DYNAMIC ROUTING AND WAVELENGTH ASSIGNMENT IN WAVELENGTH-DIVISION MULTIPLEXED (WDM) OPTICAL NETWORKS USING NEURO-DYNAMIC PROGRAMMING

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Serkan Yeşildağ

July 2001

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Murat Alanyalı(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Nail Akar

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Ezhan Karaşan

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

DYNAMIC ROUTING AND WAVELENGTH ASSIGNMENT IN WAVELENGTH-DIVISION MULTIPLEXED (WDM) OPTICAL NETWORKS USING NEURO-DYNAMIC PROGRAMMING

Serkan Yeşildağ
M.S. in Electrical and Electronics Engineering
Supervisor: Assist. Prof. Dr. Murat Alanyalı
July 2001

In this thesis work, dynamic routing and wavelength assignment (RWA) problem in optical networks is studied. Assuming memoryless interarrival and holding times for calls, minimizing the average call blocking rate can be viewed as a Markov Decision Problem. Dynamic programming is the direct method to obtain an exact solution. However, this method is intractable for large networks encountered in practice. Therefore, we use neuro-dynamic programming (NDP) which is a simulation based dynamic programming methodology to obtain successful policies. In this approach the cost-to-go function is approximated using predetermined features of the network state, so the obtained policies are based on these features. In the present context, features are selected from the most commonly used heuristics for the RWA problem. Simulation results shows that NDP approach gives significantly lower blocking rates compared to the heuristics.

Keywords: Wavelength-Division Multiplexing (WDM), Wavelength Routing, Wavelength assignment, Optical Networks, Dynamic Programming, Neuro-Dynamic Programing.

ÖZET

OPTİK DALGABOYU BÖLÜNMELİ ÇOĞULLAMA AĞLARINDA SİNİRSEL DİNAMİK PROGRAMLAMA KULLANILARAK DİNAMİK YOL VE DALGABOYU ATAMA

Serkan Yeşildağ
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Yrd. Doç. Dr. Murat Alanyalı
Temmuz 2001

Bu tezde, optik ağlarda yol ve dalgaboyu atama problemi ele alınmaktadır. Çağrılar için hafızasız varış arası zamanları ve üssel dağılımlı sürme zamanları kabul edildiğinde, ortalama reddedilme sıklığının en aza indirilmesi Markov karar verme problemi olarak görülebilir. Dinamik programlama kesin sonucu elde etmede direkt yöntemdir. Fakat bu yöntemin pratikte karşılaşılan büyük ağlar için çözümü zordur. Bu yüzden, başarılı stratejiler elde etmek için benzetime dayalı dinamik programlama yöntemi olan sinirsel dinamik programlama kullanmaktayız. Bu yaklaşımda gidiş-ücreti fonksiyonu ağ durumunun önceden belirlenmiş özellikleri kullanılarak yaklaştırıldığı için elde edilen stratejiler bu özelliklere dayalıdır. Bu durumda özellikler yol ve dalgaboyu atama problemi için sıkça kullanılan buluşsal yöntemlerden seçilmiştir.

Anahtar Kelimeler: Dalgaboyu Bölünmeli Çoğullama, Dalgaboyu Yönlendirme, Dalgaboyu Atama, Optik Ağlar, Dinamik Programlama, Sinirsel Dinamik Programlama.

ACKNOWLEDGMENTS

I would like to express my deep gratitude to my supervisor Assist. Prof. Dr. Murat Alanyalı for his guidance, suggestions and invaluable encouragement throughout the development of this thesis.

I would like to thank Assist. Prof. Nail Akar and Assist. Prof. Ezhan Karaşan for reading and commenting this thesis.

Contents

List of Figures

List of Tables

To My Parents . . .

Chapter 1

INTRODUCTION

The amount of information exchanged in communication systems increases rapidly each day. Current advances in optical communication seem to solve this problem by offering bandwidth of several gigabits per second at a very low error rate ($\sim 10^{-9}$) [1]. Beside the high capacity, optical communication also offers transparency which allows different signal formats to share the same medium. Consequently, optical communication seems to be an essential technology for wide-area communication systems.

Wavelength-Division Multiplexing (WDM) is an effective way to increase the transport bandwidth using several wavelengths in an optical fiber. WDM is a promising technique for information transport in all-optical networks. In these networks, the main idea to obtain high speeds is to maintain the signal in optical form, therefore avoiding the optical to electronic conversion and vice versa. This is because of the fact that signals can be modulated electronically at a maximum bit rate in the order of 10 gigabits per second while the optical fiber bandwidth is about 10 terahertz [2].

From the point of view of the optical layer, traffic demand is comprised of call requests which should be assigned a proper route and wavelength between

the associated pairs of network nodes. Many traffic models have been proposed to describe the traffic demands. Each of the traffic models has its pros and cons, and none of them exactly provides a realistic model [3]. In near-term scenario, call requests are likely to be retained for a fairly long time of days or months. In this static model, only provision and protection of fixed size optical paths can be obtained. For the near future, dynamic optical networking seems to be a promising way which enables the flexibility and efficiency of the electronic domain while maintaining the scalability property of the optical domain.

In this thesis work, the routing and wavelength assignment (RWA) problem in WDM optical networks is considered under a dynamic traffic model. When a new call request arrives between any source and destination node pair, a suitable route and wavelength pair should be assigned such that no two connections using the same wavelength share the same fibre. If such an assignment is not possible at the time of arrival, then incoming call request is blocked and lost. The aim of this thesis work is to minimize the long term average blocking probability using neuro-dynamic programming (NDP).

Under standard assumptions like memoryless interarrival and holding times, RWA problem can be considered as a Markov Decision Process (MDP). Therefore, minimizing the average number of calls blocked per unit time is formulated as a stochastic dynamic programming problem. However, for large problems Bellman's *curse of dimensionality* (the exponential computational explosion with the problem dimension) does not allow for an exact solution. For this reason, until now, only heuristic algorithms were presented in the literature.

NDP is a simulation-based approximate dynamic programming methodology to produce near optimal solutions for large scale dynamic optimization problems. The main idea is to construct an approximate cost-to-go function by using some features extracted from the current state of the network, and optimize it by tuning the parameters associated with these features [4]. In this thesis, features

are selected from the most commonly used heuristic algorithms for the RWA problem in WDM optical networks and simulation-based methods are employed to tune the parameters.

Approximations of the optimal costs-to-go have been used in the past in a variety of dynamic programming cases. One particular success was the development of a backgammon playing program by Tesauro [5] which motivated subsequent research. Some other interesting examples are: Dynamic Channel Allocation in Cellular Telephone Systems [6], Improving Elevator Performance [7], and Call Admission Control and Routing in Integrated Services Networks [8].

This thesis is organized as follows: in Chapter 2, RWA problem in WDM optical networks is presented. Next, Dynamic Programming Formulation of the RWA problem is given in Chapter 3. Neuro-Dynamic Programming formulation of the RWA problem is given in Chapter 4. In Chapter 5, numerical results are presented and discussed, and finally the thesis is summarized in Chapter 6.

The main contribution of this thesis is that by using neuro-dynamic programming method, together with the features of the commonly used heuristics for the RWA problem in WDM optical networks, it is possible to obtain smaller average blocking probabilities than that of these heuristic algorithms. NDP approach can utilize the features for the rich class of revenue maximization problems as well. It is not obvious how to come up with good heuristics for revenue maximization problem in WDM optical networks and NDP may possibly offer a flexible way to extend the heuristics to these problems. This direction is currently under consideration.

Chapter 2

RWA IN WDM OPTICAL NETWORKS

In this chapter, background information related to the RWA problem in WDM optical networks is presented. WDM in optical networks rapidly gains acceptance to handle the ever-increasing bandwidth demands of the future network users. In WDM optical networks, users communicate with each other via all-optical WDM channels where each channel is assigned a different wavelength. These channels are called lightpaths and may traverse multiple fiber links. Figure 2.1 shows an example WDM network composed of three nodes with optical fiber links, where each link has two fibers and each fiber contains three wavelengths denoted by (w_1, w_2, w_3). Given a set of connections, RWA problem consists of setting up lightpaths by routing and assigning a wavelength to each connection such that no two connections using the same wavelength share the same fiber.

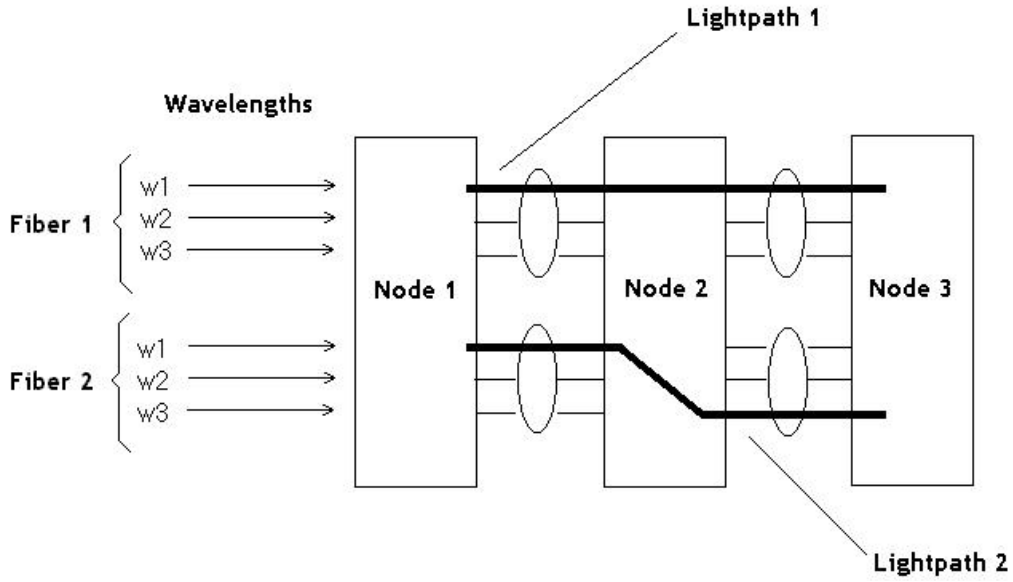

Figure 2.1: Example optical WDM network with three nodes.

2.1 Optical Switches

In all-optical networks signal remains in the optical domain in order to avoid the electrooptic conversion bottleneck. One proposed all-optical network is broadcast-and-select network [3] which is designed for a local area network (LAN) with small number of users. Here, all the nodes are fully connected to each other, thus the signal transmitted from each node is received from all nodes.

For a wide area network (WAN), boadcast-and-select architecture is not practical due to lack of wavelength reuse and budget limitations. This problem is solved by the introduction of optical switches. Optical switches, which are also known as wavelength routers, route optical signals based on the input port and wavelength of the optical signals. If all the nodes in a network have a wavelength routing capability then that network is called wavelength routing network.

Wavelength conversion devices are needed to connect lightpaths using different wavelengths. Wavelength converter is used to convert the optical signal with one wavelength into another wavelength before forwarding it on the next link. This capability is known as wavelength conversion.

Network is said to be wavelength-selective (WS), if there is no wavelength conversion capability at each node of a wavelength routing network. In this case a lightpath must occupy the same wavelength on all fiber links through which it traverses which is known as wavelength-continuity constraint.

