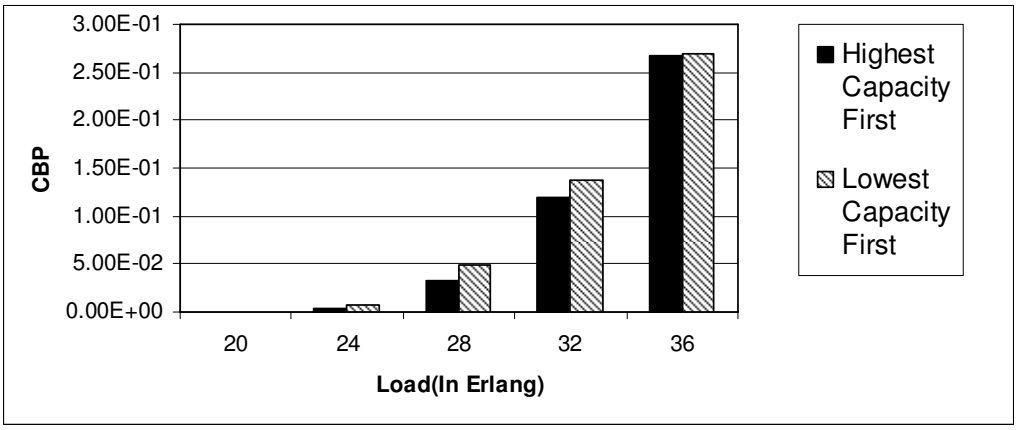

Figure 5.4: Wavelength Selection Scheme comparison in the case of critical transceiver level

5.2 LRB Grooming Policy Test

In part 3.4, an enhancement on grooming policy LR was proposed, which attempts to pack connections into existing lightpaths. The new policy tends to choose existing lightpaths, when there is no difference in terms of total resource consumption between existing and new lightpath. This policy performs better in terms of CBP, for the case of critical transceiver level, since it avoids establishing new lightpaths, when compared with LR.

Since EUPAN network involves many alternative paths for a source-destination pair, the simulation was run on EUPAN network in order to get clearer results. Critical transceiver level was obtained by setting 9 transceivers at each node and 3 wavelengths of capacity OC-192 on each physical link. Results are shown in Figure 5.5

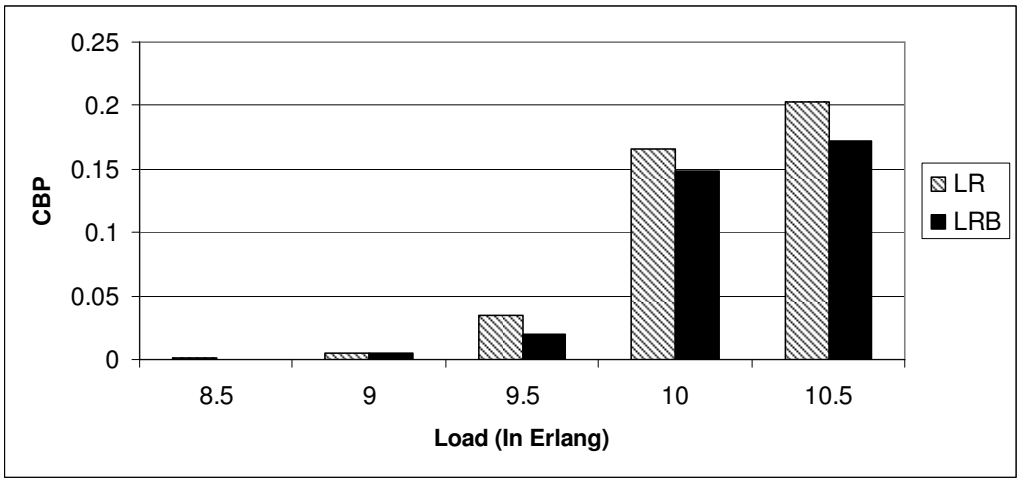

Figure 5.5: Grooming Policy comparison in the case of critical transceiver level

5.3 Cost-Check for Deciding on Reconfiguration Results

It was explained in part 4 that deleting lightpaths and rerouting connections through existing lightpaths can not give better results in terms of blocking probability, for single-layer grooming solutions, since it violates the grooming policy. An enhancement by checking the cost of old and new routes was proposed, and it was explained that new routes would at least not lead to worse resource utilization.

