Spring 4-14-2011

OFFLINE OPTIMIZATION OF ADVANCE RESERVATION OF BANDWIDTH OVER DYNAMIC CIRCUIT NETWORKS

Pragatheeswaran Angu
University of Nebraska-Lincoln, pangu@cse.unl.edu

OFFLINE OPTIMIZATION OF ADVANCE RESERVATION OF BANDWIDTH OVER DYNAMIC CIRCUIT NETWORKS

by

Pragatheeswaran Angu

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Byravamurthy Ramamurthy

Lincoln, Nebraska

May, 2011

OFFLINE OPTIMIZATION OF ADVANCE RESERVATION OF BANDWIDTH OVER DYNAMIC CIRCUIT NETWORKS

Pragatheeswaran Angu, M.S.

University of Nebraska, 2011

Advisor: Byravamurthy Ramamurthy

E-science projects require very high-speed and reliable networks to transfer data across various destinations in the world. Dynamic Circuit Network (DCN) is a networking service to make advance reservation of bandwidth between a source and a destination in a network. In this thesis we solve the problem of advance reservation of bandwidth in next-generation wavelength-division multiplexing (WDM) networks using a simulation based approach.

We implement a greedy algorithm and a genetic algorithm in parallel, in separate threads. The request for advance reservation is processed by both but the user gets the response only from the greedy algorithm. The genetic algorithm is used for offline re-optimization where we optimize the schedule of all the future reservations thereby maximizing the number of reservations possible in the network. We evaluate the approach using trace-driven traffic and simulated traffic. We observed an improvement in the blocking probability and service blocking probability of the network using our approach.

ACKNOWLEDGEMENTS

I am very grateful to my advisor, Professor Byrav Ramamurthy, for his guidance, patience and research support throughout my masters study. His technical insights, editorial comments, generous encouragement, and financial support were invaluable in completing my thesis research. I feel very privileged to have had the opportunity to work with him during my M.S. study.

I would like to thank Dr. Lisong Xu and Dr.David Swanson for serving in my committee. I would like to thank Dr. Xi Yang (USC/ISI), Tom Lehman (USC/ISI) for their help in configuring DRAGON testbed in UNL. I would like to thank Chin Guok and Evangelos Chaniotakis of Lawrence Berkeley Lab for providing OSCARS network trace. I would like to thank Kent G Christensen (IS, UNL) for helping me in switch configurations at various time. I would like to thank Dr. James Sterbenz, Justin Rohrer, Ali Sydney, Ramkumar Cherchuri, Xuan Liu of GpENI. I would like to thank Garhan Attebury and Carl Lundstedt of HCC, UNL. I would like to thank Robert Ricci of University of Utah for collaborating in a DCN demo. I would like to thank my friends Lin Liu, Mukesh Subedee, Shivashis Saha for their valuable help during my M.S study. I would like to thank my family for their love and affection towards me.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Overview of Dynamic Circuit Networks

The best-effort IP routing [22] of Internet is its greatest strength as well as its weakness. Best-effort routing is simple, opportunistic and resilient. It does not have predictability and flexibility and also does not provide any guarantees. Todays sciences are increasingly driven by highly-distributed large-scale multidisciplinary collaborations. These distributed applications rely on high-speed networks to process and integrate data from various sources for observations and simulations. The success of these collaborations depend upon the performance of high-speed networks.

Dynamic Circuit Network (DCN) [1, 25] (now called Inter-operable On-Demand Network) is a networking service in Internet2 that provides the researchers the ability to create short circuits of large bandwidth across the network. These circuits are created for the bandwidth-intensive applications that are run over the Internet2 backbone network. This service uses both the software components of OSCARS [2] and DRAGON [3, 26] to create dynamic circuits across various domain and across various network technologies. The circuits are created and deleted using the Web User

Interface provided by the OSCARS software components. The Inter Domain Controller (IDC) specification was created by the DICE Control Plane Working Group [4] which is basically the entity managing the circuit creation and deletion along with user authentication and authorization mechanisms in an Autonomous System (AS) or local domain.

On-Demand Secure Service and Advance Reservation System (OSCARS) [2] is a networking service deployed in the DoE ESnet to create dynamic, deterministic and secure circuits across the ESnet network. MPLS [5] and RSVP [6] are the key protocols used to create advance reservations of bandwidth using the software components developed as part of the OSCARS project. The Label Switched Path (LSP)s are created using MPLS both in Layer 2 and Layer 3 using OSCARS software. The circuits are created and deleted using a web interface provided by the OSCARS and hence this method is adopted in the DCN/ION project as the interface for managing virtual circuits. The major software components of OSCARS are Reservation Manager (RM), Path Setup Subsystem (PSS) and Bandwidth Scheduler Subsystem (BSS), Authentication, Authorization and Auditing Subsystem (AAA). The RM, PSS, BSS subsystems are used for reserving resources and creation and deletion of actual circuits in the network and AAA is used to provide authentication mechanisms using X.509 certificates.

Dynamic Resource Allocation via GMPLS Optical Networks (DRAGON) [3] was a NSF funded project to dynamically provision network resources across various domains and across heterogeneous networking technologies. GMPLS [7] is the key protocol used to create circuits spanning across both optical and Ethernet domains and hence DRAGON creates a layer2 virtual circuit. A set of software components has been developed to leverage this capability across a testbed in the Washington D.C area. The major components of DRAGON software are VLSR (Virtual Label

Switched Router), NARB (Network Aware Resource Broker), ASTB (Application Specific Topology Builder) and RCE (Resource Computation Engine). As DRAGON provides the capability to create circuits that span across different domains the NARB acts as the entity that represents a local domain or Autonomous System (AS). In each domain each switch needs to be configured separately for creating a circuit and hence VLSR acts as the entity controlling the switches. The RCE and ASTB are used for computing the resources required for creating circuits. Hence a particular DRAGON domain will have a NARB and one or more VLSRs depending upon the number of switches in the domain.

1.2 Overview of Lightpath Scheduling

Wavelength-routed networks [33, 34, 35, 36, 37, 38] are nodes interconnected by optical fibers of very high bandwidth. An optical fiber can carry multiple wavelengths using wavelength division multiplexing (WDM) technology [33]. WDM extends the bandwidth capability of an optical fiber to Terabits per second (Tbps). The emergence of Optical Cross-Connects (OXCs) further facilitates building networks with enhanced efficiency and survivability. The input from one optical fiber is switched to one of the output fibers using OXC. The reconfigurable OXCs enables intelligent and automatic provisioning at the optical transport layer [32]. In order to provision an end-to-end circuit, the routing and wavelength information need to be assigned. The route and wavelength together is referred as lightpath [36] and it offers an end-to-end tunnel for data transmission.

Many highend applications such as grid technologies require next-generation wavelength-division multiplexing (WDM) [33] optical networks by the provisioning of lightpaths in an on-demand fashion. The efficiency of network operations could be affected such

dynamic demands as the demands are usually static and predictable in nature. An user prefers to book reservation of end-to-end lightpaths in advance for predefined durations where the starting time can be several weeks to days in future. These advance reservations of lightpath is called scheduled lightpath demand (SLD). If the complete set of lightpath demands is known in advance it is called static scheduled lightpath demand (S-SLD) and if it is not known in advance it is called dynamic scheduled lightpath demand (D-SLD).

1.3 Motivation

May end users require deterministic and reliable service in many practical network operations. In a deterministic service a user expects a deterministic answer for his request. A deterministic response will be yes if his request can be accomodated in the system or no if his request cannot be accommodated by the system. It is difficult to achieve high network utilization while providing deterministic service in the presense of D-SLDs. The arrival S-SLDs is previously known and hence we can achieve maximum network utilization possible in the network. In case of D-SLDs it is difficult to achieve maximum network utilization as we cannot predict the arrivals of D-SLDs. In case of D-DSLDs we can only give deterministic answer that is optimal for the current scenario, but however it will be sub-optimal with the arrivals of future D-SLDs. Hence any change in the routing and wavelength information before actual provisioning of D-SLDs will not dirupt its service. Therefore, we can perform re-optimization for all D-SLDs scheduled to be set up in future.

1.4 Summary of Contributions

We present a simulation based approach to solve the problem of re-optimization of lightpath scheduling. In this approach we present a greedy algorithm and genetic algorithm to solve the problem of advance reservation of bandwidth over the backbone networks. We introduce the term dynamic scheduled bandwidth demand (D-SBD) which is analogous to the D-SLD but this allows a portion of the bandwidth of a wavelength to be requested over a period of time. In this paper we study the problem of scheduling D-SBDs and we assume that the network is slotted. The duration of a scheduled lightpath is measured in number of time slots and each time slot is of equal length. We study the reservation problem with two types of algorithms: greedy algorithm and genetic algorithm. They are run in parallel in separate threads so that the greedy algorithm interacts with the user and the genetic algorithm performs the offline optimization of all the future reservation requests.

In this thesis, we compare the performance of our approach with the re-optimization at blocking method described in [27]. We carry out simulation experiments based on the two algorithms running in parallel using the real network trace of OSCARS database as well stochastic traffic over two network topologies, ESnet network topology and a 24 node network topology. The performance results show that our approach performs much better than using the re-optimization at blocking method as well as using just the greedy algorithm.

1.5 Outline of the Thesis

The rest of the thesis is organized as follows. In Chapter 2 we describe the Related Work. Chapter 3 describes the simulation based approach and a solution to the problem of D-SBD. The Chapter 4 concludes the thesis with future work.