If any wavelength can be converted to any other wavelength in every node of a wavelength routing network, then the network is said to have full wavelength-conversion capability and referred to as Wavelength Interchanging (WI) network. A WI network is similar to circuit-switched telephone network. Therefore, only the routing problem should be solved and the wavelength assignment problem is not an issue. Noticing that a single lightpath in a WI optical network can possibly use a different wavelength at each link along its path, wavelength conversion improves the efficiency of the total network by resolving wavelength conflicts of lightpaths.

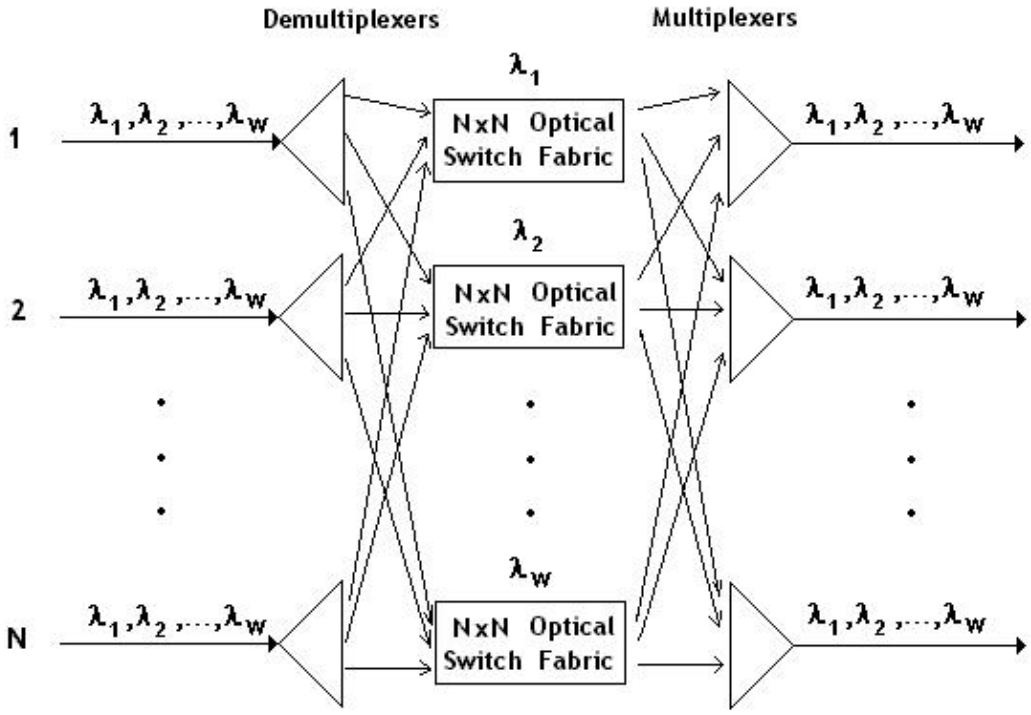

Figure 2.2: An $N \times N$ wavelength-selective cross-connect (WSXC).

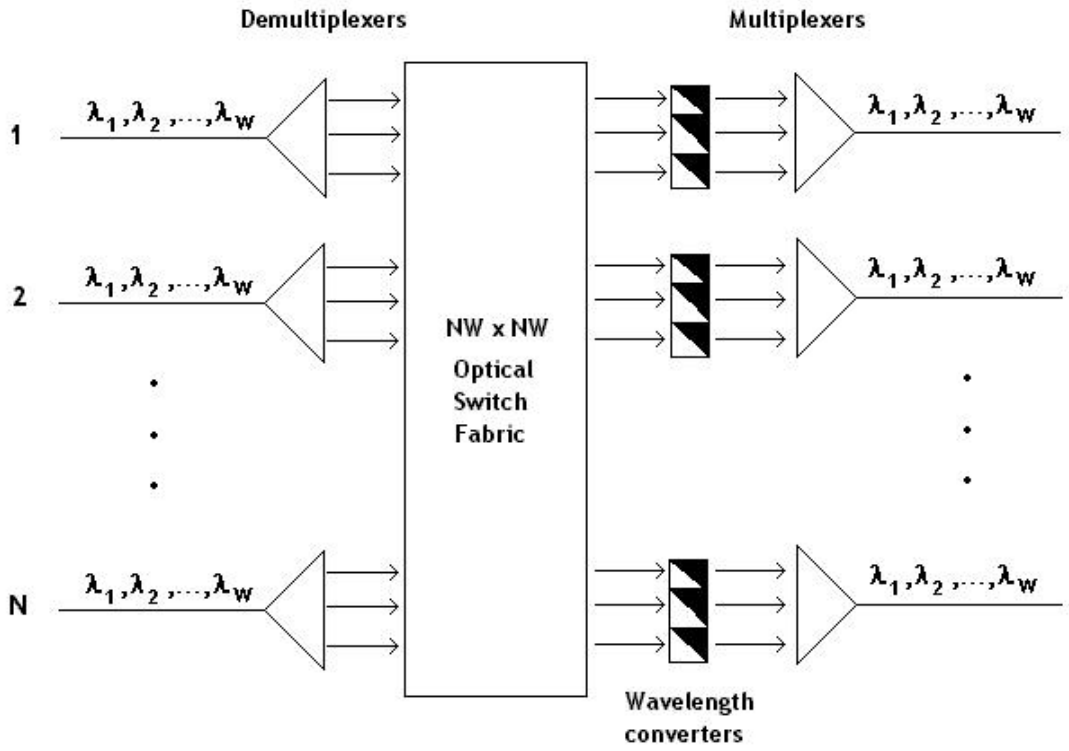

Figure 2.3: An $N \times N$ wavelength-interchanging cross-connect (WIXC).

There are two technologies possible to realize optical switches. These are wavelength-selective cross-connects (WSXC), and wavelength-interchanging cross-connects (WIXC) which allows wavelength conversion.

WSXC architecture is seen in Figure 2.2. It consists of N demultiplexers, N multiplexers and W $N \times N$ optical switch fabrics where N represents the nodal degree and W represents the number of available wavelengths for each fiber. It is apparent that a set of input ports with different wavelengths can be switched to an output port while the input ports with the same wavelengths can not be multiplexed to an output port due to wavelength conflict.

WIXC architecture is presented in Figure 2.3. It is seen that wavelength conversion significantly complicates the design of an optical cross-connect. Additional to the NW wavelength converters, this architecture uses a large single $NW \times NW$ optical switch fabric. Wavelength converters are usually placed at the output as in Figure 2.3 due to the fact that fixed-input, variable-output

wavelength converters are more difficult to implement than variable-input, fixed-output wavelength converters.

2.2 Traffic Models

In RWA problem, traffic model of connection requests typically may be of three types. These can be listed as follows:

- Static Traffic Model
- Incremental Traffic Model
- Dynamic Traffic Model

In the static model, the entire set of connections is known in advance and the problem is to minimize network resources like the number of wavelengths or the number of fibers in the network by setting up the lightpaths for these connections in a global fashion. Alternatively, the problem can be formulated as to increase the number of connections for a fixed number of given wavelengths. The RWA problem for static traffic model is referred to as Static Lightpath Establishment (SLE) problem in the literature. The SLE problem can be formulated as a mixed-integer linear program [9] that is difficult to solve. The SLE problem is shown equivalent in complexity to n-graph colorability problem, hence it is NP-complete [10]. To make the problem more tractable, the SLE might be divided into two subproblems as routing and wavelength assignment and then each subproblem can be solved separately [11].

In incremental traffic model, connection requests arrive sequentially, a lightpath is established for each connection and the lightpath remains in the network indefinitely.

In dynamic traffic model, lightpaths arrive over time, a lightpath is established for each connection and the lightpath is relased after a finite amount time.

The objective in both the incremental and dynamic traffic model is to set up lightpaths and assign wavelengths in such a way that minimizes the amount of connection blocking or that maximizes the number of connections that are established in the network. This problem is called as Dynamic Lightpath Establishment (DLE) problem. The DLE problem is more difficult to solve than SLE problem and therefore, generally heuristic algorithms are used. Heuristic algorithms exist both for the routing and wavelength assignment subproblems.

The main issue arises in the dynamic traffic model where the number of wavelengths required to obtain a non-blocking network is much higher when compared to static and incremental traffic models. It has been shown analytically that this is in fact the case for ring networks [12].

2.3 Routing

In this section various approaches to routing problem is presented.

2.3.1 Fixed Routing

In this approach, the same fixed route is chosen when a lightpath request arrives to a given source and destination node pair. One popular way is to route the connection through the shortest path where the path lengths are generally taken as the number of fiber link segments (hops). In this method, shortest path route is calculated for each source destination node pair beforehand by using well known methods like Dijkstra's algorithm or Bellman-Ford algorithm. When

a connection request arrives to a specified node pair it is routed through this pre-determined route.

This approach is easy to use but if the resources along the path are tied up it could result in high blocking probabilities in dynamic case or a large number of used wavelengths in the static case. Moreover, fixed routing is unable to handle the fault conditions like the failure of one or more links in the network.

2.3.2 Fixed-Alternate Routing

This approach considers multiple routes where each node contains a predetermined ordered list of routes to each destination node. As predetermined routes it is popular to use shortest-path route, second-shortest-path route, third-shortest-path route, etc. First route in the list is called as primary route and any alternate route should not share any link (link-disjoint) with the primary route. It is common practice to sort the routes in routing tables according to their number of hops, which means to select the shortest-path route as a primary route.

When a connection request arrives to any source destination pair, the source node tries to route the connection on any of the routes in the routing table sequentially. The connection is blocked if a route with available wavelength can not be found.

Although it is not as easy to use as Fixed Routing, this approach still provides a simple way to control the establishment of lightpaths. Additionally, it provides a fault tolerance to some degree when a link failure occurs in the network. Moreover, it was shown in [13] that this approach has the ability to reduce the blocking probability significantly compared to Fixed Routing.

2.3.3 Adaptive Routing

In adaptive routing, the main idea is to route the incoming connection request dynamically depending on the current state of the network. Currently active connections, together with their RWA, determine the state of the network.

There are several adaptive routing forms. One of them is least-congested-path routing (LCP) [14]. In this approach, like in the alternate routing case, for each source destination node pair a set of routes is pre-selected beforehand. When a connection request arrives to a specific node pair, the least congested route among the pre-determined routes is selected. The congestion on a path is determined by the least congested link in the path and the congestion on a link is measured by the number of wavelengths available on the link.

Since there are a number of pre-determined routes, LCP routing is resilient to link failures to some extent. Additionally, it has been shown that LCP routing performs much better than fixed-alternate routing [15]. However, since all links on all predetermined routes between a specified node pair have to be examined to determine the least congested route, LCP routing is computationally complex in nature.

2.4 Wavelength Assignment

In static traffic model, it is assumed that the routing problem of the given set of ligthpaths has already been solved. In this case the problem is to assign a wavelength to each lightpath in a manner that minimizes the number of wavelengths used under the wavelength-continuity constraint. Wavelengths should be assigned to each ligthpath such that no two ligthpaths share the same wavelength on a given fiber. Most common way to solve this problem is to formulate it as a graph-coloring problem [16]. Although this problem has been shown to

be NP-complete and hard to solve there are efficient graph coloring algorithms that finds the minimum number of colors that should be used [17].

For incremental and dynamic traffic models where the lightpath requests arrive one at a time, wavelength assignment heuristic algorithms are used. This is mainly due to the fact that the number of wavelengths required to obtain a non-blocking network is much higher especially in the dynamic traffic model [12]. In this dynamic wavelength assignment problem, the aim is to minimize the average blocking probability. Instead of minimizing the number of wavelengths used, here it is assumed that the number of wavelengths is fixed.