The simulation results for five-node mesh network illustrating the explained facts are given in Figure 5.6. Table 5.1 shows results in detail, since it is not easy to see the difference between no reconfiguration and reconfiguration with cost check on Figure 5.6. The results were obtained for disconnected triggering policy.

Reconfiguration without cost check performs even worse than not reconfiguring at all. Results for reconfiguration with cost check are slightly better than no rerouting results. However, the improvement is not sufficient for employing rerouting, since connections are disturbed, at expense.

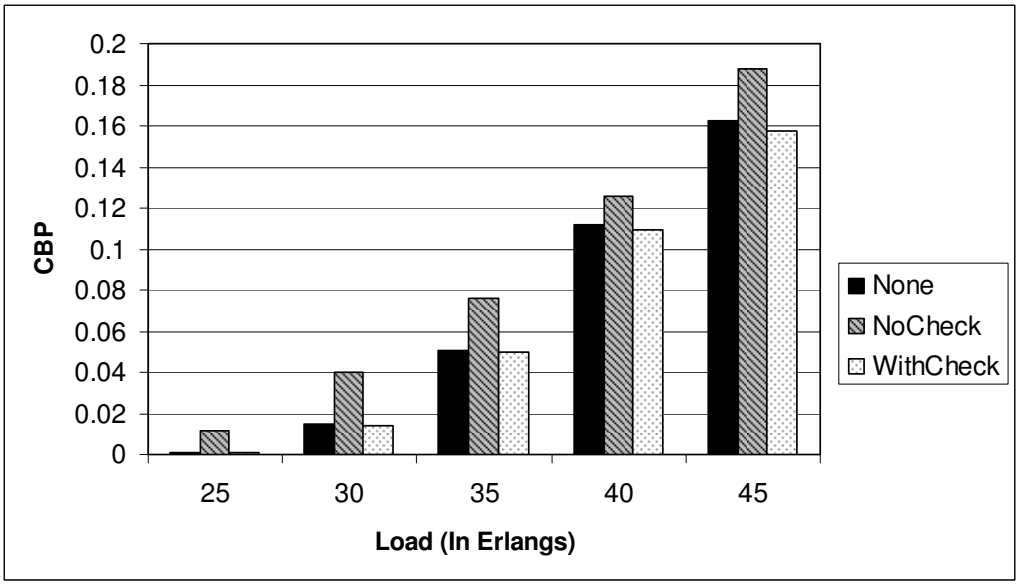

Figure 5.6: Cost-check simulation results for five-node mesh network

Table 5.1: Detailed cost-check simulation results in terms of CBP

	25	30	35	40	45
None	0.00094	0.01431	0.05033	0.11156	0.16217
NoCheck	0.01124	0.04028	0.07561	0.12555	0.18789
WithCheck	0.00094	0.01427	0.05006	0.10967	0.15786

5.4 Enhanced Reconfiguration Results

To further enhance reconfiguration, it was proposed that after deleting lightpaths, the whole auxiliary graph is searched for new routes, instead of only lightpaths. In other words, grooming policy is applied to rerouting, too.

The simulation results for five-node mesh network comparing enhanced reconfiguration (LPDelE) with reconfiguration through existing lightpaths (LPDel) are given in Figure 5.7 and Table 5.2. The results were obtained for disconnected triggering policy and cost check was applied for both.

Enhancement dropped blocking probability more significantly.