Chapter 2

Background and Related Work

2.1 Dynamic Circuit Network (DCN)

DCN [8] is a networking service in Internet2 used to create circuits of short-term between users who require dedicated bandwidth. DCN is used by users who require reliable connections that lasts from minutes to hours to days. The DCN over Internet2 is also called Inter-operable On-Demand Network (ION) [1]. DCN enables users to create point-to-point circuits across the Internet2 network. The setup and teardown of the circuits is automated by the use of control plane software. Multiple administrative domains must coordinate to create optical circuits across different autonomous systems. This requires each autonomous system to have a control plane software running so that the circuit creation across these different domain boundaries along with authentication and authorization is performed by this control plane software. This control plane software was developed through collaboration between several projects including the NSF-funded DRAGON project [3], the ESnet OSCARS program [2], and the GEANT2 AutoBAHN project [9]. the University of Southern California/Information Sciences Institute East (USC/ISI-East), Mid-Atlantic Cross-

roads (MAX), the University of Amsterdam, Nortel, and other regional and national networks and other participants involved in the development of this control plane software. The interdomain protocol is developed within GLIF and a working group is formed at OGF for standardizing the DCN protocols. The DCN software used by Internet2 is an open source software which is also called the ION software suite. This include both the control plane software developed by DRAGON and ESnet/Internet2 developed OSCARS software suite. The ION software suite is available as open source to all and hence other institutions can use it to create their own local DCN domains.

2.1.1 OSCARS

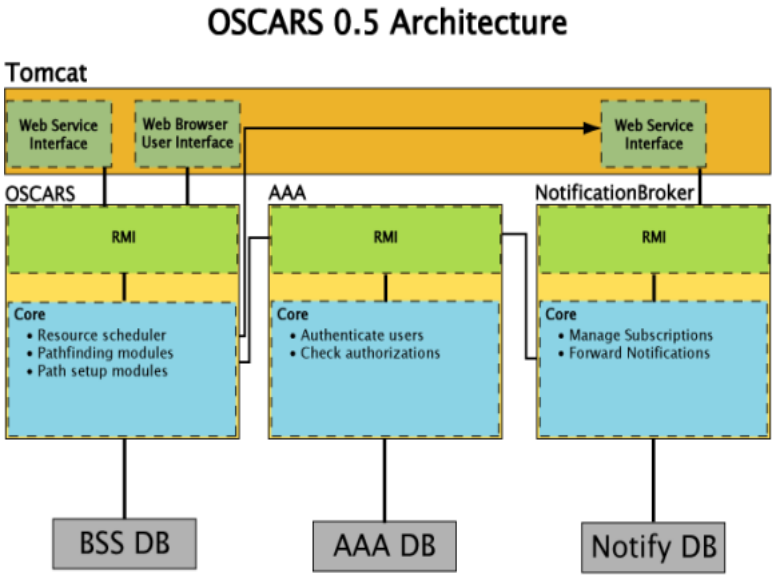

Figure 2.1: OSCARS Architecture

source:https://wiki.internet2.edu/confluence/display/DCNSS/OSCARS+0.5+Architecture

ESnet [2] network of Department of Energy (DoE) connects to over 100 other networks and the ESnet network if formed between over 40 institutions. ESnet is used for LHC and other E-science projects. Reliable connections and high-speed bandwidth to researchers at various universities, national laboratories and other research

institutions are the primary goals of ESnet. This enables the research institutions to do collaborative projects that range from climate science, energy and the origins of the universe. The On-demand Secure Circuit and Advanced Reservation System (OSCARS) [22] of ESnet enables user-driven advance reservations of dynamic virtual circuits at layer 3 (IP) and layer2 (Ethernet VLANs). Hence OSCARS is used as a local controller of the ESnet domain to manage the network resources of ESnet. OSCARS is also uses ad IDC (Inter-Domain Controller) [22] and it communicates with other domain controllers to provision an inter-domain circuit. OSCARS has a web based interface to create and manage circuits providing an easy to use interface for the network administrators. However the users are expected to have prior knowledge of network to use this web based interface to setup VCs. The requests by users can be in advance and can coincide with the timing of the data generation for an experiment distributed over the network. There are two interface provided by OSCARS for circuit creation. One is the SOAP-based messaging system that can be used by the programmer to communicate with OSCARS and other is the web browser based interface that can be used by a general user.

In layer2 VC the VLAN number is negotiated at the time of request and it is used to identify the packets that are for the particular layer2 VC. The customer edge device is configured for correct VLAN so that the packets will be tagged correctly before entering the provider edge (PE) device.

In layer3 VC, the IP flow specifications are used filters on the PE device to filter out the packets in to layer3 VC. The parameters used for this purpose are source, destination, address, port and protocol.

The algorithm used by OSCARS to calculate the path for a reservation of a circuit is based on graphs. The capacity and topology information is obtained periodically once an hour from the network devices. Then, it is imported to the topology database

of the OSCARS. Whenever a new request for circuit reservation is received, a topology graph is created from the topology database considering any existing time overlapping reservations into account.

A virtual circuit can be instantiated by an alarm-clock process which queries the database periodically for reservations when their start times have reached or by a trigger from the user. The alarm-clock process is the one which take care of the end times for all reservations, though a tear down of the circuit can be triggered by a user much before its end time.

The reservations creation and tear down needs to be done in an auditable and secure manner and also the limited resources of the network devices need to be configured in a secure way. The X.509 certificates are used for signing messages coming through the SOAP API. The web browser based requests are authenticated using username and passwords. The authorization information is stored in database tables and the policies of authentication are grouped by role attributes that are assigned to each user.

A recently founded project called DYNES [10] is promoting the adoption of DCN/ION services by regional and campus networks. UNL is also part of the DYNES project and we are looking forward to make use of DYNES to experiment with our research problems.

2.1.2 DRAGON

The major goal of DRAGON project [3, 25] is to conduct research and develop technologies based on the research to enable dynamic circuit provisioning across the network resources. The DRAGON project also takes care of the inter-domain dynamic circuit creation across several heterogeneous network technologies. The DRAGON project uses the emerging next generation optical networking technologies to demon-

strate and develop the flexibility and power of a hybrid circuit and packet switched network.

Provisioning deterministic service across various domains and heterogeneous network technologies is the key objective of DRAGON software. Mostly, important scientific computing resources are distributed across various places around the world and each resource is administered by various autonomous domains and hence provisioning circuits across various domains is the only solution to interconnect all these resources. Each administrative domain uses technologies that are most suitable to their domain based on various needs. For e.g. DRAGON installation in Washington, DC uses all-optical dense wavelength division multiplex (DWDM) infrastructure with the support for multiprotocol client interfaces. Another administrative domain can use Ethernet, SONET [21] or any technology. Hence to share resources in such a diverse multidomain environment there is certainly a need to provision circuits across these heterogeneous networking technologies. The traffic engineering (TE) constrains is addressed by the use of GMPLS protocol as it suits the most for a heterogeneous environment. GMPLS also addresses path computation, routing and signaling in this multidomain environment.

2.2 GpENI

Great Plains Environment for Network Innovation (GpENI) [28, 11] is a programmable regional network testbed operated by the collaborating universities: University of Nebraska Lincoln (UNL), University of Kansas (KU), Kansas State University (KSU) and University of Missouri Kansas City (UMKC). The goal of GpENI is to provide a programmable network infrastructure to conduct experiments in Future Internet architecture. Each participating GpENI university has a Netgear GSM7224 Ethernet

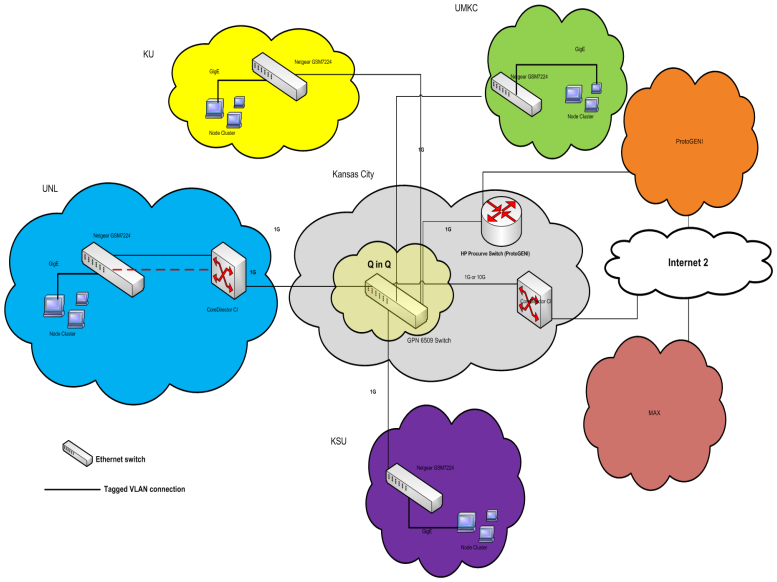

Figure 2.2: GpENI Network Diagram

switch which can be used for creating regional DCN testbed. We modified the open source DCN source code to support the Netgear switch and deployed the regional DCN testbed in GpENI.

Q in Q or Double tagging [12] is a method to add one more outer VLAN tag to already tagged packets. This is used by the Internet Service Providers to separate network traffic between different use groups so that one user group will be isolated from other group. However each member in a group can have their packets tagged differently so that they can protect their packet from other members of the same group. Q-in-Q is configured in the GPN switch (Fig. 2.2) [16] so that it acts as a pass through for packets of any VLAN tag generated by any of the GpENI universities. Hence to connect to MAX in this method we need to just include the interface of the GPN switch which is connected to the Juniper router of the Kansas City PoP in the Q-in-Q cloud. In this manner we can create circuits of any VLAN tag from any of the GpENI university to the MAX. The IDC and VLSR are placed in UNL and we have configured one VLSR per switch in GpENI. The above configuration will enable us to

create a DCN circuit between GpENI universities and also from a GpENI university to MAX through Internet2. Since the GPN switch is used for production traffic the network operators of GPN considered including the Internet2 port in GPN to GpENI QinQ as a potential risk. Hence whenever we want to connect to MAX the network operators of GPN configured a static VLAN in GPN switch so that UNL can establish the DCN circuit with MAX only in that predefined VLAN id. In this configuration the UNL port of GPN switch is taken off from GpENI QinQ configuration. When we want to establish DCN circuit from UNL to any of the GpENI university the UNL port is included in the GpENI QinQ and hence we cannot connect to MAX in this scenario.