In the following subsection commonly used heuristic algorithms for the dynamic traffic model are described in detail.

2.4.1 Heuristic Algorithms

Since it is hard to find an optimal solution to the dynamic wavelength assignment problem, only heuristic algorithms have been proposed in the literature upto now. Most common heuristic algorithms can be listed as follows:

1) Random Wavelength Assignment (R) [18, 19]

2) First-Fit (FF) [10, 20]

3) Most-Used (MU) [20, 21]

4) Min-Sum (MS) [22]

5) Least-Loaded (LL) [17, 22]

6) Max-Sum (MΣ) [21, 23]

These algorithms are all on-line algorithms and can be used with different routing schemes. Let $\mathcal{C} = \{1, 2, \ldots, C\}$ represents the set of connections between all

possible source and destination node pairs, $\mathcal{P}_i = \{1, 2, \ldots, P_i\}$ represents the set of paths for connection i, $\mathcal{S}_p = \{1, 2, \ldots, S_p\}$ represents the set of available wavelengths along path p, M_l represents the number of fibers on link l and A_{lj} represents the number of used wavelengths, say j, on link l.

Random Wavelength Assignment (R)

This algorithm first finds the set of available wavelengths on the required route. Then the algorithm randomly selects a wavelength among the available wavelengths (usually with uniform probability).

First-Fit (FF)

This scheme first assigns an index to all wavelengths. When a new connection request arrives, lower-indexed wavelength is considered before the higher-indexed wavelength. Then the first available wavelength is selected. The main idea is to pack all of the used wavelengths to the lower end of the wavelength set so that the higher-indexed wavelengths have higher probability of being available. This scheme outperforms the random wavelength assignment heuristic and has a lower computational cost when compared to random wavelength assignment since it does not need to find the set of available wavelengths beforehand.

Most-Used (MU)

Like in the random case, the algorithm first finds the set of available wavelengths on the required route and then tries to select the most used wavelength in the network among all available wavelengths. Although MU slightly outperforms FF [17, 22], it introduces additional communication overhead. In other words,

global information about the network is needed to implement the MU heuristic algorithm.

Min-Sum (MS)

Optical network is called a single-fiber network if all the links in the network contain only one fiber. Otherwise, the optical network is called a multi-fiber network. This scheme is proposed for multi-fiber networks and reduces to the First-Fit heuristic algorithm for single-fiber network. This algorithm selects the wavelength that has the minimum average utilization. The route P and minimum indexed wavelength j are selected for the connection i that achieves

$$\min_{P \in \mathcal{P}_i, j \in \mathcal{S}_P} \sum_{l \in P} \frac{A_{lj}}{M_l} \ .$$

Least-Loaded (LL)

Like the Min-Sum algorithm, this heuristic is also designed for multi-fiber networks. The main idea is to select the wavelength that has the largest residual capacity on the most-loaded link along route P. LL selects the route P and minimum indexed wavelength j that achieves

$$\max_{P \in \mathcal{P}_i, j \in \mathcal{S}_P} \min_{l \in P} [M_l - A_{lj}] \ .$$

In single-fiber network the residual capacity is 1 or 0, therefore in this case LL reduces to FF algorithm. Simulation results shows that LL outperforms the MU and FF heuristics in terms of blocking probability [17, 22].

Max-Sum (MΣ)

The main idea is to maximize the remaining path capacities after ligthpaths are established. MΣ assumes that the set of routes is pre-determined for each

connection. Let x be the current state of the network (routes and wavelength assignments) and $\acute{x}$ be the next state of the network if route P and wavelength j is assigned to the connection. Then MΣ assigns the route P and wavelength j for the connection i that maximizes the quantity

$$\sum_{i \in \mathcal{C}, P \in \mathcal{P}_i, j \in \mathcal{S}_P} \min_{l \in P} [M_l - A_{lj}]$$

in the next state ($\acute{x}$) of the network. This algorithm can be applied to both single-fiber and multi-fiber networks.

Chapter 3

DYNAMIC PROGRAMMING FORMULATION

In this chapter, the problem of minimizing the average blocking rate in WDM optical networks is formulated as a continuous time, average cost dynamic programming problem. Let us consider a communication network where $\mathcal{N} = \{1, 2, \ldots, N\}$ represents the set of nodes, $\mathcal{W} = \{1, 2, \ldots, W\}$ represents the set of wavelengths in an optical fiber, $\mathcal{L} = \{1, 2, \ldots, L\}$ represents the set of unidirectional links.

At time t the state of the network is represented as x_t and consists of a set of active calls whose routing and wavelength assignment problem has been resolved. This finite set of all possible states will be referred to as the state space $\mathcal{S}$. Although call requests arrive in time in continuous manner, it is sufficient to consider the state of the network at discrete instants when certain events take place. These events might be either a new call request or the termination of an existing call. Let us represent this finite event space as Ω.

If the system is in state x and an event e takes place, then a proper decision u should be made. Let $U(x, e)$ denote the set of possible decisions. If e corresponds

to a new call request then $U(x, e)$ consists of assigning a possible route and wavelength pair to this call or simply rejecting the incoming call. If e corresponds to a call termination, then there is no need to make a decision since $U(x, e)$ consists of only terminating the specified call. Assume that the current state of the network is x, an event e occurs and a proper decision $u \in U(x, e)$ is made. Then the whole system moves to a next state which will be denoted as $\acute{x}$.

Let us denote the immediate cost that occurs when rejecting the incoming call request as q. Then the resulting cost will be shown as $g(x, e, u)$ such that: if e corresponds to a new call and u corresponds to rejecting that call then $g(x, e, u) = q$, otherwise if u corresponds to assigning a proper route and wavelength pair then $g(x, e, u) = 0$. If e corresponds to a call termination then $g(x, e, u) = 0$.

Given a current state of the network x, the set of decisions like rejecting the call or which route and wavelength pair should be assigned to a specified call will be referred as policy μ. Policy μ is simply a mapping which satisfies

$$\mu(x, e) \ \in \ U(x, e)$$

and whose domain is $\mathcal{S} \times \Omega$.

Assuming memoryless interarrival and holding times for calls and a fixed policy μ, x_t evolves as a continuous time, finite state Markov process. Let us represent the k-th event as e_k, the time of the k-th event as t_k, the state of the network just prior to time t_k as x_{t_k} and the decision made at time t_k as $u_{t_k} = \mu(x_{t_k}, e_k)$. Then the average blocking cost over the infinite time horizon associated with the policy μ can be given as follows:

$$v(\mu) = \lim_{N \to \infty} \frac{1}{t_N} \sum_{k=0}^{N-1} g(x_{t_k}, e_k, u_{t_k}). \tag{3.1}$$

The average blocking cost might be interpreted as the average cost per unit time for the system in steady state. Under the assumption of finite average call holding times, the whole system is modeled as an ergodic Markov process. Reaching any

state j from any state i is possible. Therefore the average blocking cost of state j is the same as that of state i since the costs incurred in the process of reaching state j from state i do not contribute to average blocking cost as $N \to \infty$. Therefore, $v(\mu)$ is independent of the initial state. Additionally, in ergodic processes it is known that time averages converges to ensemble averages. Therefore, average blocking cost converges to a deterministic constant with probability 1.

Let us define the differential cost-to-go of state x_{t_k} under policy μ as:

$$h^{\mu}(x_{t_k}) = \sum_{m=k}^{\infty} [g(x_{t_m}, e_m, u_{t_m}) - (t_{m+1} - t_m)v(\mu)], \tag{3.2}$$

where $u_{t_m} = \mu(x_{t_m}, e_m)$. It might be interpreted as the expectation of the difference of total blocking costs under policy μ over the infinite time horizon for a system initialized at state x_{t_k} compared to the system in steady-state.

A policy is said to be optimal if the average blocking cost of all other policies is greater than or equal to the average blocking cost of that policy. If the optimal policy is denoted as μ^*, then the optimal policy is defined as:

$$\mu^*(x, e) = arg \min_{u \in U(x,e)} [g(x, e, u) + h^*(\acute{x})] \tag{3.3}$$

where $h^*(x)$ represents the differential cost-to-go of state x associated with optimal policy μ^*. In order to find the optimal policy, one should know the optimal differential cost-to-go of all possible states.

Optimal differential cost-to-go $h^*(x)$ values for each possible state x of the network can theoretically be computed as follows [4, 8]:

$$v^* E\{\tau \mid x\} + h^*(x) = E\{\min_{u \in U(x,e)} [g(x, e, u) + h^*(\acute{x})]\}, \; x \in \mathcal{S} \tag{3.4}$$

where $v^* = v(\mu^*)$ represents the average blocking cost associated with the optimal policy and τ represents the time until the next event occurs. $E\{\tau \mid x\}$ represents the expectation of the time required until the next event occurs and $\acute{x}$ represents the next state of the network. If the number of all possible states is denoted as

$| \mathcal{S} |$, then Equation 3.4 is a system of $| \mathcal{S} | +1$ unknowns and $| \mathcal{S} |$ nonlinear equations for each possible state. The unknowns are $| \mathcal{S} |$ optimal differential cost-to-go values for each possible state and the value v^*. Therefore, one more equation is needed to solve Equation 3.4.

If the vector consisting of optimal differential cost-to-go values $h^*(x)$ for all possible states x is denoted as H^*, then it is seen that if H^* solves the Equation 3.4 then $H^* + re$ also solves that equation where r is a constant and e represents a vector of ones with length $| \mathcal{S} |$. In other words, what is important is the difference values $h^*(x) - h^*(y)$ for all possible states x and y. Therefore, if the empty system is taken as a reference state and denoted as $\hat{x}$, one can assume without loss of generalization that

$$h^*(\hat{x}) = 0. \tag{3.5}$$

Under this assumption Equation 3.4 together with the Equation 3.5 is known as Bellman equations and constitute a system with $| \mathcal{S} | +1$ unknowns and $| \mathcal{S} | +1$ nonlinear equations which is solvable.

Policy iteration and value iteration are two well known approaches to solve a dynamic programming problem [4]. We comment briefly on the policy iteration, in which, one starts with an arbitrary policy μ_0 and generates a sequence of new policies $\mu_1, \mu_2, \ldots$. Given a policy μ_k, first the $v(\mu_k)$ value is computed according to Equation 3.1 and $h^{\mu_k}(x)$ values for all possible states x are computed according to Equation 3.2. This process is also known as policy evaluation step. Then, the policy improvement step is performed to find the next policy μ_{k+1} according to the following formula:

$$\mu_{k+1}(x, e) = arg \min_{u \in U(x,e)} [g(x, e, u) + h^{\mu_k}(\acute{x})], \ x \in \mathcal{S}. \tag{3.6}$$

It is shown in [4, Proposition 2.4] that policy iteration algorithm generates an improving sequences of policies which terminates with the optimal policy μ^*.

Even for networks consisting of a few nodes and links, computation and storage of the optimal differential cost-to-go $h^*(x)$ values for every possible state x of the network by using Bellman equations might be impractical. For this reason, in the next chapter the neuro-dynamic programming approach, which is an approximate policy iteration method to find near optimal policies, will be presented.