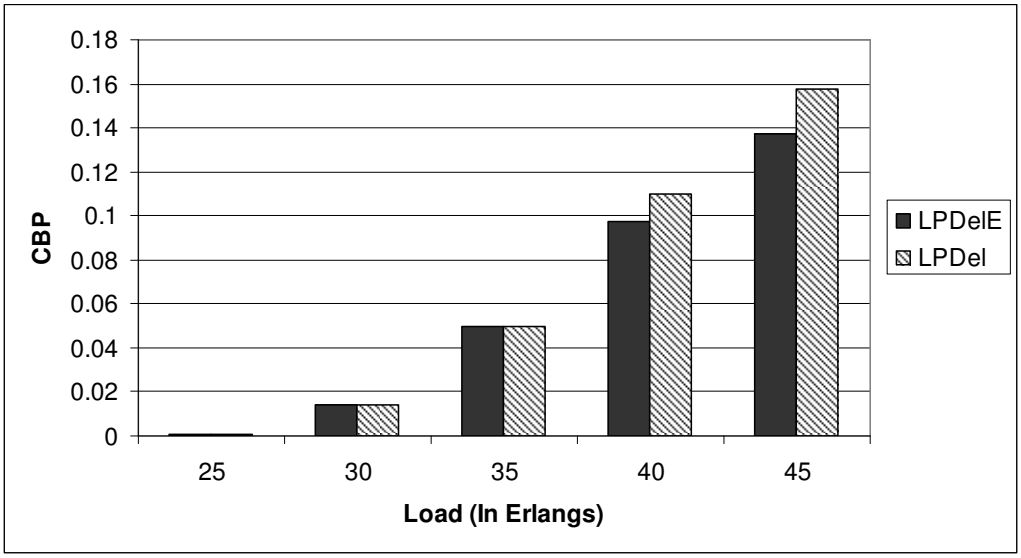

Figure 5.7: LPDelE results for five-node mesh network

Table 5.2: Detailed LPDelE results in terms of CBP

	25	30	35	40	45
LPDelE	0.00097	0.01425	0.04949	0.09704	0.13715
LPDel	0.00094	0.01427	0.05006	0.10967	0.15786

Since disconnected triggering policy is not acceptable due to disturbing connections too frequently, periodical time-triggered policy was employed for the rest of simulations.

Table 5.3 gives blocking probability results for enhanced reconfiguration method, under disconnected and periodical time-triggered policies, along with no reconfiguration results. Figure 5.8 shows rerouted connection counts for two triggering policies.

Blocking probability gain might seem to be low, but the number of disturbed connections is significantly lower, so employing reconfiguration can be sensible. Furthermore, the period can be further adjusted for decreasing blocking probability.

Table 5.3: Blocking probabilities for enhanced reconfiguration method under disconnected and periodical time-triggered policies

	25	30	35	40	45
■ Disconnected	0.00097	0.01425	0.04949	0.09704	0.13715
▧ Periodical	0.00097	0.01425	0.05029	0.10163	0.14915
□ None	0.00094	0.01431	0.05033	0.11156	0.16217

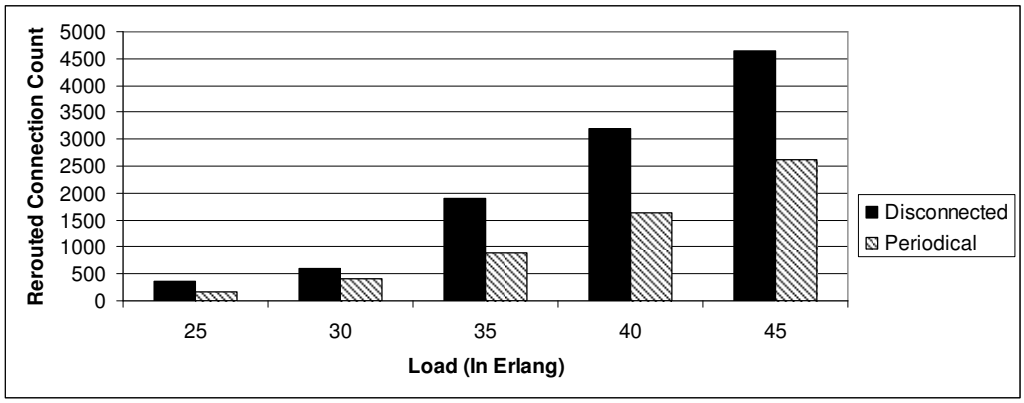

Figure 5.8: Rerouted connection counts for two triggering policies

5.5 IRD Results

Another enhancement was for selecting lightpath to delete. Instead of checking cost, it is proposed to check cost deviation, in IR.