We have shown the demo of intra domain DCN circuit creation within GpENI in various GENI conferences (GEC8 - GEC10). We have also demonstrated the inter-domain DCN circuit creation between GpENI and MAX (a regional network based in Washington D.C.) in several conferences [17]. The intra-domain demo involves creating a DCN circuit between any of the GpENI universities by specifying start time, end time, source, destination and an arbitrary VLAN number. The inter-domain circuit creation with MAX uses the same parameters as intra-domain except that VLAN id the one that is statically configured in GPN switch. Our work in GpENI is described in several deliverables [13, 14, 15] which are available in the GpENI wiki.

2.3 Lightpath Scheduling and Advance Reservation

The lightpath scheduling problem is primarily driven by highly distributed applications, which require end-to-end lightpath connections to be scheduled in advance for

a specific duration. The best example of such an application can be the grid applications. The other applications that are based on regional and national optical networking infrastructure, such as National Lambda Rail (NLR) [40], Internet2 Hybrid Optical Packet Infrastructure (HOPI) [39], TeraGrid [42] and UltraLight [41] also look into the problem of provisioning reliable circuits across various administrative domains. Scheduling of a real-time, collaborative experiment can be one of the best examples of such an application. In these applications, a lightpath needs to be reserved in advance between data processing and data collecting facility, starting at 9:00 pm and ending at 10:00 pm on Sunday.

In order to establish a lightpath in advance, the route and wavelength assignment (RWA) need to be found first. The RWA information need to be calculated for the duration of the lightpath request. Such a lightpath demand is called scheduled lightpath demand (SLD). A SLD is represented by a tuple (source, destination, starting time, duration) where the duration and starting time are the extra parameters when compared to the lightpath demand without scheduling. The network need to be slotted inorder to support SLDs in wavelength routed networks. The wavelength availability is considered to be independent in each time slot.

The advance reservation of network resource is a common problem communication networks to accommodate the future traffic demands in a network. In [49], the advance reservation and path computation complexity is studied and they advance reservation problem is formulated for generic networks. The authors found that some problems are NP-complete while some can be solved in polynomial time. In [43], the advance reservation properties are discussed by authors and they proposed an architecture to improve the network performance by use of a bandwidth broker which has the knowledge of all the advance reservations. The Globus Architecture for Reservation and Allocation (GARA), presented in [44], deals about reservations of network

resources in advance for Grid. The authors also discuss about the reservation of heterogeneous network paths for Grid computing and they propose a hierarchy of network resource for integration of path management with authentication service.

In [19], the advance reservation along heterogeneous network topologies are discussed and the authors have proposed a hierarchy of network resources to integrate the management of paths with authentication services and Grid information. The advance reservations certainly reduce the utilization of resources when compared to the immediate reservations because of increased fragmentation of network resources [18]. In [30] a sliding traffic model is proposed by the authors and an algorithm to resolve demand time conflicts is proposed to minimize the network resource wastage. In [23] a Flexible Advance Reservation Model (FARM) is proposed by the authors. The authors also discuss about its implementation and the results show that this approach increases the network resource utilization and the acceptance rate of the bandwidth demands.

There are more constraints in the lightpath scheduling problem in wavelength routed networks. Some of them are wavelength continuity constraint, path length constraint. Most of the previous work concentrated on static lightpath scheduling problem. In static lightpath scheduling problem the set of SLDs are known in prior. In [29], a traffic is proposed by authors for reservation of lightpaths in advance in wavelength routed networks. In [24], time-disjointedness of the scheduled lightpath demands are taken in to consideration and it is modeled as a combinatorial optimization problem. The objective of this model is to minimize the total number of wavelengths used. A tabu-search mechanism is used to solve this combinatorial optimization model of the problem.

2.4 Routing and Wavelength Assignment

The routing and wavelength assignment (RWA) [45] problem finds the route for the requested lightpath along the network topology and also finds a wavelength to be assigned on each link throught the route. There three types [45] of lightpath request: static, incremental and dynamic. In static type all the future connections are known in advance and hence it is possible to find the best route and wavelength assignment information so that we minimize the network resources used and maximize the network utilization. The routing and wavelength assignment problem for static traffic is called Static Lightpath Establishment (SLE) problem. The SLE problem is formulated as mixed-integer linear program in [46]. SLE is a NP-complete problem [21]. We can partition the SLE problem in to two subproblems, one is the routing problem and the other is the wavelength assignment problem. This approach is described in [45], where each of these subproblems is solved separately to solve the whole problem. In incremental traffic, the lightpath demands arrive sequentially and the end time of each lightpath requested is infinity i.e. the lightpath remains int he network forever. In case of dynamic traffic, the RWA for each lightpath request is found as and when a lightpath arrives and the lightpath is terminated after certain amount of time. In case of both incremental and dynamic traffic, the main objective is to find the routing and wavelength information in such a way that it minimizes the blocking probability of the network. The number of connections established in the network is maximized if the blocking probability is minimized. This dynamic traffic problem is referred to as Dynamic Lightpath Establishment (DLE) problem. The RWA problem is a complex problem and hence there are many heuristics proposed to solve the routing and wavelength problem separately [47]. If the setup and teardown times of all the lightpath demands are known in advance the problem is called Scheduled Traffic Model [29]. There are two types of requests in this model, fixed window requests and flexible

window requests. If the start time and end time of a lightpath request cannot be changed it is called fixed window model and in case of flexible window model, we can change the start time and end time within the window interval. In [48], the authors propose an ILP model to solve these problems. In [45] the routing and wavelength assignment algorithm for various types for advance reservations are discussed and the algorithms are presented for requests having fixed specific start time and specific duration (STSD), specific start time and unspecified duration (STUD) and Unspecified Start Time and Specified Duration (UTSD).

2.5 Network Reoptimization

The network re-optimization enables the network to carry maximum traffic without any addition in its capacity. In case of dynamic connections, the route and wavelength information are found as and when the connection request arrive and hence this leads to network inefficiencies. The network inefficiencies is majorly due to the non availability of information about the future connection requests when we are selecting the route and wavelength for a given connection request. If we continuously select route and wavelength based on simple heuristic based algorithms this will lead to suboptimal schedules which results in under utilization of the network resources.

Network re-optimization increases the network utilization. Re-optimization of lightpath is discussed in various papers and most of them have proposed solutions for static traffic demands and there have been many heuristics proposed for long-term ondemand traffic flows [25]. In [47] a re-optimization scheme for advance reservation is proposed that does not change the starting and ending time of the reservations. In this, when a request for reservation is blocked then all the time overlapping reservations are removed and then they are rescheduled after re-ordering them according

to their start times. This sorting of reservation requests can result in provisioning of the earlier blocked request. If this re-ordering does not help then the reservations are restored in their original order.

All the papers on network re-optimization consider only a greedy based algorithm or an evolutionary algorithm to reoptimize the network resources. The greedy algorithm works fast but does not produce result as good as an evolutionary algorithm. The disadvantage of evolutionary algorithm is it runs slow than the greedy algorithm and hence it cannot be used in practical situations where a user waits for the response of a reoptimization mechanism. Hence in the next chapter we have run both the greedy and genetic algorithm in parallel in separate threads to get the best of both the worlds.

Chapter 3

Continuous and Parallel Reoptimization of Lightpath Scheduling

3.1 Introduction

In this chapter we study the problem of scheduling D-SBDs and we assume that the network is slotted. The duration of a scheduled lightpath is measured in number of time slots and each time slot is of equal length. We study the reservation problem with two types of algorithms: greedy algorithm and genetic algorithm. They are run in parallel in separate threads so that the greedy algorithm interacts with the user and the genetic algorithm performs the offline optimization of all the future reservation requests.

We ensure a deterministic response to the user i.e. after submitting an advance reservation request the user will immediately know if his/her request is granted but if it cannot be granted by the greedy algorithm he/she needs to wait till the end of

the current time slot to know whether his/her request is a confirmed reject or it can be accommodated due to the optimization done by the genetic algorithm. However once the request is granted the request cannot be rejected by the offline optimization process anytime in the future.

An example for a D-SBD can be a request for a bandwidth of 5 Gbps between 5:00 P.M and 9:00 P.M. where both starting and ending time are fixed. We also allow the user to specify a range of start times i.e. the scheduled bandwidth demand can start anytime between a min start time and max start time. For example a user can request a bandwidth of 2.5 Gbps between two end points on a weekly basis with min start time as 9.00 A.M Monday and max start time of 1.00 P.M. Monday for a duration of 5 hours.

3.2 Network Model

We consider WDM wavelength-routed mesh networks as our network model for the proposed bandwidth reservation problem. In this, optical fiber links interconnect a set of reconfigurable optical cross-connects (OXCs). There are two unidirectional fibers in each link and there are a fixed number of wavelengths over each fiber. Each wavelength is of some fixed bandwidth and hence it is possible to reserve some bandwidth of particular wavelength for a period of time. It is assumed that there is no wavelength conversion capability for the OXCs and hence there is wavelength continuity constraint for all the lightpaths. It is assumed that a bandwidth reservation cannot span more than one wavelength.

The following are our assumptions.

- We assume that the network time is slotted where a time slot represents the smallest unit of time in the network. Each time slot is of equal size and fixed

length. An bandwidth request can only be scheduled at the beginning of a time slot and the duration of the D-SBD is represented as a multiple of time slots.

- Each wavelength is assumed to be of some fixed bandwidth B.
- Each D-SBD occupies b units of bandwidth where $0 < b \leq B$. The i^{th} D-SBD in the network is denoted by $(s^i, d^i, b^i, t_x^i, t_y^i, \tau^i, l^i)$, where s^i is the source node, d^i is the destination node, t_x^i is the minimum start time, t_y^i is the maximum start time and τ^i is the duration (in time slots) for scheduled bandwidth request, b^i is the number of units of bandwidth, and l^i is the maximum path length in kilometers. For a time-fixed D-SBD, t_x^i and t_y^i are same and is a fixed value. For time-window D-SBD $t_x^i < t_y^i$.
- It is assumed that D-SBD requests that cannot be accommodated by both the greedy and genetic algorithms are considered as lost.