Chapter 4

NEURO-DYNAMIC PROGRAMMING FORMULATION

In this chapter, the theoretical ground of the neuro-dynamic programming solution to RWA problem in WDM optical networks is presented. Neuro-dynamic programming method is an approximate dynamic programming method based on simulations which produces near-optimal solutions to large scale dynamic programming problems.

Neuro-dynamic programming starts with an arbitrary policy μ_k and approximates the cost-to-go function of this policy $h^{\mu_k}(\cdot)$ with an approximate cost-to-go function $\tilde{h}(f(x), \theta_{\mu_k})$, where $f(x)$ represents a vector of features extracted from the network and θ_{μ_k} represents a vector of tunable parameters associated with each used feature. Then, policy iteration on $\tilde{h}(f(x), \theta_{\mu_k})$ is performed to obtain better policy μ_{k+1}. This process is known as approximate policy iteration and can be summarized as follows:

1. Start with a fixed policy μ_k.

2. Approximate the values $h^{\mu_k}(x)$ and $v(\mu_k)$, which are hard to find, with an approximate function $\tilde{h}(f(x), \theta_{\mu_k})$ and a scalar quantity $\tilde{v}(\mu_k)$ respectively by using simulations.

3. Use policy iteration on approximate values $\tilde{h}(f(x), \theta_{\mu_k})$ and $\tilde{v}(\mu_k)$ to obtain a better policy μ_{k+1}.

4. Repeat this process by using μ_{k+1} instead of μ_k until obtained blocking probabilities do not change much with each iteration.

Approximations of optimal cost-to-go functions have been commonly used in the past [5–8]. Here, we are interested in the problems with a large number of states, and approximate cost-to-go functions $\tilde{h}(\cdot, \theta)$ that can be described with relatively few numbers (θ of small dimension). The main idea in these problems is to use state evaluators to rank different states and make a decision that results in the state with maximum reward or minimum cost. The state evaluator calculates a numerical value for each state using a heuristic formula which includes weights for the various features of the state. In other words, state evaluator calculates the approximate cost-to-go function $\tilde{h}(\cdot, \theta)$, where the weights of the features correspond to the parameter vector θ.

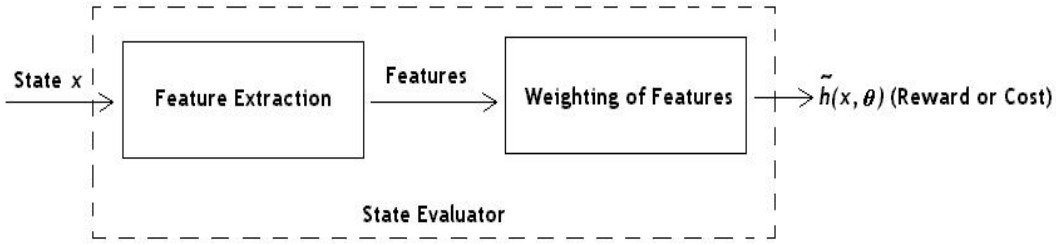

Figure 4.1: Structure of a state evaluator.

In Neuro-dynamic programming there are three steps to be considered

- Determining the general form of the approximate function $\tilde{h}(\cdot, \theta)$.
- Deciding on which features should be used in the approximate function $\tilde{h}(\cdot, \theta)$.

- Selecting a proper method to tune the parameter vector θ and scalar quantity $\tilde{v}$.

which will be presented in the following sections.

4.1 Approximation Architecture

Selection of an architecture means the choice of a parametric class of functions $\tilde{h}(\cdot, \theta)$ that suits best to the considered problem and is an important issue in function approximation.

Approximation architectures can broadly be classified into two main groups as linear and nonlinear ones. A linear architecture is of the form

$$\tilde{h}(x, \theta) = \sum_{k=0}^{K} \theta(k) f_k(x) \tag{4.1}$$

where $\theta(k),\ k = 0, 1, \ldots, K$, are the elements of a real parameter vector θ, and $f_k : \mathcal{S} \to \Re$ are known functions extracted from the network state. These functions are preferred to be easily computable for the simplicity of the architecture.

Assume that some training data pairs $(x, h^{\mu}(x))$ obtained by simulations under a policy μ are wished to fit using linear architecture. This issue can be formulated as a least squares problem where the aim is to minimize the squared error

$$\sum_{x} \Big[\sum_{k} \theta(k) f_k(x) - h^{\mu}(x) \Big]^2 \tag{4.2}$$

over all parameter vectors θ. Equation 4.2 is a linear least squares problem even if the functions $f_k(x)$ are nonlinear and can be solved using linear algebra techniques.

In nonlinear architecture, the dependence of $\tilde{h}(x,\theta)$ on θ is nonlinear and the least squares problem of minimizing the squared error

$$\sum_{x} \left[\tilde{h}(x,\theta) - h^{\mu}(x)\right]^2 \tag{4.3}$$

can not be reduced to linear algebraic problem. Therefore, Equation 4.3 should be solved by means of nonlinear programming methods. Multilayer Perceptron [4, 24] method is most commonly used nonlinear approximation architecture in the literature and has the power of approximating arbitrary functions of feature vector $f(x)$.

In this thesis, linear architecture is used as an approximation architecture due to two main reasons. First, the linear dependence of the approximation architecture to the parameter vector θ enables us to use fast and well-tested linear algebra algorithms. Second reason is to determine the features that are most relevant to the decision making process since the parameters associated with the most relevant features in decision process will dominate that of irrelevant features in magnitude.

4.2 Features

A feature is simply a mapping $f_k : \mathcal{S} \to \Re$ where $\Re$ represents the set of real numbers. Once the set of features $f_1, \ldots, f_K$ are determined, the feature vector is formed $f(x) = (f_1(x), \ldots, f_K(x))$. Some forms of the approximation architecture $\tilde{h}(x,\theta)$ can be seen in Figure 4.2. The function $h(x)$ that should be approximated is often a highly complicated nonlinear map as in the case of RWA problem in WDM optical networks. Therefore, it is sensible to break this complexity into smaller and less complex pieces by feature extraction. These features are usually handcrafted based on available insight and prior experience on the problem.

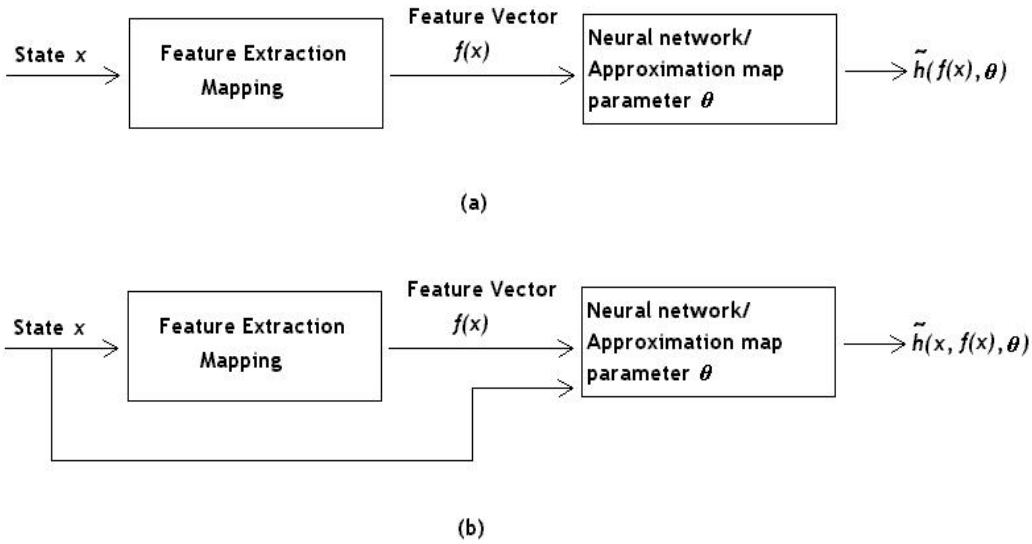

Figure 4.2: (a) A feature-based approximation architecture. (b) An approximation architecture that uses both raw encoding of state as well as feature vector $f(x)$.

In simulation part of the thesis, feature-based architecture of Figure 4.2(a) is employed where the features of the most commonly used heuristic algorithms are used. We next introduce the selected features which were used in neuro-dynamic programming in simulations. The notation used next is inherited from Section 2.4.1.

1. Global Availability (GA) : As a feature of each wavelength j, the following value:

$$f(j) = \sum_{i \in C, P \in \mathcal{P}_i} \min_{l \in P} [M_l - A_{lj}], \ j \in \mathcal{W}$$

is calculated. This feature is used by the $M\Sigma$ heuristic algorithm. Let us define the residual capacity of a wavelength j for the given path P as the maximum number of lightpaths that can be assigned path P and wavelength j. Then GA feature gives the total number of residual capacity of any wavelength throughout the whole network.

2. Local Availability (LA) : As a feature of wavelength j which is used in connection i and path P, the following value:

$$f(i,P,j) = \min_{l \in P}[M_l - A_{lj}],\ i \in \mathcal{C}, P \in \mathcal{P}_i, j \in \mathcal{W}$$

is calculated. This feature is used by both LL and $M\Sigma$ heuristic algorithms and gives the number of residual capacity of a wavelength for the given path of a connection. It is obvious that Global Availability features can easily be obtained from Local Availability features.

3. Global Utilization (GU) : As a feature of each wavelength j, the following value:

$$f(j) = \sum_{i \in C, P \in \mathcal{P}_i, l \in P} A_{lj},\ j \in \mathcal{W}$$

is calculated. This feature is used by the Most-Used(MU) heuristic algorithm and gives the total number of fibers in which a specified wavelength is used throughout the whole network.

4. Sum (Σ) : As a feature of wavelength j which is used in connection i and path P, the following value:

$$f(i,P,j) = \sum_{l \in P} \frac{A_{lj}}{M_l},\ i \in \mathcal{C}, P \in \mathcal{P}_i, j \in \mathcal{W}$$

is calculated. This feature is used in Min-Sum heuristic algorithms and gives an intuition about the relative utilization of a wavelength over a given path of a connection.

5. Normal-Sum (NΣ) : As a feature of wavelength j which is used in connection i and path P, the following value:

$$f(i,P,j) = \sum_{l \in P} \frac{A_{lj}}{\max_{w \in \mathcal{W}} A_{lw}},\ i \in \mathcal{C}, P \in \mathcal{P}_i, j \in \mathcal{W}$$

is calculated [25].

6. Sharing (S): As a feature of wavelength j which is used in connection i and path P, the following value:

$$f(i,P,j) = \sum_{l \in P} m_{lj},\ i \in \mathcal{C}, P \in \mathcal{P}_i, j \in \mathcal{W}$$

is calculated [25]. If it is assumed that n_l represents the number of most used wavelengths on link l, then

$$m_{lj} = \begin{cases} \frac{1}{n_l} & \text{if } A_{lj} = \max_{w \in \mathcal{W}} A_{lw} \\ 0 & \text{otherwise.} \end{cases}$$

It is seen that most of the features are connection and path-based. This choice of features is intuitively preferable due to the wavelength continuity constraint in WS optical networks. It is shown that link-based features give good results for call admission control and routing problem in integrated services network when used with NDP approach [8]. Link-based features entail smaller computational complexity. One such feature set adopted from the heuristics of [17] is currently under consideration in the present setting.