The simulation results for five-node mesh network comparing IR and IRD are given in Figure 5.9 and Table 5.4. The results were obtained for periodic time-triggered policy and cost check was applied for both.

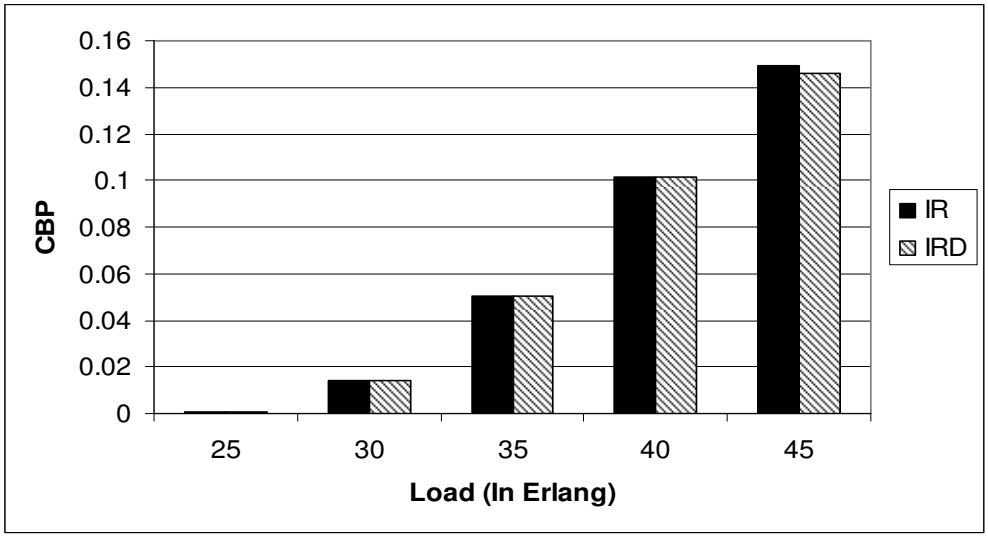

Figure 5.9: Simulation results for five-node mesh network comparing IR and IRD

Table 5.4: Detailed IR and IRD results in terms of CBP

	25	30	35	40	45
IR	0.00097	0.01425	0.05029	0.10163	0.14915
IRD	0.00096	0.01423	0.05034	0.10127	0.14593

Examining the results, IRD performs better than IR, in terms of blocking probability. However, since there are not many alternative paths for source-destination pairs in five node network, cost deviations of lightpaths are usually 0 (zero). Thus, the difference made by implementing IRD instead of IR is not very significant. On the other hand, in EUPAN network which is denser, i.e. has many alternative paths for source-destination pairs, the difference is more significant. Simulation results for EUPAN network are given in Figure 5.10.

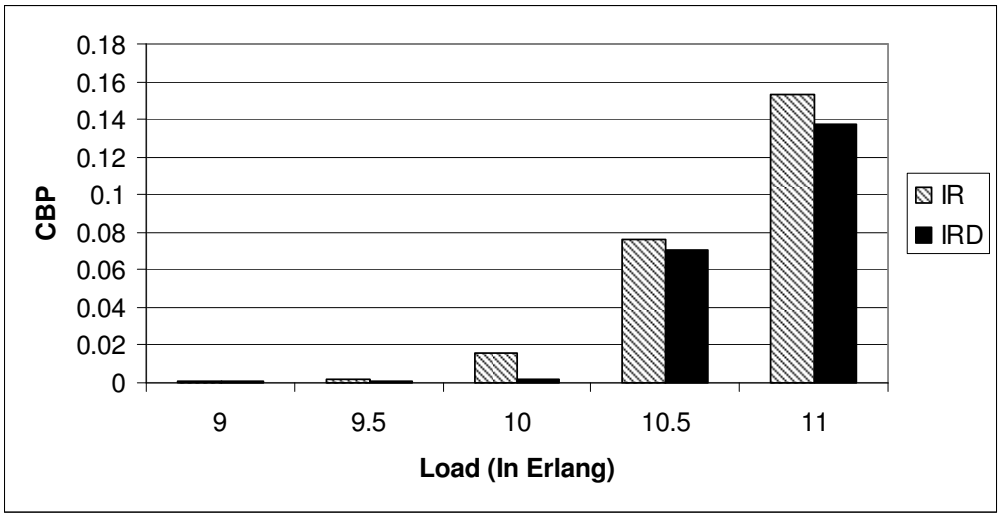

Figure 5.10: Simulation results for EUPAN network comparing IR and IRD

The last simulation is a summary of all the enhancements offered. In Figure 5.11, NSFNET results for IRD with periodic time-triggered policy and cost check are shown. Using LPDelE with IRD, significant performance increase is obtained in terms of blocking probability.