A bandwidth request in the network must be in one of the following three states:

- *not-scheduled*: In this state, the D-SBD has not been scheduled and its starting time, route and wavelength can be changed. The incoming D-SBDs will be in this state.
- *scheduled*: In this state, the D-SBD request has been granted to the user and its starting time cannot be changed but its routing and wavelength information can be changed.
- *in-service*: In this state, the D-SBD has been physically provisioned and its starting time, routing and wavelength assignment cannot be changed.

3.3 Problem Definition and Our Proposed Approach

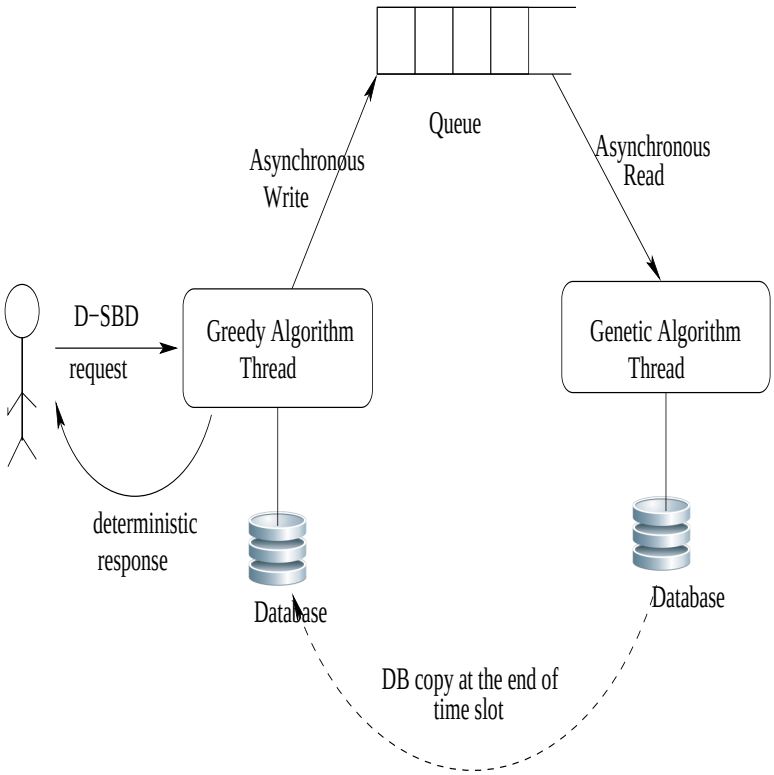

Figure 3.1: System Overview.

Given a D-SBD, we need to provide a deterministic response to the user quickly and at the same time we want to find the best possible schedule so that we can accommodate the maximum number of requests. A greedy algorithm runs fast and hence, can give a quick response to the user and the genetic algorithm which runs more slowly finds the best possible schedule for a given set of D-SBDs. In our approach both the greedy algorithm and the genetic algorithm are run in parallel in separate threads and hence this approach takes advantage of the nature of both to perform better as a system.

A user reserving bandwidth in advance from a source node to a destination node in the network interacts only with the greedy algorithm i.e. response for the bandwidth request will be given immediately by the greedy algorithm and he is unaware of the

optimization done by the genetic algorithm. Both the greedy algorithm and the genetic algorithm maintain independently a reservation database which stores the reservation details of all the D-SBDs. The i^{th} entry in the database can be expressed by a tuple of $(s^i, d^i, b^i, t^i, \tau^i, r^i, w^i)$ where s^i is the source node, d^i is the destination node, b^i is the bandwidth requested, t^i is the start time, τ^i is the duration, r^i is the path from the source to destination which is one of the k-shortest paths between them and w^i is the chosen wavelength across the entire path from source to destination. The size of the database is the number of D-SBDs that are in *scheduled* and *in-service* states. The t^i is the start time between the minimum start time and maximum start time specified by the user. This database maintains the reservation details for the entire topology.

As the time is slotted all the D-SBD requests arrive at the start of every time slot and are passed on to both the greedy and the genetic algorithm. The D-SBD requests can also be considered to arrive at any point in a time slot but in that case whether it will be included for re-optimization during the current or next time slot is an issue. Hence we did not consider this scenario. The greedy algorithm processing the request can give any of the following three types of response to the user:

- scheduled: The requested bandwidth has been allocated on some path from source to destination and it has been added to the greedy algorithm database.
- waiting: The requested bandwidth is not available in the greedy algorithm database but it could be allocated by the re-optimization done by the genetic algorithm.
- not-scheduled: This response can be given only at the end of any time slot after the database copy operation (which will be described next). This means that the requested bandwidth cannot be allocated and will not be considered by the

genetic algorithm anymore.

The genetic algorithm processing the request will try to find the best possible routing and wavelength assignment information for all the future D-SBDs so that the maximum number of D-SBDs can be scheduled. At the end of every time slot the greedy algorithm copies the database of the genetic algorithm and ensures the following two conditions: (a) the number of bandwidth demands that are in scheduled state is more than that in the database of the greedy algorithm and (b) all the bandwidth demands that are in scheduled state in greedy algorithm will remain in the same state even after the database copy. This database copy may change the route and wavelength information of the bandwidth demands that are already in scheduled state in the greedy algorithm database. It might also change some of the bandwidth demands that are in non-scheduled state to scheduled state. In this case the user will be notified that his request is scheduled and if the genetic algorithm also cannot schedule a particular D-SBD then the user will be notified of the rejection for his/her bandwidth request. This database copy will have no effect on the bandwidth demands that are in in-service state. Hence a user's bandwidth request which might have been rejected by the greedy algorithm could be changed to scheduled state at the end of a time slot and it will remain in scheduled state in all the future time slots. This process is continuous i.e. the genetic algorithm is interrupted only during the database copy and then it continues to run by taking in the D-SBDs that arrive in the current time slot.

The greedy algorithm writes into the database whenever a D-SBD request is changed to scheduled or in-service state at the start of a time slot and also during the database copy operation which occurs at the end of a time slot. The genetic algorithm writes into its database just before the database copy operation. It chooses the best genome in its current population and reserves the routing and wavelength

information as given in each of its genes (which represents a D-SBD request). It then triggers the database copy operation and continues its execution after it receives the incoming D-SBDs during the next time slot. As there are only two threads running in the system a binary semaphore is used to synchronize the database copy operation. There will not be any starvation in the system as both the threads have access to a common system clock and at each iteration both the algorithms check whether it is the start/end of a time slot and perform their operations accordingly. The database copy operation is the critical section of this algorithm. There will be no concurrent writes on a single database by both the algorithms as they both maintain their own independent databases. The output of the greedy algorithm is given to the genetic algorithm through an asynchronous mechanism and hence the threads are not interrupted by this operation. The greedy algorithm writes into a global queue its routing and wavelength information for all the D-SBDs arriving in the current time slot according to its algorithm. At each iteration the genetic algorithm polls the queue to check whether the greedy algorithm has finished its operation. If the information is available then the genetic algorithm adds the routing and wavelength information in the queue to its population and it will serve as the lower bound for the performance of the genetic algorithm.

3.4 Greedy Algorithm for Bandwidth Scheduling

In this section we describe a deterministic bandwidth scheduling algorithm based on a greedy algorithm for the problem mentioned in Section 3.3. Given a D-SBD, we need to find the routing and wavelength assignment and a starting time slot for the given bandwidth request. It is expected that the greedy algorithm finds a solution in a short amount of time.

Fixed-alternate routing is used as the path selection scheme. We compute k-shortest paths [31] between the source and destination denoted by $\{P_1, P_2, \ldots, P_k\}$ and use them as the set of candidate paths for the bandwidth request. It is assumed that they satisfy the path length constraint [36].

The greedy algorithm is based on the Phase I algorithm of [27] except that the algorithm described in this section also allows requests for a portion of bandwidth of any wavelength. We use *Slotted First-Fit* (*SFF*) wavelength assignment scheme. SFF selects the first common wavelength over the entire path from source to destination which has the requested bandwidth available for reservation. Bandwidth is said to be available only if it is available over all the time slots requested by the user.

Algorithm 1 describes the working of the greedy algorithm. The inputs to the algorithm are a topology graph $G(V, E)$ and a D-SBD represented by (s, d, b, t, τ, l), where the value of t is an integer in a range $[t_x, t_y]$. $t_x = t_y$ for a time-fixed D-SBD and $t_x < t_y$ for a time-window D-SBD. In Step 3, this algorithm loops through each of the starting time slots between t_x and t_y and in Step 4 it loops through each of the k-shortest paths from s to d. In Step 5 it uses the *SFF* to find the first wavelength containing the requested bandwidth available for the request. In Step 7 it inserts the solution to the solution list S and if the solution list is empty after the loop is finished then the requested bandwidth demand is marked as blocked (in Step 12). Otherwise, the requested bandwidth demand is provided a solution from the solution list which has the minimum objective value. The objective value used is MWL which is the minimum number of wavelength links in the path.

Algorithm 1 Greedy Bandwidth Scheduling Algorithm

Input: A D-SBD request (s, d, b, t, τ, l) and $G(V, E)$
Output: A schedule of the demand or reject notification

empty the solution list S
compute the k-shortest path set $\{P_1, P_2, \ldots, P_k\}$ between s and d with path length no greater than l
for $t = t_x$ to t_y **do**
  for $P \in \{P_1, P_2, \ldots, P_k\}$ **do**
    use SFF as wavelength assignment scheme to find an available wavelength w on the path P
    if a wavelength is found **then**
      insert (P, w, t) as a solution into the solution list S
    end if
  end for
end for
if the solution list S is empty **then**
  the demand is blocked
else
  compute the objective value for each solution in the list S and select the one with minimized objective value
end if

3.5 Genetic Algorithm for Bandwidth Scheduling

Genetic algorithm (GA) [20] is a class of evolutionary algorithms (EA) which uses techniques that are inspired by evolutionary biology such as mutation, selection, crossover and inheritance. GA is a search technique using probabilistic rules to find near-optimal solutions to the search and optimization problems. GA maintains a pool of possible solutions, called a genome (or chromosomes). In our approach each gene in the genome is a tuple of $(s^i, d^i, b^i, t^i, \tau^i, r^i, w^i)$ where s^i is the source node, d^i is the destination node, b^i is the bandwidth requested, t_i start time, τ^i is the duration, r^i is the path from the source to destination which is one of the k-shortest paths between them and w^i is the chosen wavelength across the entire path from source to destination. Thus a gene represents both the D-SBD information and the routing and wavelength assignment information for that particular D-SBD. Hence a genome

in our algorithm is a list of all future reservations that are in scheduled as well as not-scheduled state but not in in-service state.