4.3 Training Method

There are several on-line and off-line methods to train the parameter vector θ [4]. Our main goal is to train the approximation architecture so as to identify θ, $\tilde{v}$ in such a way that $\tilde{h}(\cdot,\theta)$ and $\tilde{v}$ are good estimates of h^μ and $v(\mu)$ respectively. In this section, Sutton's [26] TD(0) (temporal differences) method which is a commonly used method in neuro-dynamic programming applications will be presented. This method is used in all simulations presented here.

TD(0) method has originally been proposed for discrete time, discounted cost problems. Since RWA in WDM optical networks can be considered as a continuous-time, average-cost problem, average cost TD(0) method [27] is used

which is a modified version of a discounted cost TD(0) method [28]. It is shown that average cost TD(0) method [27] has the same convergence properties with discounted cost TD(0) method [29].

One way to approximate $v(\mu)$ in Equation 3.1 is to sum a large number of immediate costs observed in a long trajectory of the system and then normalize by dividing with the total elapsed time. However, one should wait till the last event happens in order to find the average blocking cost $v(\mu)$ associated with policy μ. Another way is to use Robbins-Monro stochastic approximation algorithm which can be defined as follows:

$$\tilde{v}_k = \tilde{v}_{k-1} + \eta_k[g(x_{t_{k-1}}, e_{k-1}, u_{t_{k-1}}) - (t_k - t_{k-1})\tilde{v}_{k-1}], \tag{4.4}$$

where η_k is a diminishing step size parameter with increasing k. This is an online algorithm and each $\tilde{v}_k$ value is updated after the k'th event e_k. It has been shown in [27, Theorem 1] that $\tilde{v}_k$ converges to $v(\mu)$ as $k \to \infty$.

Temporal difference d_k computed after k'th event is defined as follows:

$$d_k = \underbrace{[g(x_{t_k}, e_k, u_{t_k}) - (t_{k+1} - t_k)\tilde{v}_k + \tilde{h}(x_{t_{k+1}}, \theta_k)]}_{Estimated\ cost-to-go\ based\ on\ simulation} - \underbrace{\tilde{h}(x_{t_k}, \theta_k)}_{Current\ estimate} . \tag{4.5}$$

Temporal difference d_k represents the difference between an estimate of the cost-to-go based on the simulated outcome and the current estimate $\tilde{h}(x_{t_k}, \theta_k)$. Therefore, the temporal difference provides an indication as to whether the current estimate should be raised or lowered.

In order to minimize the squared error in Equation 4.3, incremental gradient flow method is used which can be found in standard texts such as [4, Section 3.2.4]. By using this method the parameter vector θ is updated by

$$\theta_{k+1} = \theta_k - \gamma_k \sum_{m=0}^{\infty} \Big[\nabla_\theta[\tilde{h}(x_{t_m}, \theta_m) - h^\mu(x_{t_m})]\Big][\tilde{h}(x_{t_m}, \theta_m) - h^\mu(x_{t_m})], \tag{4.6}$$

where γ_k is a diminishing step size parameter with increasing k and ∇_θ represents the gradient with respect to parameter vector θ. When the definition of $h^\mu(x)$ in

Equation 3.2 is substituted into Equation 4.6, we get

$$\theta_{k+1} = \theta_k - \gamma_k \sum_{m=0}^{\infty} \nabla_\theta \tilde{h}(x_{t_m}, \theta_m) \Big[\tilde{h}(x_{t_m}, \theta_m) - \sum_{k=m}^{\infty} [g(x_{t_k}, e_k, u_{t_k}) - (t_{k+1} - t_k) v(\mu)] \Big]. \tag{4.7}$$

Since it is known that $\tilde{v}_k$ converges to $v(\mu)$ as $k \to \infty$, Equation 4.7 can be rewritten in terms of temporal differences as

$$\theta_{k+1} = \theta_k + \gamma_k \sum_{m=0}^{\infty} \nabla_\theta \tilde{h}(x_{t_m}, \theta_m) \sum_{k=m}^{\infty} d_k. \tag{4.8}$$

A more general update rule, known as TD(λ), where $0 \leq \lambda \leq 1$, uses weighted sum of temporal differences and is defined as

$$\theta_{k+1} = \theta_k + \gamma_k \sum_{m=0}^{\infty} \nabla_\theta \tilde{h}(x_{t_m}, \theta_m) \sum_{k=m}^{\infty} \lambda^{k-m} d_k. \tag{4.9}$$

To see the intuition behind Equation 4.9, one can refer to [4, Section 6.3]. It is seen that Equation 4.8 is a special form of 4.9 when λ is set to 1. Therefore Equation 4.8 is known as TD(1) method. Equation 4.9 is an off-line version of TD(λ) since parameter vector θ should be updated after all the trajectory $i_0, i_1, i_2, \ldots$ is simulated. However, in the on-line version, parameter vector θ is updated as soon as temporal difference d_k becomes available. More specifically, following the state transition $(i_{t_k}, i_{t_{k+1}})$ parameter vector θ is updated as follows:

$$\theta_{k+1} = \theta_k + \gamma_k d_k \sum_{m=0}^{k} \lambda^{k-m} \nabla_\theta \tilde{h}(x_{t_m}, \theta_m). \tag{4.10}$$

On-line TD(0) method is obtained by setting $\lambda = 0$ in Equation 4.10 and is given by the following Equation:

$$\theta_{k+1} = \theta_k + \gamma_k d_k \nabla_\theta \tilde{h}(x_{t_k}, \theta_k). \tag{4.11}$$

It is obvious that TD(0) does not evaluate previous gradients of $\nabla_\theta \tilde{h}(x_{t_m}, \theta_m)$ for $0 \leq m \leq k$ in order to update θ_k as in Equation 4.10. Therefore on-line TD(0) update equation is very suitable for real-time implementations. This was the main reason that TD(0) method is used in simulations. It should be noted that when the linear approximation architecture is used, $\nabla_\theta \tilde{h}(x, \theta) = f(x)$.

Under a fixed policy μ, let us assume that θ_k values are updated according to Equation 4.11 (on-line TD(0) method) and $\tilde{v_k}$ values are updated according to Equation 4.4. Additionally, assume that γ_k and η_k are diminishing step size parameters such that:

a) γ_k is positive, deterministic constant for $\forall\ k$ and satisfies $\sum_{k=0}^{\infty} \gamma_k = \infty$ and $\sum_{k=0}^{\infty} \gamma_k^2 < \infty$.

b) There exists a positive scalar n such that the sequence η_k satisfies $\eta_k = n\gamma_k$ for $\forall\ k$.

Then, it is proven in [27, Theorem 1] that $\tilde{v}_k$ converges to $v(\mu)$ and θ_k converges to a limiting vector θ such that the squared error between $\tilde{h}(\cdot, \theta)$ and $h^{\mu}(\cdot)$ is minimized with respect to θ under the given approximation architecture.

The method used in all simulation runs can be briefly explained as follows. First we start with an arbitrary policy μ and apply TD(0) until the parameter vector and average cost value converges. Then the resulting limiting value of parameter vector θ is used to define a new policy by means of policy iteration. This process is repeated until it is observed that obtained blocking probabilities do not change much with each iteration. This approach is known as approximate policy iteration and has some weak theoretical guarantees that the policy iteration algorithm generates an improving sequence of policies [4, Proposition 2.4].

This policy iteration process might be interpreted as an actor-critic system. In this interpretation, the critic is responsible for the policy evaluation step and evaluates the performance of the current policy, in other words it calculates the estimate of h^{μ_k} by tuning the parameters. On the other hand, actor is responsible for the policy improvement step who takes into account the latest evaluation of the critic, h^{μ_k}, to obtain the next policy μ_{k+1} (see Figure 4.3).

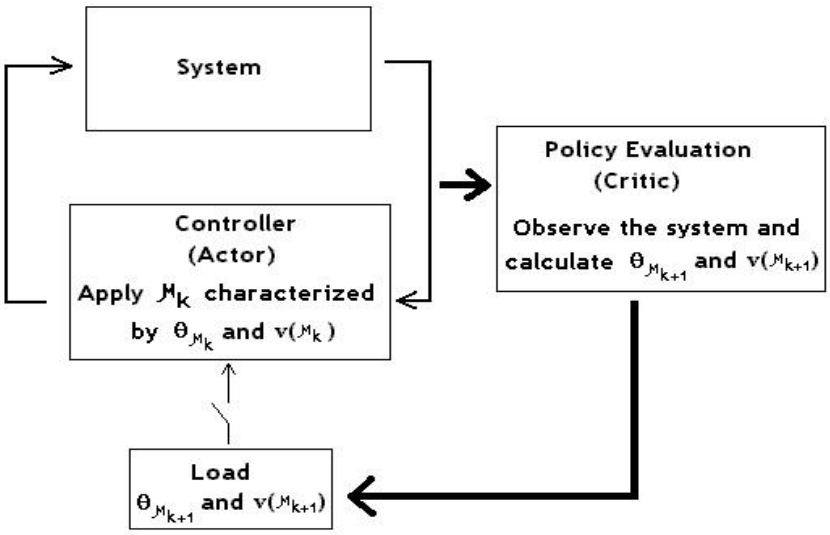

Figure 4.3: Policy iteration block diagram.

There is an alternative to the standard version of policy iteration in which policy update is performed after each update by the policy evaluation algorithm without waiting for the critic's computations to converge. Methods of this type is known as optimistic policy iteration in the literature and have been widely used in practice. Although this method has no theoretical convergence guarantees, it has been shown to perform well in some situations [30–32].

4.4 Decomposition Approach

To obtain distributed algorithms and faster convergence especially for networks with large number of nodes and links, the immediate cost of an incoming call might be associated with its connection. This method is known as decomposition approach and was found to perform well and lead to shorter training times in [8].

In this approach, local states x^c, associated with each connection $c \in \mathcal{C}$ are considered although they are not real states in true sense. This is because of the fact that they are affected by the global state x and they do not evolve as a Markov process. Assuming that a new call arrives to connection c and its routing

and wavelength assignment is done, then the immediate cost of that connection becomes $g^c(x_{t_k}, e_k, u_{t_k}) = g(x_{t_k}, e_k, u_{t_k})$. For all other events, the immediate cost associated with connection c is set equal to 0.

Given a fixed policy μ, it can be shown that:

$$v(\mu) = \sum_{c \in \mathcal{C}} v^c(\mu) \tag{4.12}$$

where $v^c(\mu)$ represents the average blocking cost associated with connection c under policy μ. For each connection, it can be introduced a scalar $\tilde{v}^c$ as an estimate of $v^c(\mu)$ and an approximation architecture $\tilde{h}^c(x^c, \theta^c)$ where θ^c represents a parameter vector that belongs to the features associated with connection c. In this approach each connection c has its own policy μ^c. When a call request arrives to connection c then local policy μ^c is defined as:

$$\mu^c(x^c, e_k) = arg \min_{u \in U(x,e)} [g^c(x_{t_k}, e_k, u_{t_k}) + \tilde{h}^c(\acute{x}^c, \theta^c)] \tag{4.13}$$

The main idea of decomposition approach is not to update global parameter vector θ after each event, but instead update the parameter vector θ^c which is local to each connection. Under these definitions, if the incoming call belongs to connection c, then local TD(0) algorithm for this connection is given as:

$$\theta^c_k = \theta^c_{k-1} + \gamma^c_k d^c_k \nabla_\theta \tilde{h}^c(x^c_{t_{k-1}}, \theta^c_{k-1}) \tag{4.14}$$

$$\tilde{v}^c_k = \tilde{v}^c_{k-1} + \eta^c_k (g^c(x_{t_{k-1}}, e_{k-1}, u_{t_{k-1}}) - (t_k - t_{k-1})\tilde{v}^c_{k-1}) \tag{4.15}$$

$$d^c_k = g^c(x_{t_{k-1}}, e_{k-1}, u_{t_{k-1}}) + \tilde{h}^c(x^c_{t_k}, \theta^c_{k-1}) - (t_k - t_{k-1})\tilde{v}^c_{k-1} - \tilde{h}^c(x^c_{t_{k-1}}, \theta^c_{k-1}) \tag{4.16}$$

where γ^c_k and η^c_k are small diminishing step size parameters explained in the previous section. Since decomposition approach ignores some dependencies, it should not be expected to obtain better average blocking probabilities when compared to global TD(0) algorithm. However, at the expense of introducing an additional modeling error and obtaining higher average blocking probabilities one can obtain distributed algorithms and faster convergence rates.