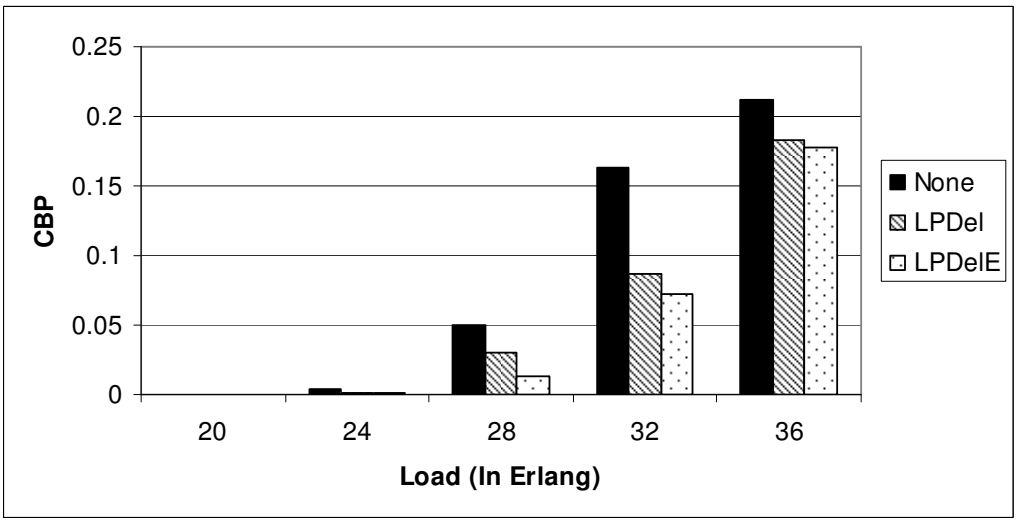

Figure 5.11: Simulation results for NSFNET

6. CONCLUSION AND FUTURE WORK

In this study, different traffic grooming solutions for dynamic traffic case were examined. A single-layer solution implementing an auxiliary graph was selected, due to better performance in terms of blocking probability, ability to reflect heterogeneous network, simplicity in applying grooming policies by edge weight assignments and tractable graph size. Two wavelength selection schemes and a grooming policy were proposed and simulated on the graph model.

On the selected graph model, rerouting methods for preventing bad resource utilization and indirectly decreasing the blocking probability of future requests were applied.

Rerouting concept has not been widely considered for dynamic traffic. Furthermore, existing works on dynamic traffic rerouting concentrate only on connection rerouting, i.e. handle rerouting only on virtual layer. In this study, starting with an earlier method which selects a lightpath for deletion at every reconfiguration attempt and tries to route effected connections through existing lightpaths (virtual topology), methods for single-layer rerouting were proposed.

6.1 Simulation Results

Proposed wavelength selection schemes were simulated on NSF network and the obtained results proved that using Highest-Capacity First selection scheme for critical transceiver resource level leads to lower blocking probability.

Proposed grooming policy LRB was simulated on EUPAN network along with LR grooming policy proposed in [10]. Results showed that LRB provides improvement in blocking probability, when critical resource in the network is transceivers.

Proposed enhancement methods for single-layer rerouting considering grooming were simulated for five-node mesh, NSF and EUPAN network topologies, under different traffic densities.