We use the basic genetic algorithm. In this algorithm, first a set of genomes are created by the initialization algorithm and the population is evaluated by a fitness function to find the best, average and worst genome of the population. Until the termination criteria is not reached a set of genomes from the population is selected by roulette-wheel selection method [20] for the crossover and mutation. In roulette-wheel selection, the fitness function assigns a fitness to possible solutions and this fitness level is used to associate a probability of selection with each individual chromosome. Again the resulting population is passed on to the fitness function to find the best, average and worst genome. Our genetic algorithm continues its optimization until all the requested D-SBDs are in the in-service state.

The genetic algorithm runs in parallel as a separate thread to the greedy algorithm. At the start of each time slot it takes in the D-SBDs arriving at the current time slot and continues the optimization along with previous D-SBDs. It also removes the D-SBDs that will be changed to in-service state from the current time slot.

3.5.1 Initialization

This phase initializes all the genomes of the entire population. Each genome of the population represents some ordering of all the future D-SBDs along with the routing and wavelength information for each D-SBD. The ordering is randomized so that over the generations we expect to find an ordering which accommodates more number of D-SBDs than the rest. The ordering of D-SBDs and the routing and wavelength assignment according to the greedy algorithm mentioned in the Section 3.4, is taken as one of the genomes in the population. This ensures that the genetic algorithm performs at least as well as the greedy algorithm.

Algorithm 2 Initialization Algorithm

Let S represent the entire population and let the i^{th} genome be G^i where $1 \geq i \leq |S|$. Let g^k represent the k^{th} gene in the genome where $1 \geq k \leq |G^i|$.
Input: set of D-SBDs arriving at the current time slot
Output: initialized population with each gene in the genome containing $(s^i, d^i, b^i, t^i, \tau^i, r^i, w^i)$ information

$i \leftarrow 1$
while $i \leq |S|$ **do**
  for a random D-SBD that is not yet assigned from all the future DSLDs **do**
    $r^k \leftarrow$ choose a random $k - shortest$ path from source to destination
    $w^k \leftarrow$ choose a random wavelength from the available wavelength
    $t^k \leftarrow$ choose a random start time between t_{lower} and t_{upper} if the current D-SBD is in non-scheduled state.
    add g^k containing $(s^i, d^i, b^i, t^i, \tau^i, r^i, w^i)$ to G^i
  end for
  $i \leftarrow i + 1$
end while

3.5.2 Fitness Function

The fitness function used in our algorithm aims to minimize the blocking probability of the future D-SBDs. Blocking Probability (BP) can be defined as the ratio of number of blocked connections to the number of requested connections. Let δ be blocking probability and hence the fitness function is

$$\text{minimize}(\delta)$$

where $\delta = \frac{\text{number of blocked connections}}{\text{number of requested connections}}$

3.5.3 Mutation

The input to the mutation algorithm is the genome which is selected from the population using the roulette wheel selection method. In our algorithm we randomly pick two genes from the genome and interchange their positions in addition to changing

their route and wavelength information. If the picked D-SBD is in non-scheduled state then we also change its starting time. The starting time, route and wavelength information are assigned random values similar to the initialization algorithm. Mutation is the key process in the genetic algorithm which enables the genetic algorithm to find a better solution than the greedy algorithm and this avoids the genetic algorithm from falling into local minima.

3.5.4 Crossover

In the genetic algorithm, we pick two genomes (mom, dad) based on the roulette-wheel selection from the entire population. First we take the mom genome and find a random number less than the size of the genome. The size of the genome is the total number of future D-SBDs. We create the sister genome by copying the genes before the random point in the mom genome and copying the rest of the D-SBDs from the dad genome. In a similar manner brother genome is created by taking random point from dad genome. The mom and dad genomes are replaced by the sister and brother genomes if their fitness functions are better than that of the mom and dad genomes.

3.6 Experimental Results

In this section, we present the numerical results for our proposed approaches to solve the problem of dynamic bandwidth scheduling. We conduct our simulation experiments on two networks: a 24 node network topology with 43 bidirectional links (Fig. 3.2) and over the ESnet network which has 51 nodes (Fig. 3.3). Each link in the network has W wavelengths, where $W \in \{4, 8, 16, 32\}$. Each wavelength is assumed to be of bandwidth 10 Gbps. We perform experiments comparing our approach with the re-optimization at blocking method described in [27]. In each experiment we use

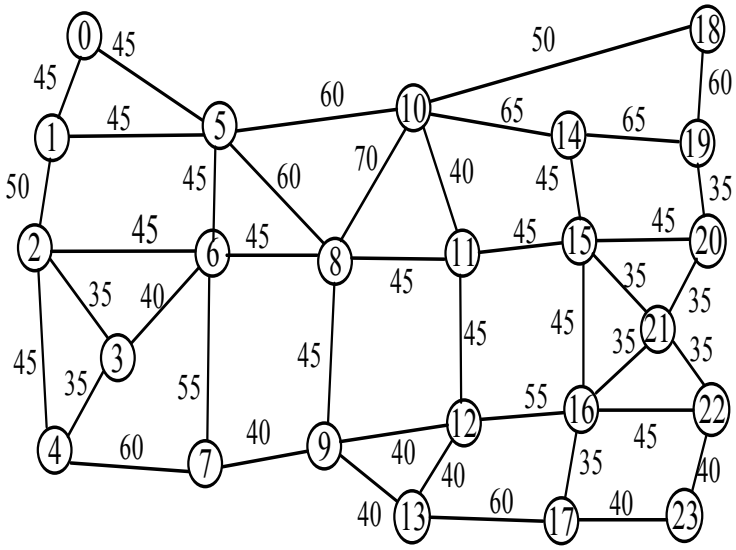

Figure 3.2: 24node Network Topology

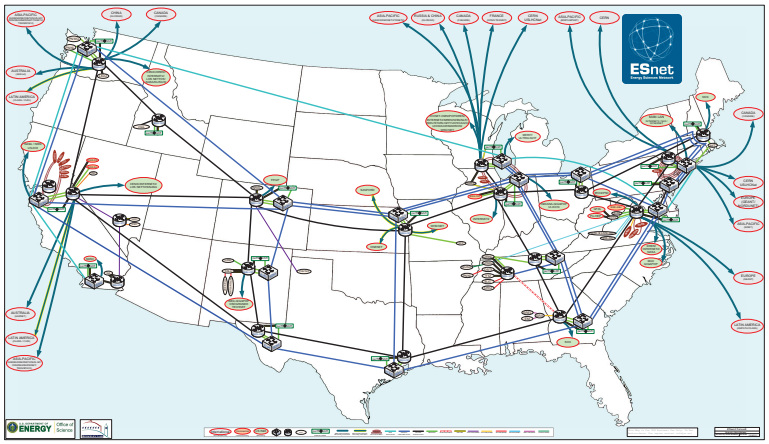

Figure 3.3: ESnet Network Topology (51 nodes)

blocking probability (bp) and service blocking probability (sbp) as the performance metrics. The service blocking probability is defined as the ratio of the sum of the durations of blocked D-SBDs to the sum of durations of all the D-SBDs.

Parameter	Value
Number of Generations	∞
Population Size	100
Mutation Percentage	50%
CrossOver Percentage	50%
Replacement Percentage	50%

Table 3.1: Genetic Algorithm Parameter List

Table 3.1 shows the parameters we used for the genetic algorithm. These parameters have been obtained after several experiments. The number of generations used and the percentage of mutation, crossover and replacement has direct control over the running time of the algorithm. Hence they are kept small enough to run within the specified time slot as well as optimal enough to find a better solution than the greedy algorithm. The replacement percentage represents the number of genomes replaced at every generation as we use elitism approach in our genetic algorithm. The number of generations is ∞ because the genetic algorithm runs continuously till there are no more D-SBDs arriving during the current time slot and all the D-SBDs in the database of genetic algorithm are in the in-service state.