Chapter 5

NUMERICAL RESULTS

In this chapter simulation results of neuro-dynamic programming method and heuristic algorithms on ring and mesh networks for three different network loads will be presented. Additionally, neuro-dynamic programming results are compared with the results of heuristic algorithms for the evaluation of performance. To train the parameter vector θ, both decomposition approach and global TD(0) algorithms are employed to obtain better average blocking probabilities. NDP method used in simulations is summarized in Figure 5.1. Finally, the numerical results are discussed.

5.1 Simulation Setup

All the simulations were done using *C++* programming. Since the aim of this thesis is to minimize the average blocking probability, one kind of service type is used in all simulations. This means that when a new call request is not admitted or a proper route and wavelength pair can not be found, immediate cost is chosen to be equal for each connection. For this reason, the problem of

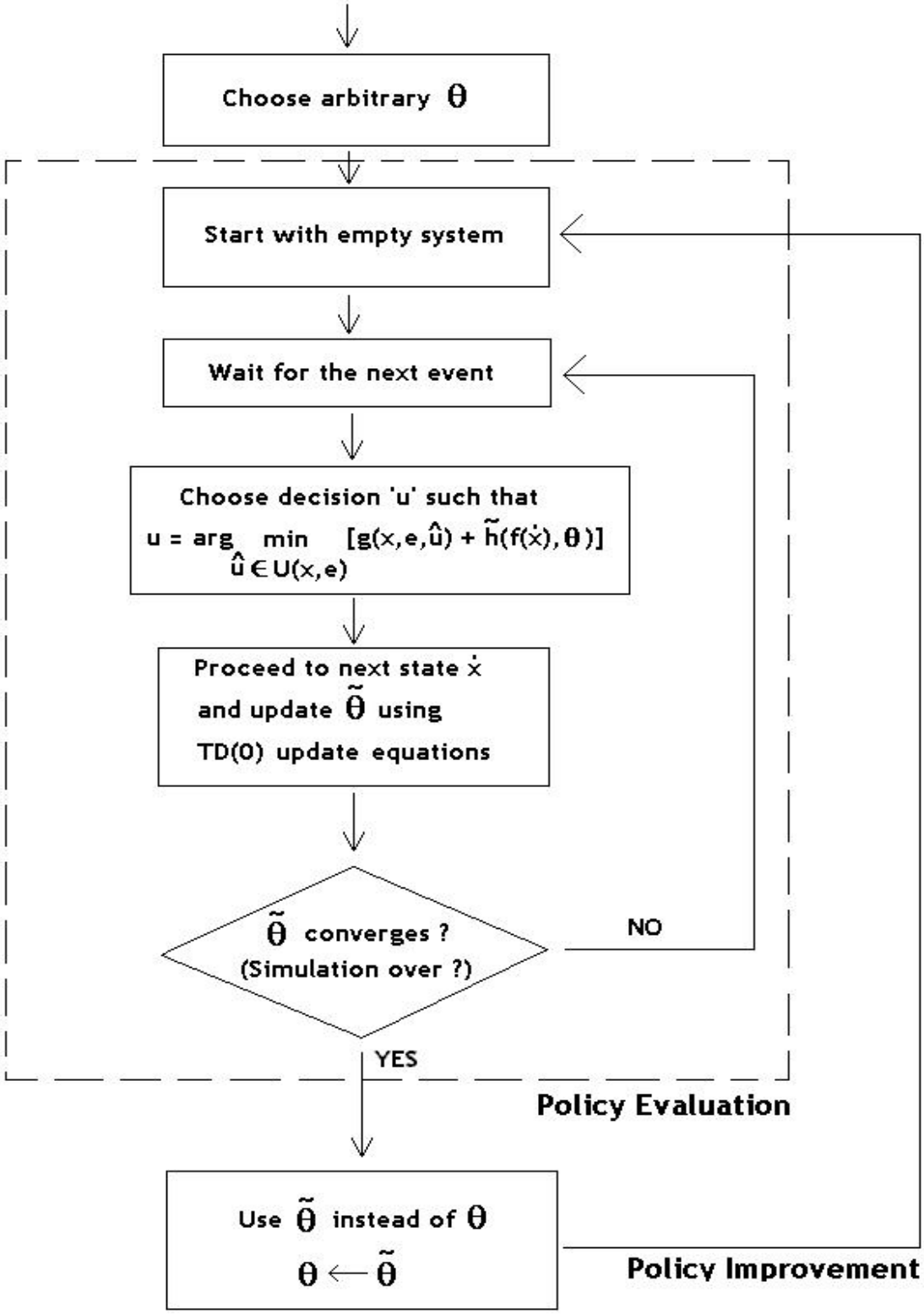

Figure 5.1: Flowchart of the NDP method.

minimizing average cost over an infinite time horizon is the same as the problem of minimizing average blocking probability.

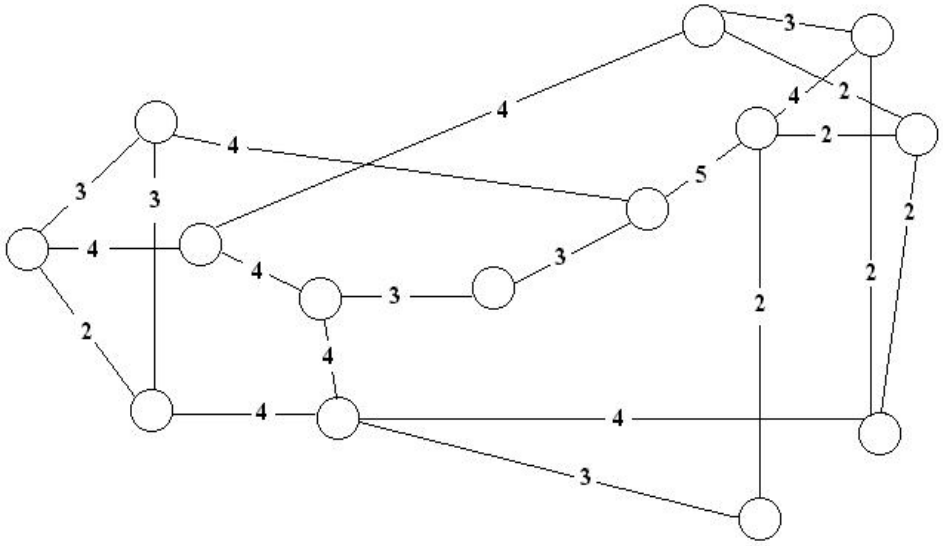

Figure 5.2: NSF mesh network. Number on each link shows the number of used optical fibers on that link.

For simulations, an 8 node ring network and a 14 node NSF mesh network was considered which is seen in Figure 5.2. In these networks, determining the number of fibers used on each link for the multi fiber case can be viewed as a new design problem [22, 33]. Most of the times, the design goal is to minimize the number of used fibers throughout all network. However, since an optimal network design is not a primary goal, number of fibers used on each link was calculated roughly proportional to the number of connections using that link and inversely proportional to the number of used wavelengths on each fiber. Additionally it is assumed that each fiber contains $W = 32$ wavelengths in both networks.

All the simulations uses m on-off type independent traffic sources on each possible connection between any source and destination pair. On-off type traffic can be explained as follows: each open traffic source closes after an exponentially distributed time with unit mean and each closed traffic source opens after an exponentially distributed time with mean $1/\lambda$.

Each network is inspected under four different situations. These are:

a) Fixed Routing, Single Fiber: For each connection, minimum hop route is selected for routing [34]. For each link only one fiber is used in both networks.

b) Fixed Routing, Multi Fiber: For each connection, minimum hop route is selected for routing. For 8 node ring network 3 fibers are used on each link and for the NSF network the numbers in Figure 5.2 are used.

c) Fixed Alternate Routing, Single Fiber: For each connection, minimum hop route is selected as a primary route and second shortest path is selected as an alternate route. For each link only one fiber is used in both networks.

d) Fixed Alternate Routing, Multi Fiber: For each connection, minimum hop route is selected as a primary route and second shortest path is selected as an alternate route. For 8 node ring network 3 fibers are used on each link and for the NSF network the numbers in Figure 5.2 are used.

Together with the results of heuristic algorithms and neuro-dynamic programming using several features, the average blocking probabilities when full wavelength conversion is done at each node (WI) are also presented for the fixed routing case. For the fixed alternate routing case with WI, the path P is chosen for call request coming to connection i that achieves

$$\max_{P\in\mathcal{P}_i} \min_{l\in P} \left[W M_l - \sum_{j=1}^{W} A_{lj} \right] \tag{5.1}$$

Since the wavelength continuity constraint is ignored, the resulting blocking probabilities clearly serve as a lower bound in the WS case.

In all simulations, 10^5 event steps are used for policy evaluation (critic). After 10^5 event steps, the policy is updated by adopting new parameter vector θ and

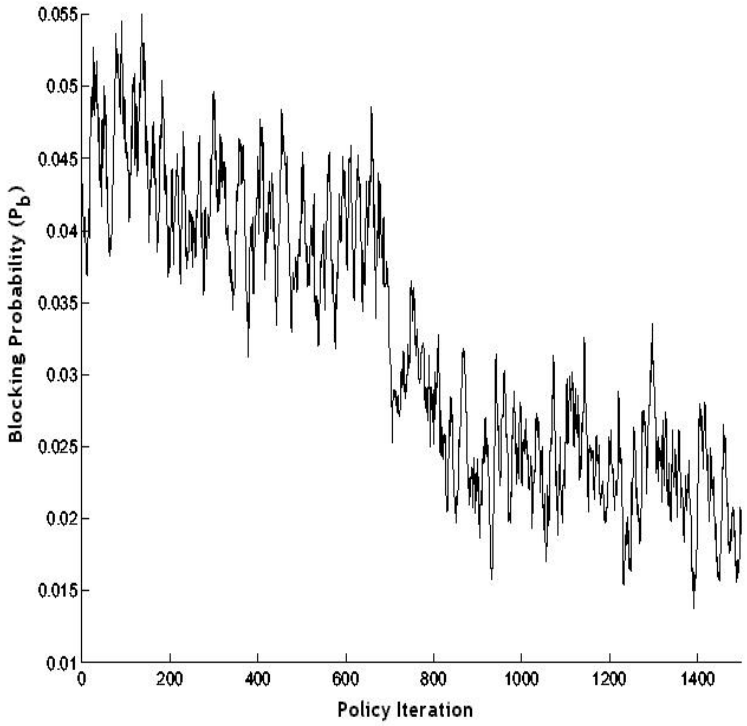

Figure 5.3: Blocking Probability (P_b) evaluated over 10^5 event steps versus Policy Iteration.

this process is repeated until a steady state behaviour in average blocking probabilities is obtained. Typical improving behaviour of neuro-dynamic programming can be seen in Figure 5.3.