- In the first simulation, an earlier method was applied: at every reconfiguration attempt a lightpath is selected for deletion and effected connections are routed through existing lightpaths. Without checking if the total cost of new routes is lower than the old cost, this method led to even higher blocking probabilities than no rerouting, for auxiliary graph model. Adding cost-check mechanism decreased blocking probability, to a level lower than no rerouting.
- Expanding a better route check to the whole auxiliary graph (LPDelE), instead of restricting it with existing lightpaths (LPDel), decreased blocking probability more significantly, by more efficiently utilizing resources.
- In selecting lightpath for deletion, checking cost deviation (IRD) instead of cost (IR) was proposed. The idea behind this enhancement is to discover the routes that utilize resources badly and get rid of them. Simulation results showed that smaller blocking probabilities can be achieved, implementing cost deviation comparison, especially in dense networks like EUPAN.

6.2 Future Work

In [9, 11], it is asserted that under dynamic scenario, splitting or rerouting a lightpath is not acceptable, since it leads to disturbing several connections. However, the terms of Service Level Agreements (SLAs) vary from client to client. Some clients may accept short interruptions in traffic, while some may not tolerate any interruption. A parameter reflecting the interruption tolerance of client can be added to connection request tuple. Then in the lightpath selection phase, this parameter can be aggregated with other factors, for sorting lightpaths. Or, alternatively, after lightpath selection, each connection's interruption tolerance can be checked, before deciding to perform reroruting.

REFERENCES

[1] **Mukherjee, B.,** 1997. Optical Communication Networks. McGraw-Hill, New York.

[2] **Modiano, E. and Lin, P**., 2001. Traffic Grooming in WDM networks, *IEEE Communications Magazine,* **39,** 124-129.

[3] **Akgün, İ. and Buzluca, F**., 2006. Virtual Topology Reconfiguration on Optical WDM Networks Considering Traffic Grooming, *Optical Switching and Networking,* **3,** 11-23.

[4] **Gerstel, O., and Kutten, S**., 1997. Dynamic Wavelength Allocation in All-Optical Ring Networks*, IEEE ICC*, Montreal, Quebec, CA, June 8-12, 432-436.

[5] **Zhu, H., Zang, H., Zhu, K. and Mukherjee, B**., 2003. A Novel Generic Graph Model for Traffic Grooming in Heterogeneous WDM Mesh Networks, *IEEE/ACM Transactions on Networking,* **11,** 285-299.

[6] **Zhu, K. and Mukherjee, B**., 2002. Traffic Grooming in an Optical WDM Mesh Network, *IEEE J. Select. Areas Commın.,* **20,** 122-133.

[7] **Zhu, K., and Mukherjee, B**., 2002. On-line Approaches for Provisioning Connections of Different Bandwidth Granularities in WDM Mesh Networks*, Proc. OFC*, Anaheim, CA, March 17-22, 549-551.

[8] **Ho, O., and Lee, M**., 2005. Practical Dynamic Traffic Grooming in Large WDM Mesh Networks*, Proc. 2nd International Conference on Broadband Networks*, Boston, Massachusetts, USA, October 3-7, 1194-1196.

[9] **Zhu, H., Zang, H., Zhu, K. and Mukherjee, B**., 2002. Dynamic Traffic Grooming in WDM Mesh Networks Using a Novel Graph Model*, Proc. IEEE GLOBECOM*, Taipei, Taiwan, November 17-21, 2681-2685.

[10] **Yao, W. and Ramamurthy, B**., 2005. A Link Bundled Auxiliary Graph Model for Constrained Dynamic Traffic Grooming in WDM Mesh Networks, *IEEE J. Select. Areas Commun.,* **23,** 1542-1555.

[11] **Madhyastha, H.V. and Siva Ram Murthy, C**., 2004. Efficient Dynamic Traffic Grooming in Service-Differentiated WDM Mesh Networks, *Computer Networks,* **45,** 221-235.

[12] **Park, M.H., and Choi, J.S**., 2004. An Implementation of Optical Network Design and Evaluation Simulator for Wavelength Routed Optical Networks*, Applied Communications Symposium*, Arlington, Virginia, USA, April 18-22, 143-148.

CURRICULUM VITAE

Pınar Karagülle was born in Istanbul in year 1982. She received the B.S. degree in Computer Engineering from Istanbul Technical University, in 2004. Since then, she has been working as a Software Developer at Bizitek Software Development & Internet Technologies Inc. and working toward the M.S. degree in Computer Engineering at Istanbul Technical University.