3.6.1 Stochastic Simulation

We use a mixed ratio of time-fixed and time-window demands of 6:4 as we expect that there will be greater number of time-fixed bandwidth requests than the time-window demands. Each time slot is assumed to be of duration 15 minutes [27]. The time-window demands are uniformly distributed in the range of 5 to 50 and the inter-arrival mean is exponentially distributed with mean λ. It is also assumed that the

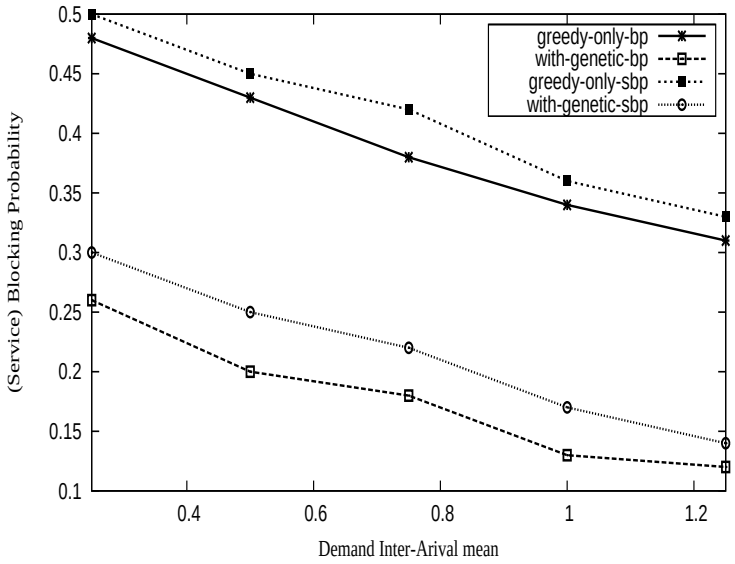

Figure 3.4: 24 node wavelength=16 bandwidth=5 Gbps

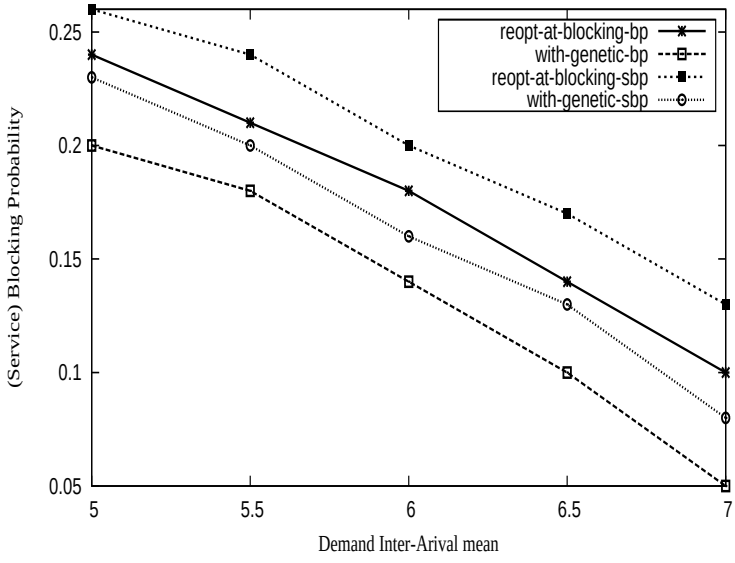

Figure 3.5: 24 node, wavelengths=4, bandwidth=10 Gbps

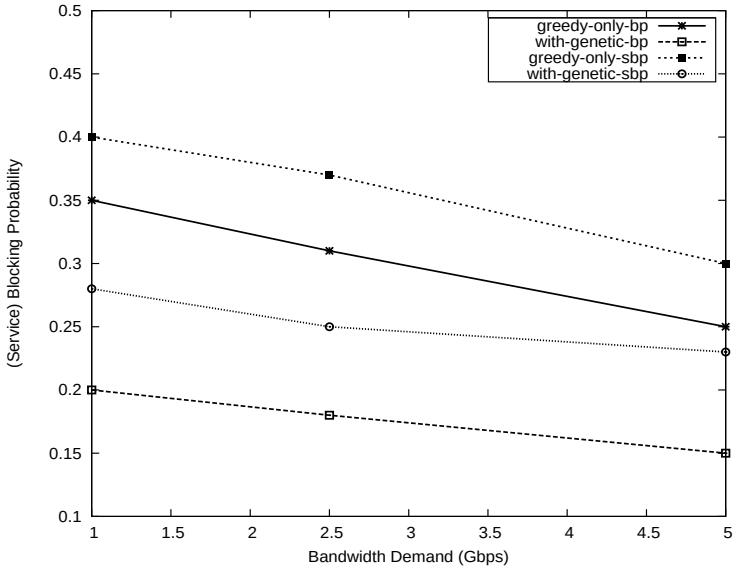

Figure 3.6: 24 node, wavelengths=8, mean inter arrival time=1min

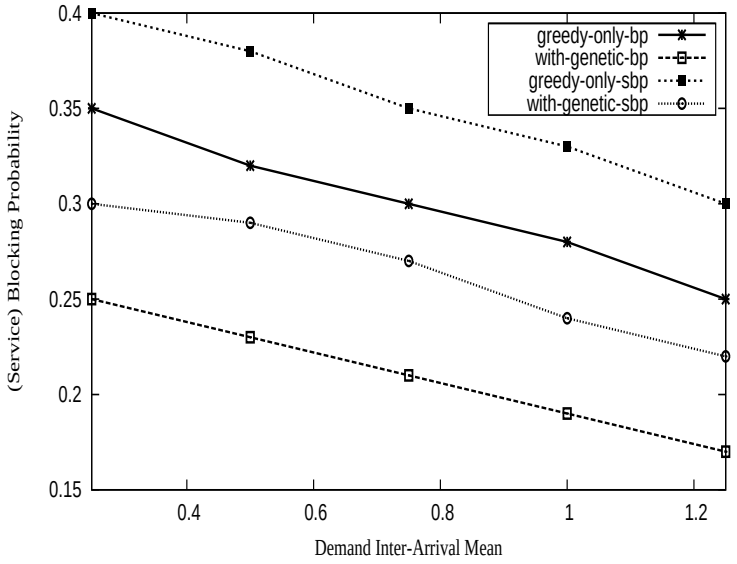

Figure 3.7: ESnet, wavelengths=32, bandwidth=2.5 Gbps

time duration between the arrival time of the D-SBD and the minimum starting time of the D-SBD is exponentially distributed with a mean of 80 time slots. As the time is slotted the duration of the D-SBDs is measured as a multiple of time slots and it is uniformly distributed in the range $[1, 50]$. It is also assumed that the duration of the D-SBD is uniformly distributed in the range of $[1, 15]$ for 50% demands, in the range of $[16, 25]$ for 25% of demands, in the range of $[26, 30]$ for 10% of demands, in the range of $[31, 40]$ for 10% of demands and in the range of $[41, 50]$ for the remaining 5% of demands. The traffic mix we use is similar to the one used in an earlier work [27]. We are using k value as 10 for $k-$shortest paths algorithm and the path length constraint is kept as 600 km for 24 node network topology. All the results have been computed with 95% confidence interval but the confidence intervals have not been shown in the figure so that it is readable. We observed that the running time for greedy algorithm was in seconds and the running time of one generation of genetic algorithm was also in seconds. As we run the genetic algorithm for one time slot it goes through several generations before we get the best possible solution.

Figures 3.4−3.7 show the performance of the stochastic simulation. In Fig. 3.4 and 3.5 we vary the inter arrival time keeping the number of wavelengths to be 16 and 4 and bandwidth as 5 Gbps and 10 Gbps respectively over the 24 node network topology. In Fig. 3.6 we vary the bandwidth keeping the number of wavelengths to be 8 over the 24 node network topology. In Fig. 3.7 we vary the mean inter-arrival time over the ESnet topology. It can be seen from the graphs that the genetic approach outperforms both the greedy and also the re-optimization at blocking.

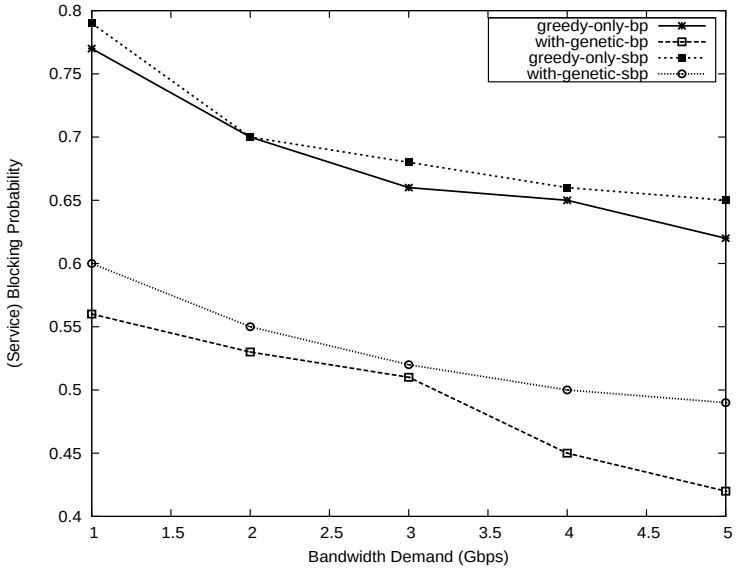

Figure 3.8: ESnet, wavelengths=8, trace-driven

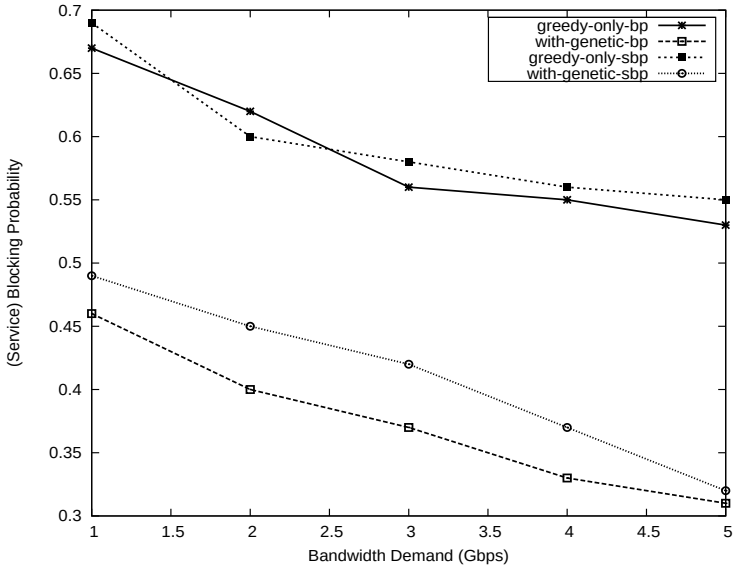

Figure 3.9: ESnet, wavelengths=16, trace-driven

3.6.2 Trace Driven Simulation

We compared our approach by taking the real network trace (Appendix B) of OSCARS database over ESnet topology. The parameters we extracted from the reservation database of OSCARS are (startTime, endTime, duration, startNode and endNode). All the reservations in the OSCARS database are of type time-fixed and hence the startime time cannot be changed for any of the bandwidth requests. In Figs. 3.8 and 3.9 we compare the blocking probability and service blocking probability of the greedy algorithm with the genetic algorithm over the ESnet topology. We vary the bandwidth demand from 1Gbps to 5Gbps. In Fig. 3.8 we set the number of wavelengths to be 8 and in Fig. 3.9 the wavelength value is assumed as 16. The number of bandwidth demands used in the experiment is 3000 which is the number of requests given to the OSCARS reservation system in ESnet domain over a period of 697 days. It can be seen that the genetic optimization significantly outperforms the greedy approach in both blocking probability and service blocking probability.