Tables 5.1-5.12 gives the steady-state average blocking probabilities obtained by neuro-dynamic programming using the specified features together with the results of heuristic algorithms. The feature named as "*All*" in these tables represents the neuro-dynamic programming which uses the all features for the given case.

There can be maximum of $\binom{N}{2}m$ traffic sources active at any time in the network where N represents the number of nodes in the network. Let us assume that k represents the number of possible paths per connection. Each traffic source is assigned a path among k different paths and assigned a wavelength among W different wavelengths when it is activated. Therefore, for comparison purposes,

upper bound for the total number of possible states $\mid \mathcal{S} \mid$ can be given as:

$$\mid \mathcal{S} \mid \leq \left[kW\right]^{\binom{N}{2}m}.$$

$\mid \mathcal{S} \mid$ is usually smaller than the upper bound due to the capacity constraints. In simulations, minimum $\mid \mathcal{S} \mid$ occurs for the fixed routing case of 8 node ring network where $k = 1$, $N = 8$, $m = 32$ and $W = 32$. Even for this case, the upper bound for $\mid \mathcal{S} \mid$ is calculated to be in the order of $\sim 10^{1348}$ which is a very large number.

5.2 Discussion of Numerical Results

When we look at the average blocking probabilities presented in Tables 5.1-5.12, it is seen that neuro-dynamic programming method using some features can obtain better average blocking probabilities than all used heuristic algorithms. After careful inspection, it is seen that global availability and local availability features performs well while the other features does not provide significant improvement in terms of average blocking probability under the linear approximation architecture.

Better results are obtained when all features are combined compared to those results obtained by each feature set. In other words, when the number of features extracted from the network increases smaller blocking probabilities are observed, which is very intuitive. This is because of the fact that neuro-dynamic programming has the freedom to set all the parameters to zero except of those associated with the feature set that gives the best result of all. Therefore, neuro-dynamic programming guarantees to obtain at least the same result obtained by the feature set that gives the best average blocking probability when several number of feature sets are used.

From the given tables, it is seen that when the number of used fibers on each link and the number of alternate paths for each connection increases, the performance of the neuro-dynamic programming increases. It is seen that obtaining up to one third of the average blocking probabilities of the heuristic algorithms might be possible when all features are used in fixed alternate routing and multi fiber case.

For all simulations, the average blocking probability obtained by WI case is smaller than those obtained by heuristic algorithms and NDP. It is seen that the improvement of WI in terms of blocking probability is higher over heuristics for fixed alternate routing case. For both routing cases, NDP obtains very close results to WI, which is the main reason that NDP performs much better than heuristics in fixed alternate routing case.

One of the reasons to prefer linear approximation architecture was to determine the most relevant features in decision making process. Using the all feature sets, when the parameter vector θ is carefully examined after the steady state behaviour is obtained, it is seen that parameters associated with the local availability features are greater in magnitude to those associated with other feature sets. This observation shows that local availability features are more dominant in RWA problem in WDM optical networks. This observation also agrees with the simulation results that Max-Sum(MΣ) and Least-Loaded(LL) heuristic algorithms, which use local availability features, perform better than all other heuristics. Finally, this example shows us that NDP approach might be used to make an assesment on which heuristic performs better on large scale dynamic programming problems.

Max-Sum(MΣ) heuristic algorithm which obtains the best average blocking probabilities among all other heuristic algorithms can be interpreted as a policy which gives equal importance to local availability features in all connections. However, neuro-dynamic programming using local availability features, which

has the same computational complexity with MΣ heuristic, obtained smaller average blocking probabilities by tuning the parameters associated with these features such that they are negative and small in magnitude in shorter paths but negative and greater in magnitude in longer paths. In other words, as can be seen from 5.13, NDP blocked the long connections more and short connections less when compared to MΣ heuristic.

To observe the effects of the total number of on-off traffic sources per connection m over blocking probabilities, m is increased from 32 to 64 and the rate per traffic source λ is reduced to half. The results are presented in Tables 5.9 and 5.10. It is seen that blocking probabilities increases significantly with increasing m. Additionally, the improvements obtained by NDP over heuristics decreases. For $m = 1$, no blocking occured, since, even if all the on-off traffic sources are open for each connection, network is non-blocking. This is related to the design of the network where the number of used fibers on each link was calculated proportional to the number of connections using that link. However, for large values of m the network is blocking for some states due to capacity constraints. When m is increased and λ is increased by the same ratio, mean traffic rate does not change while the variance of the traffic increases. Therefore, for large values of m, the probability of the network to be in the blocking states increases. It is a well known fact that when m approaches to infinity traffic type converges to Poisson process.

In Table 5.11 and 5.12 the results of the NDP using decomposition approach for the fixed alternate routing and multi-fiber case of both ring and NSF mesh network can be seen. Although the improvement obtained in terms of average blocking probabilities is very small in both topologies compared to heuristic algorithms, the time required to obtain steady state policies is much smaller when compared to global TD(0) algorithm. Moreover, it should be noted that obtained policies are completely distributed and have the same computational

complexity as that of heuristic algorithms. The decomposition approach may possibly yield an effective and practical solution for the reward maximization problem. This extension is currently under consideration.

Fixed Routing - Single Fiber				
	λ=0.06383	λ=0.07527	λ=0.11111	Total # of features
Random	0.00562	0.02052	0.11339	-
First-Fit	0.00224	0.01167	0.09948	-
Most-Used	0.00146	0.00917	0.08831	-
Max-Sum	0.00142	0.00879	0.08321	-
WI	0.00071	0.00510	0.06889	-
Global Availability	0.00141	0.00878	0.08321	32
Local Availability	0.00133	0.00852	0.08315	896
Global Utilization	0.00145	0.00916	0.08827	32
All	0.00128	0.00826	0.08012	960

Table 5.1: Average blocking probabilities for the 8 node ring network with fixed routing and single fiber per link. m=32 is assumed.

Fixed Routing - Multi Fiber				
	λ=0.31579	λ=0.33333	λ=0.35135	Total # of features
Random	0.00459	0.00909	0.01575	-
First-Fit	0.00235	0.00539	0.01172	-
Most-Used	0.00196	0.00482	0.00958	-
Min-Sum	0.00222	0.00539	0.01077	-
Max-Sum	0.00131	0.00324	0.00761	-
Least-Loaded	0.00170	0.00346	0.00833	-
WI	0.00107	0.00294	0.00640	-
Global Availability	0.00129	0.00319	0.00758	32
Local Availability	0.00126	0.00311	0.00746	896
Global Utilization	0.00192	0.00478	0.00951	32
Sum	0.00211	0.00512	0.00987	896
Normal-Sum	0.00205	0.00506	0.00975	896
Sharing	0.00198	0.00495	0.00962	896
All	0.00111	0.00301	0.00721	3648

Table 5.2: Average blocking probabilities for the 8 node ring network with fixed routing and multi fiber per link. m=32 is assumed.

Fixed Alternate Routing - Single Fiber				
	λ=0.07527	λ=0.08696	λ=0.09890	Total # of features
Min-Sum	0.00294	0.01445	0.04244	-
Least-Loaded	0.00290	0.01328	0.03873	-
MΣ	0.00195	0.01127	0.03615	-
WI	0.00105	0.00526	0.02536	-
Sum	0.00269	0.01387	0.04006	1792
Local Availability	0.00151	0.00926	0.03221	1792
All	0.00138	0.00681	0.02749	3584

Table 5.3: Average blocking probabilities for the 8 node ring network with fixed alternate routing and single fiber per link. m=32 is assumed.

Fixed Alternate Routing - Multi Fiber				
	λ=0.35135	λ=0.36986	λ=0.38889	Total # of features
Min-Sum	0.00732	0.01438	0.02440	-
Least-Loaded	0.00159	0.00471	0.01202	-
MΣ	0.00142	0.00365	0.01153	-
WI	0.00045	0.00126	0.00365	-
Sum	0.00340	0.00885	0.01824	1792
Local Availability	0.00123	0.00284	0.01026	1792
All	0.00064	0.00143	0.00427	3584

Table 5.4: Average blocking probabilities for the 8 node ring network with fixed alternate routing and multi fiber per link. m=32 is assumed.

Fixed Routing - Single Fiber				
	λ=0.04167	λ=0.04712	λ=0.05263	Total # of features
Random	0.00372	0.01174	0.02648	-
First-Fit	0.00174	0.00703	0.01931	-
Most-Used	0.00133	0.00593	0.01664	-
Max-Sum	0.00091	0.00461	0.01193	-
WI	0.00058	0.00260	0.00849	-
Global Availability	0.00090	0.00452	0.01189	32
Local Availability	0.00082	0.00421	0.01053	2912
Global Utilization	0.00134	0.00601	0.00165	32
All	0.00074	0.00331	0.00980	2976

Table 5.5: Average blocking probabilities for the NSF mesh network with fixed routing and single fiber per link. m=64 is assumed.

Fixed Routing - Multi Fiber				
	λ=0.31579	λ=0.33333	λ=0.35135	Total # of features
Random	0.00549	0.01090	0.01897	-
First-Fit	0.00392	0.00892	0.01702	-
Most-Used	0.00362	0.00825	0.01612	-
Min-Sum	0.00273	0.00732	0.01403	-
Max-Sum	0.00186	0.00482	0.00969	-
Least-Loaded	0.00191	0.00513	0.01071	-
WI	0.00178	0.00421	0.00939	-
Global Availability	0.00186	0.00478	0.00968	32
Local Availability	0.00184	0.00456	0.00956	2912
Global Utilization	0.00360	0.00822	0.01611	32
Sum	0.00269	0.00719	0.01389	2912
Normal-Sum	0.00258	0.00708	0.01382	2912
Sharing	0.00246	0.00685	0.01356	2912
All	0.00180	0.00432	0.00945	11712

Table 5.6: Average blocking probabilities for the NSF mesh network with fixed routing and multi fiber per link . m=64 is assumed.

Fixed Alternate Routing - Single Fiber				
	λ=0.05263	λ=0.05820	λ=0.06383	Total # of features
Min-Sum	0.00611	0.01613	0.03221	-
Least-Loaded	0.00606	0.01515	0.03203	-
MΣ	0.00532	0.01465	0.03096	-
WI	0.00256	0.00885	0.02267	-
Sum	0.00515	0.01451	0.03220	5824
Local Availability	0.00435	0.01340	0.02817	5824
All	0.00313	0.01153	0.02783	11648

Table 5.7: Average blocking probabilities for the NSF mesh network with fixed alternate routing and single fiber per link. m=64 is assumed.

Fixed Alternate Routing - Multi Fiber				
	λ=0.34228	λ=0.36054	λ=0.38889	Total # of features
Min-Sum	0.00634	0.01357	0.03079	-
Least-Loaded	0.00214	0.00377	0.01059	-
MΣ	0.00162	0.00309	0.01045	-
WI	0.00052	0.00148	0.00384	-
Sum	0.00291	0.00754	0.01788	5824
Local Availability	0.00121	0.00228	0.01026	5824
All	0.00075	0.00176	0.00427	11648

Table 5.8: Average blocking probabilities for the NSF mesh network with fixed alternate routing and multi fiber per link. m=32 is assumed.