3.7 Conclusion

In this chapter, we studied the problem of dynamic scheduled bandwidth demands (D-SBDs) over WDM mesh networks. We run a greedy and genetic algorithm in parallel over two network topologies for offline network re-optimization. The performance results show that our approach yields a considerable improvement over existing re-optimization techniques. The reoptimization technique discussed in this chapter can be extended in several methods which are discussed in the next chapter. We also discuss how re-optimization could help manage circuits in realtime in a software like OSCARS.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

In this thesis, we studied the problem of dynamic scheduled bandwidth demands (D-SBDs) over WDM mesh networks. We proposed an approach in which we run a greedy algorithm and a genetic algorithm in parallel in separate threads. The user interacts only with a deterministic greedy bandwidth scheduling algorithm and he is unaware of the running of the genetic algorithm. The genetic algorithm performs the offline optimization of the arriving D-SBDs in the background. We studied the performance of our algorithm with and without genetic optimization and also compare our results with those of the re-optimization at blocking method. We used both the stochastic traffic as well as real network trace from OSCARS reservation system of ESnet. The performance results show that our algorithm outperforms the greedy as well as the re-optimization at blocking methods.

4.2 Future Work

The network reoptimization method discussed in previous chapter can be improved by considering reoptimization during protection scenarios. In case of protection, a path or a link is protected against failures by preassigning resources for a backup path, while in restoration schemes an alternate route is discovered dynamically for each failed connection after the failure occurs. Protection schemes have faster recovery time and provide guaranteed recovery but they require more network resources. Hence network reoptimization under protection mechanism is a significant challenge as the method has to take care of both the working link and protection link into consideration. If the path of the working link is changed then the path of the protection link might or might not change according to the reoptimization requirement.

Another important area of future work is to implement this in an operational software framework such as OSCARS (Appendix C). The intent of OSCARS is to create a service for dynamic QoS path establishment that is simple for users to use, and easy to administer. The only task required of a user is to make a bandwidth reservation. Reservation can be made either for immediate use or in advance for either one-time use or persistent use, e.g. for the same time everyday. All necessary mechanisms needed to provide the user with a guaranteed bandwidth path are coordinated by a Reservation Manager (RM) and managed by the routers in the network. A user submits a request to the RM (using either an API or an optional Web front-end) to schedule an end-to-end path (e.g. between an experiment and computing cluster) specifying start and end times, bandwidth requirements, the source host that will be used to provide an application access to the path, and the destination host. The status of a circuit in OSCARS is ACTIVE if it is accepted by the RM and currently provisioned in the routers, PENDING if it is accepted by RM but will be provisioned at a later time, CANCELLED if the reservation request is cancelled by the user,

FAILED if the reservation request is denied by the RM.

Re-optimizing the reserved circuits (PENDING) in OSCARS is a necessary feature to accomodate maximum number of circuits in the ESnet backbone network. Currently OSCARS does not have more number of circuits in PENDING state and hence combined reoptimizaiton of PENDING and ACTIVE circuits will certainly help to improve the current OSCARS software. Re-optimization in active circuits can be done by changing the path of the long ACTIVE circuits during their maintainance period. Hence a user who is reseving a long circuit should have provision in OSCARS to specify the maintainance time so that the reoptimizaiton can be done over that time. Another way to re-optimize the ACTIVE circuits can be to minimize the overall paths taken by the circuits so that we can accomodate more number of circuits with larger bandwidth requirements. The re-optimization feature can also be given as a privilige to a user where the system re-optimizes the circuits of only a particular user but the effectiveness of such a feature will certainly depend on other reserved future circuits. If there are more number of reserved future circuits that are distributed throught all the paths in the backbone network then reoptimizing the circuits of a particular user might not yield good result.

Appendix A

List of Acronyms

AAA	Authentication, Authorization and Audition Subsystem
AS	Autonomous System
ASTB	Application Specific Topology Builder
BSS	Bandwidth Scheduler SubSystem
BP	Blocking Probability
D-SBD	Dynamic Scheduled Bandwidth Demand
D-SLD	Dynamic Scheduled Lightpath Demand
DCN	Dynamic Circuit Network
DLE	Dynamic Lightpath Establishment
DRAGON	Dynamic Resource Allocation via GMPLS Optical Networks
DWDM	Dense Wavelength Division Multiplexing
ESnet	Energy Sciences Network
GA	Genetic Algorithm
GENI	Global Environment for Network Innovations
GMPLS	Generalized Multi-Protocol Label Switching
GpENI	Great Plains Environment for Network Innovation

HCC	Holland Computing Center
HOPI	Hybrid Optical and Packet Infrastructure
IDC	Inter Domain Controller
ION	Inter operable On-Demand Network
IP	Internet Protocol
LHC	Large Haldron Collider
LSP	Label Switched Path
MAX	Mid-Atlantic CrossRoads
MPLS	Multi-Protocol Label Switching
NARB	Network Aware Resource Broker
NLR	National Lambda Rail
OSCARS	On-Demand Secure Service and Advance Reservation System
OXC	Optical Cross-Connect
PE	Provider Edge
PSS	Path Scheduler SubSystem
QinQ	IEEE 802.1QinQ standard
RCE	Resource Computation Engine
RM	Resource Manager
RSVP	Resource Reservation Protocol
RWA	Routing and Wavelength Assignment
S-SLD	Static Scheduled Lightpath Demand
SBP	Service Blocking Probability
SFF	Slotted First Fit
SLD	Scheduled Lightpath Demand

SOAP	Simple Object Access Protocol
SONET	Synchronous Optical Network
STSD	Start Time and Specific Duration
STUD	Start Time and Unspecified Duration
TE	Traffic Engineering
UNL	University of Nebraska - Lincoln
UTSD	Unspecified Start Time and Specified Duration
VC	Virtual Circuit
VLAN	Virtual LAN
VLSR	Virtual Label Switched Router
WDM	Wavelength Division Multiplexing s

Appendix B

OSCARS Network Trace

B.1 OSCARS Database Schema

We use the mysql dump of OSCARS 0.5.4 to extract the required network trace. The following are the mysql tables of the bss module of OSCARS.

```
+---------------------------+
| Tables_in_doebss          |
+---------------------------+
| domainServices            |
| domains                   |
| edgeInfos                 |
| idSequence                |
| interdomainRoutes         |
| ipaddrs                   |
| l2SwitchingCapabilityData |
| layer2Data                |
| layer3Data                |
```

```
| links                     |
| mplsData                  |
| nodeAddresses             |
| nodes                     |
| pathElemParams            |
| pathElems                 |
| paths                     |
| ports                     |
| reservations              |
| resvPathReport            |
| routeElems                |
| tokens                    |
+---------------------------+
```

The topology information of the network is stored in the domains, nodes, links and ports tables. The reservations are stored in the reservations table and the path of each reservations is stored in the layer2Data and layer3Data. The id of the domains is used as a foreign key in the nodes, id of the nodes is used as a foreign key in the link and id of the links is used as a foreign key in the ports table. The following is the output of the describe command of the reservations table.

```
 mysql> desc reservations;
+--------------------+--------------------+------+-----+---------+
| Field              | Type               | Null | Key | Default |
+--------------------+--------------------+------+-----+---------+
| id                 | int(11)            | NO   | PRI | NULL    |
| startTime          | bigint(20) unsigned | NO   |     | NULL    |
```

```
| endTime             | bigint(20) unsigned | NO   |     | NULL    |
| createdTime         | bigint(20) unsigned | NO   |     | NULL    |
| bandwidth           | bigint(20) unsigned | NO   |     | NULL    |
| login               | text                | NO   |     | NULL    |
| payloadSender       | text                | YES  |     | NULL    |
| status              | text                | NO   |     | NULL    |
| localStatus         | tinyint(1)          | YES  |     | 0       |
| description         | text                | YES  |     | NULL    |
| statusMessage       | text                | YES  |     | NULL    |
| globalReservationId | varchar(63)         | YES  | UNI | NULL    |
+---------------------+---------------------+------+-----+---------+
12 rows in set (0.00 sec)
```

The reservations table does not store the source and destination details of the reservation requested by the user. The paths table stores the source and destination node details and it uses the id of the reservations table as its foreign key. The following is the output of the desc command of the paths table.

```
 mysql> desc paths;
+---------------+---------+------+-----+---------+----------------+
| Field         | Type    | Null | Key | Default | Extra          |
+---------------+---------+------+-----+---------+----------------+
| id            | int(11) | NO   | PRI | NULL    | auto_increment |
| reservationId | int(11) | NO   |     | NULL    |                |
| pathSetupMode | text    | YES  |     | NULL    |                |
| nextDomainId  | int(11) | YES  |     | NULL    |                |
| pathType      | text    | NO   |     | NULL    |                |
```

```
| priority      | int(11) | YES  |     | NULL    |                |
| grouping      | text    | YES  |     | NULL    |                |
| direction     | text    | YES  |     | NULL    |                |
+---------------+---------+------+-----+---------+----------------+
8 rows in set (0.00 sec)
```

The layer2Data table has the urn detail of the reservation and it uses id of the paths table as foreign key. In order to retrieve the reservations that belong only to esnet we need to do a join operation on the tables layer2Data, paths, reservations and list only the source, destination, startTime, endTime and bandwidth fields as a result. The following is the output of the desc command of the layer2Data table.

```
 mysql> desc layer2Data;
+--------------+---------+------+-----+---------+----------------+
| Field        | Type    | Null | Key | Default | Extra          |
+--------------+---------+------+-----+---------+----------------+
| id           | int(11) | NO   | PRI | NULL    | auto_increment |
| pathId       | int(11) | NO   | UNI | NULL    |                |
| srcEndpoint  | text    | NO   |     | NULL    |                |
| destEndpoint | text    | NO   |     | NULL    |                |
+--------------+---------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

B.2 Sample Network Trace

We consider only the traces of ESnet domain and we extract parameters (srcNode, destNode, startTime, destTime, bandwidth) from the database. The following is an example of a network trace.

src: urn:ogf:network:domain=es.net:node=fnal-mr1:port=
TenGigabitEthernet4/3:link=*
dest: urn:ogf:network:domain=es.net:node=chi-sl-mr1:port=
TenGigabitEthernet3/1:link=*
startTime: 1207156740
endTime: 1301764740
bandwidth: 2000000000

The source and destination are mentioned in the form of Uniform Resource Name (URN) in the database. The URN in OSCARS is of the form $[urn : ogf : network : domain = * : node = * : port = * : link =]$. The URN is the form agreed on by the DICE Control Plane working group and originates from work performed in the Open Grid Forum's (OGF) Network Markup Language Working Group (NML-WG). The URN is used in describing topologies and in performing path computation and reservation. The startTime, endTime are represented as seconds elapsed since 12am, Jan 1, 1970. The bandwidth is represented in bytes and hence the above value corresponds to value of 2 Gbps.

Appendix C

Modifying OSCARS code

After downloading the OSCARS software, the source code for all the packages is found under the folder $\$OSCARS\_HOME/dcn\text{-}software\text{-}suite$-0.5.3$/idc/src/net/es/oscars$. BSS is the folder that contains the *OSCARSCore.java* file which has the initialization methods for all the components of OSCARS. The *ReservationManager.java* is the file that contains methods such as submitCreate() and create() which are responsible for submitting createReservation job and handling createReservation job respectively. The pss is the folder that contains the source code for path setup modules. The *PathSetupManager.java* is the file that contains the method *create*() which is responsible for triggering the actual path setup code. The path setup options in OSCARS are terce, database and static. The terce will use DRAGON to compute the path, database option uses the mysql tables to check for the path and the static uses the file *static-routes.xml* file for calculation of path.

Hence in order to incorporate the re-optimization feature in OSCARS the trigger for reoptimization should be written in the *ReservationManager.java* file. The actual reoptimization function can be written in the pss folder and called from *PathSetupManager.java* file.

Bibliography

[1] http://www.internet2.edu/ion/.

[2] http://www.es.net/oscars/.

[3] http://dragon.maxgigapop.net/twiki/bin/view/DRAGON/WebHome.

[4] http://www.controlplane.net/.

[5] http://www.ietf.org/rfc/rfc3031.txt.

[6] http://www.faqs.org/rfcs/rfc2205.html.

[7] http://www.rfc-archive.org/getrfc.php?rfc=3471.

[8] https://wiki.internet2.edu/confluence/display/DCNSS/Home.

[9] http://www.geant2.net/.

[10] http://www.internet2.edu/ion/dynes.html.

[11] http://www.gpeni.net.

[12] http://www.ieee802.org/1/pages/802.1Q.html.

[13] https://wiki.ittc.ku.edu/gpeni/OnInterconnectingGpENIwithMAX:AReport.

[14] https://wiki.ittc.ku.edu/gpeni/CoreDirectorComponentManagerInterface

[15] https://wiki.ittc.ku.edu/gpeni/DCN/IONinGpENIInvestigationReport

[16] P. Angu and Byrav Ramamurthy. Experiences with dynamic circuit creation in a regional network testbed. In *IEEE INFOCOM High-Speed Networks (HSN) Workshop*, 2011.

[17] P. Angu, M. Subedee, B. Ramamurthy, X. Yang, and T. Lehman. Inter-domain dynamic circuit network creation (demo). In *IEEE Global Telecommunications Conference*, 2010.

[18] Lars-Olof Burchard, Hans-Ulrich heiss, and Ceaser A. F. De Rose. Performance issues of bandwidth reservation for grid computing. *Proceedings of 15th Symposium on Computer Architecture and High Performance Computing*, 2003.

[19] Chiara Curti, Tiziana Ferrari, Leon Gommans, S. van Oudenaarde, Elisabetta Ronchieri, Francesco Giacomini, and Cristina Vistoli. On advance reservation of heterogeneous network paths. *Future generation Computer Systems*, pages 525–538, April 2005.

[20] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, January 1989.

[21] Walter Goralski. *SONET: A guide to Synchronous Optical Network*. McGraw-Hill, 1997.

[22] Chin Guok, D. Robertson, M. Thompson, J. Lee, B. Tierney, and W. Johnston. Intra and Interdomain Circuit Provisioning Using the OSCARS Reservation System. In *Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on*, pages 1–8, October 2006.

[23] Eric He, Xi Wang, and Jason Leigh.

[24] J. Kuri and et al. Routing and wavelength assignment of scheduled lightpath demands. *IEEE Journal of Selected Areas in Communications (JSAC)*, 21:1231–1240, October 2003.

[25] T. Lehman, J. Sobieski, , and B. Jabbari. Dragon: a framework for service provisioning in heterogeneous grid networks. *IEEE Communications Magazine*, 44:84–90, March 2006.

[26] F. Leung, J. Flidr, C. Tracy, X. Yang, T. Lehman, B. Jabbari, D. Riley, and J. Sobieski. The dragon project and application specific topologies. In *Proceedings, BROADNETS 2006*, pages 1–10, 2006.

[27] L. Shen, X. Yang, A. Todimala, and B. Ramamurthy. A two-phase approach for dynamic lightpath scheduling in wdm optical networks. In *ICC '07. IEEE International Conference on Communications*, pages 2412–2417, 2007.

[28] James P.G. Sterbenz, Deep Medhi, Byrav Ramamurthy, Caterina Scoglio, David Hutchison, Bernhard Plattner, Tricha Anjali, Andrew Scott, Cort Buffington, Gregory E. Monaco, Don Gruenbacher, Rick McMullen, Justin P. Rohrer, John Sherrell, Pragatheeswaran Angu, Ramkumar Cherukuri, Haiyang Qian, and Nidhi Tare. The great plains environment for network innovation (gpeni): A programmable testbed for future internet architecture research. In *Proceedings of the 6th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, pages 1–14, 2010.

[29] W. Su and G. Sasaki. Scheduling of periodic transfers with flexibility. *Annual Allerton Conference on Communication, Control, and Computing*, October 2003.

[30] Bin Wang, Tianjian li, Xubin Luo, Yuqi Fan, and Chunsheng Xin. On service provisioning under a scheduled traffic model in reconfigurable wdm optical

networks. *2nd International Conference on Broadband Networks*, pages 13–22, October 2005.

[31] J.Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712 – 716, 1971.

[32] D. Awduche and Y. Rekhter, "Multiprotocol lambda switching: combining MPLS traffic engineering control with optical crossconnects," *IEEE Communications Magazine,* pp. 111-116, March 2001.

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

[34] B. Mukherjee, "WDM Optical communication networks: Progress and challenges," *IEEE Journal on Selected Areas in Communications,* vol. 18, no. 10, pp. 1810–1824, October 2000.

[35] B. Mukherjee, Optical WDM Networks, *Springer,* ISBN 0-387-29055-9, 2006.

[36] B. Mukherjee, D. Banerjee, S. Ramamurthy and A. Mukherjee, "Some principles for designing a wide-area WDM optical network," *IEEE/ACM Transactions on Networking,* vol. 4, no. 5, pp. 684-696, October 1996.

[37] B. Ramamurthy, Design of Optical WDM Networks: LAN, MAN and WAN Architectures, *Kluwer Academic Publishers,* ISBN 0-7923-7281-6, December 2000.

[38] R. Ramaswami and K. Sivarajan, Optical Networks: A Practical Perspective, *Elsevier Science & Technology Books,* ISBN 1558606556, 2001.

[39] Internet2, "The Hybrid Optical and Packet Infrastructure Project," *Available on WWW* http://hopi.internet2.edu.

[40] NLR, "Projects Supported by National LambdaRail Infrastructure," *Available on WWW* http://www.nlr.net/supported.html.

[41] The UltraLight Collaboration, "UltraLight Annual Report for 2004-2005," *Available on WWW:* http://ultralight.caltech.edu/web-site/ultralight/documents/2005/02annualreport/

[42] J. Towns et al., "SC03 TeraGrid Tutorial: Applications in the TeraGrid Environment," *Available on WWW* http://www.teragrid.org/userinfo/docs/SC03-TeraGrid-tutorial.ppt.

[43] Lars-Olof Burchard, Networks with advance reservations: Applications, architecture, and performance. In *Journal of Network and Systems Management, 13(4)*,December 2005.

[44] I. Foster, C.Kesselman, C. Lee, R. Lindell, K. Nahrstedt, and A. Roy. A distributed resource management architecture that supports advance reservations and co-allocation. In *7th International Workshop on Quality of Service (IWQoS)*, pages 2736, 1999.

[45] H. Zang, J. Jue, and B. Mukherjee, "A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks," *Optical Networks Magazine,* vol. 1, no. 1, pp 10-15 January 2000.

[46] R. Krishnaswamy and K. Sivarajan, "Algorithms for routing and wavelength assignment based on solutions of LP-Relaxations," *IEEE Communications Letter,* vol. 5, no. 10, pp 435-437 October 2001.

[47] X. Yang, L. Shen, A. Todimala and B. Ramamurthy, "An Efficient Scheduling Scheme for On-Demand Lightpath Reservations in Reconfigurable WDM Optical," *Optical Fiber Communication (OFC),* pp 1-3, March 2006

[48] A.Jaekel, Lightpath scheduling and allocation under a flexible scheduled traffic model. *pp 1-5, GLOBECOMM 2006.*

[49] R. Guerin and A. Orda, "Networks with advance reservations: The routing perspective," *IEEE INFOCOM '00,* pp. 118 - 127, Tel-Aviv, Israel, March 2000.