Fixed Alternate Routing - Multi Fiber				
	λ=0.175675	λ=0.184930	λ=0.194445	Total # of features
Min-Sum	0.06515	0.08532	0.10544	-
Least-Loaded	0.05304	0.07638	0.09489	-
MΣ	0.05016	0.07362	0.09273	-
WI	0.03618	0.05721	0.08261	-
Sum	0.05948	0.07927	0.09756	1792
Local Availability	0.04567	0.06938	0.08956	1792
All	0.04294	0.06543	0.08512	3584

Table 5.9: Average blocking probabilities obtained by heuristic algorithms and NDP using global TD(0) for the 8 node ring network. m=64 is assumed.

Fixed Alternate Routing - Multi Fiber				
	λ=0.171140	λ=0.180270	λ=0.194445	Total # of features
Min-Sum	0.02475	0.03469	0.06159	-
Least-Loaded	0.02436	0.03131	0.05924	-
MΣ	0.02375	0.03056	0.05812	-
WI	0.01853	0.02431	0.05216	-
Sum	0.02452	0.03246	0.06026	5824
Local Availability	0.02214	0.02958	0.05768	5824
All	0.02184	0.02769	0.05543	11648

Table 5.10: Average blocking probabilities obtained by heuristic algorithms and NDP using global TD(0) for the NSF mesh network. m=64 is assumed.

Fixed Alternate Routing - Multi Fiber				
	λ=0.35135	λ=0.36986	λ=0.38889	Total # of features
Min-Sum	0.00732	0.01438	0.02440	-
Least-Loaded	0.00159	0.00471	0.01202	-
MΣ	0.00142	0.00365	0.01153	-
WI	0.00045	0.00126	0.00365	-
Sum	0.00722	0.01395	0.02363	1792
Local Availability	0.00149	0.00452	0.01154	1792
All	0.00139	0.00394	0.00993	3584

Table 5.11: Average blocking probabilities obtained by heuristic algorithms and NDP using decomposition approach for the 8 node ring network. m=32 is assumed.

Fixed Alternate Routing - Multi Fiber				
	λ=0.34228	λ=0.36054	λ=0.38889	Total # of features
Min-Sum	0.00634	0.01357	0.03079	-
Least-Loaded	0.00214	0.00377	0.01059	-
MΣ	0.00162	0.00309	0.01045	-
WI	0.00052	0.00148	0.00384	-
Sum	0.00612	0.01313	0.02986	5824
Local Availability	0.00209	0.00369	0.01042	5824
All	0.00185	0.00345	0.00967	11648

Table 5.12: Average blocking probabilities obtained by heuristic algorithms and NDP using decomposition approach for the NSF mesh network. m=32 is assumed.

Fixed Alternate Routing - Multi Fiber				
λ=0.33333	1 Hop	2 Hop	3 Hop	4 Hop
Max-Sum	0.00039	0.00155	0.00414	0.00939
Local Availability	0.00021	0.00119	0.00452	0.01010

Table 5.13: Average blocking probability versus path-length obtained by Max-Sum heuristic and NDP using Local Availability feature. The experiments are conducted on the 8 node ring network with m=32 traffic sources for each connection.

Chapter 6

SUMMARY

Simulation results with two different networks show us that neuro-dynamic programming techniques can be applied to the RWA problem in WDM optical networks. It is seen that smaller average blocking probabilities than those of all heuristic algorithms might be obtained. Moreover, by combining the features of several heuristics, a new policy which performs better or at least the same as those heuristics can be obtained.

The time required to obtain steady state behaviour in neuro-dynamic programming increases when the size of the network and the number of features that is required to extract from the network increases. It is seen from simulations that when the number of features are increased, better performance might be obtained. Therefore, there is a trade-off between obtaining smaller average blocking probabilities and the time required for the parameter vector θ to converge. For example, the time required to obtain steady state behaviour for different simulations changes between one and eight hours in Pentium III 650 MHz PC. From this observation, NDP seems to be best suited as a tool for off-line rather than online policy improvement. Although training process of the parameter vector θ might take long, once the proper parameter vector is

obtained, the implementation of neuro-dynamic programming method requires only the extraction of the necessary features from the network.

In this thesis a novel connection based decomposition approach for WDM optical networks is proposed. Although it does not provide as high improvements over heuristics as in global TD(0), it is seen that this method greatly reduces the training time when compared to global TD(0) method where all the parameters are updated together. The proposed decomposition approach allows us to obtain distributed algorithms for each connection which only require the features associated with the specified connection. Therefore, once the parameters are obtained, in the implementation phase these distributed algorithms have the same computational complexity as used heuritics.

Average blocking probabilities of call requests belonging to connections with longer paths is greater than those that belongs to connections with shorter paths. This problem is a well known fairness issue, and it limits us to give fair service to all call requests that belongs to different connections. The aim of this thesis is to minimize the average blocking probability. Therefore, in all simulations the immediate cost of all call requests were set equal for all connections between any source and destination pairs. To overcome the fairness issue, one can increase the immediate cost of call requests belonging to connections that greatly suffer from this issue and might obtain smaller average blocking probabilities for these connections. However, when different immediate costs are assigned to call requests belonging to different connections, minimizing the average blocking cost does not mean minimizing the average blocking probability. It is clear that there is a trade-off between the problem of minimizing the average blocking probability and the fairness issue and NDP seems to be a suitable tool for this trade off.

Finally, maximizing the total revenue for the networks where different connections are of different values is a commonly encountered problem in practical

implementations. NDP method presented in this thesis can also be applied to such problems in order to obtain effective policies.

Bibliography

[1] R. Ramaswami, "Multiwavelength lightwave network for communication," *IEEE Communication Magazine*, pp. 78–88, 1993.

[2] M. Settembre and F. Matera, "All optical implementations of high capacity TDMA networks," *Fiber and Integrated Optics*, vol. 12, pp. 173–186, 1993.

[3] R. Ramaswami and K. N. Sivarajan, *Optical Networks: A Practical Perspective*. San Francisco, California: Morgan Kaufmann Publishers, Inc., 1998.

[4] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[5] G. J. Tesauro, "Practical issues in temporal difference learning," *Machine Learning*, vol. 8, pp. 257–277, 1992.

[6] S. Singh and D. Bertsekas, "Reinforcement learning for dynamic channel allocation in cellular telephone systems," *Submitted to NIPS96*, 1996.

[7] R. H. Crites and A. G. Barto, "Improving elevator performance using reinforcement learning," *Advances in Neural Information Processing Systems 8*, 1996.

[8] P. Marbach, O. Mihatsch, and J. N. Tsitsiklis, "Call admission control and routing in integrated services networks using neuro-dynamic programming," *IEEE JSAC*, vol. 18, pp. 197–208, February 2000.

[9] R. Ramaswami and K. N. Sivarajan, "Routing and wavelength assignment in all-optical networks," *IEEE/ACM Trans. on Networking*, vol. 3, pp. 489–500, October 1995.

[10] A. G. I. Chlamtac and G. Karmi, "Lightpath communications: an approach to high bandwidth optical WAN's," *IEEE Trans. on Communications*, vol. 40, pp. 1171–1182, July 1992.

[11] D. Banerjee and B. Mukherjee, "A practical approach for routing and wavelength assignment in large wavelength-routed optical networks," *IEEE JSAC*, vol. 14, pp. 903–908, June 1996.

[12] O. Gerstel, G. Sasaki, S. Kutten, and R. Ramaswami, "Wortst-case analysis of dynamic wavelength allocation in optical networks," *IEEE/ACM Trans. on Networking*, vol. 7, pp. 833–844, December 1999.

[13] S. Ramamurthy, *Optical Design of WDM Network Architectures*. PhD thesis, University of California, 1998.

[14] K. Chan and T. P. Yum, "Analysis of least congested path routing in wdm ligthwave networks," *Proc. IEEE INFOCOM*, vol. 2, pp. 962–969, April 1994.

[15] L. Li and A. K. Somani, "Dynamic wavelength routing using congestion and neighborhood information," *IEEE/ACM Trans. on Networking*, 1999.

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

[17] H. Zang, J. P. Jue, and B. Mukherjee, "A review of routing and wavelength assignment approaches for wavelength routed optical WDM networks," *Optical Networks Magazine*, 2000.

[18] M. Kovacevic and A. Acampora, "On wavelength translation in all-optical networks," *Proc. IEEE INFOCOM*, pp. 413–422, April 1995.

[19] R. A. Barry and P. A. Humblet, "Models of blocking probability in all-optical networks with and without wavelength changers," *Proc. IEEE INFOCOM*, pp. 402–412, April 1995.

[20] A. Mokhtar and M. Azizoglu, "Adaptive wavelength routing in all-optical networks," *IEEE/ACM Trans. on Networking*, vol. 6, pp. 197–206, April 1998.

[21] S. Subramaniam and R. Barry, "Wavelength assignment in fixed routing WDM networks," *Proceedings of IEEE ICC*, pp. 406–415, November 1997.

[22] E. Karasan and E. Ayanoglu, "Effects of wavelength routing and selection algorithms on wavelength conversion gain in WDM optical networks," *IEEE/ACM Trans. on Networking*, vol. 6, pp. 186–196, April 1998.

[23] R. A. Barry and S. Subramaniam, "The max-sum wavelength assignment algorithm for WDM ring networks," *Proc. OFC*, February 1997.

[24] S. Hayking, *Neural Networks: A Comprehensive Foundation.* New York: McMillian, 1994.

[25] M. Alanyali and E. Ayanoglu, "Provisioning algorithms for WDM optical networks," *IEEE/ACM Trans. on Networking*, vol. 7, pp. 767–778, October 1999.

[26] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, pp. 9–44, 1988.

[27] J. N. Tsitsiklis and B. V. Roy, "Average cost temporal-difference learning," *MIT, Cambridge, MA, Lab. Inform. Decision Syst. Report LIDS-P-2390*, May 1997.

[28] J. N. Tsitsiklis and B. V. Roy, "On average versus discounted reward temporal-difference learning," *MIT, Cambridge, MA, Lab. Inform. Decision Syst. Report*, March 1999.

[29] J. N. Tsitsiklis and B. V. Roy, "An analysis of temporal-difference learning with function approximation," *IEEE Trans. Automatic Control*, vol. 42, pp. 674–690, May 1997.

[30] S. Singh and D. P. Bertsekas, "Reinforcement learning for dynamic channel allocation in cellular telephone systems," *Advances in Neural Information Processing Systems 9*, pp. 974–980, 1997.

[31] J. Tesauro, "Practical issues in temporal difference learning," *Machine Learning*, vol. 8, 1988.

[32] W. Zhang and T. G. Dietterich, "High performance job-shop scheduling with a time-delay TD(λ) network," *Advances in Neural Information Processing Systems 8*, pp. 1024–1030, 1996.

[33] G. Jeong and E. Ayanoglu, "Comparison of wavelength-interchanging and wavelength-selective cross-connects in multiwavelength all-optical networks," *Proc. IEEE INFOCOM*, pp. 156–163, March 1996.

[34] A. Girard, *Routing and dimensioning in circuit-switched networks*. Addison-Wesley, 1990.