Traffic Adaptive WDM Networks: A Study of Reconfiguration Issues

Ilia Baldine and George N. Rouskas, *Member, IEEE*

***Abstract*—This paper studies the issues arising in the reconfiguration phase of broadcast optical networks. Although the ability to dynamically optimize the network under changing traffic conditions has been recognized as one of the key features of multiwavelength optical networks, this is the first in-depth study of the tradeoffs involved in carrying out the reconfiguration process. We develop and compare reconfiguration policies to determine *when* to reconfigure the network, and we present an approach to carry out the network transition by describing a class of strategies that determine *how* to retune the optical transceivers. We identify the degree of load balancing and the number of retunings as two important, albeit conflicting, objectives in the design of reconfiguration policies, naturally leading to a formulation of the problem as a Markovian decision process. Consequently, we develop a systematic and flexible framework in which to view and contrast reconfiguration policies. We show how an appropriate selection of reward and cost functions can be used to achieve the desired balance among various performance criteria of interest. We conduct a comprehensive evaluation of reconfiguration policies and retuning strategies and demonstrate the benefits of reconfiguration through both analytical and simulation results. The result of our work is a set of practical techniques for managing the network transition phase that can be directly applied to networks of large size. Although our work is in the context of broadcast networks, the results can be applied to any wavelength-division multiplexing network where it is necessary to multiplex traffic from a large user population into a number of wavelengths.**

***Index Terms*—Broadcast optical networks, Markov decision process, reconfiguration policies, wavelength-division multiplexing (WDM).**

I. Introduction

ONE OF the key features of multiwavelength optical networks is *rearrangeability* [4], i.e., the ability to dynamically optimize the network for changing traffic patterns, or to cope with failure of network equipment. This ability arises as a consequence of the independence between the logical connectivity and the underlying physical infrastructure of fiber glass. By employing tunable optical devices, the assignment of transmitting or receiving wavelengths to the various network nodes may be updated on the fly, allowing the network to closely track changing traffic conditions.

While the rearrangeability property makes it possible to design traffic-adaptive self-healing networks, the reconfiguration phase will interfere with existing traffic and disrupt network performance, causing a degradation of the quality of service perceived by the users. The issues that arise in reconfiguring a lightwave network by retuning a set of slowly tunable transmitters or receivers have been studied in the context of multihop networks in [5], [6], and [9]. In [5], the problem of obtaining a virtual topology that minimizes the maximum link flow, given a set of traffic demands, was studied, while in [6], algorithms were developed for minimizing the number of branch-exchange operations required to take the network from an initial to a target virtual topology, once the traffic pattern changes. The objective of [9], on the other hand, was to obtain near-optimal policies to dynamically determine when and how to reconfigure the network.

In this paper, we study the reconfiguration issues arising in single-hop lightwave networks, an architecture suitable for local- and metropolitan-area networks (LANs and MANs) [7]. The single-hop architecture employs wavelength-division multiplexing (WDM) to provide connectivity among the network nodes. The various channels are dynamically shared by the attached nodes, and the logical connections change on a packet-by-packet basis creating all-optical paths between sources and destinations. Thus single-hop networks require the use of rapidly tunable optical lasers and/or filters that can switch between channels at high speeds.

When tunability only at one end, say, at the transmitters, is employed, each fixed receiver is permanently assigned to one of the wavelengths used for packet transmissions. In a typical near-term WDM environment, the number of channels supported within the optical medium is expected to be smaller than the number of attached nodes. As a result, each channel will have to be shared by multiple receivers, and the problem of assigning receive wavelengths arises. Intuitively, a wavelength assignment (WLA) must be somehow based on the prevailing traffic conditions. More specifically, the stability condition, derived in [11], for the HiPeR-ℓ reservation protocol for broadcast WDM networks suggests that in determining an appropriate WLA, the objective should be to balance the offered load across all channels, such that each channel carries an approximately equal portion of the overall traffic.[1] But with fixed receivers, any WLA is permanent and cannot be updated in response to changes in the traffic pattern.

Alternatively, one can use *slowly tunable*, rather than fixed, receivers. We will say that an optical laser or filter is rapidly tunable if its tuning latency (i.e., the time it takes to switch from one wavelength to another) is on the order of a packet trans-

Manuscript received March 15, 2000; revised January 3, 2001. This work was supported by the National Science Foundation under Grant NCR-9701113.

The authors are with the Department of Computer Science, North Carolina State University, Raleigh, NC 27695-7534 USA (e-mail: rouskas@csc.ncsu.edu).

Publisher Item Identifier S 0733-8724(01)02752-9.

[1]This result is intuitive and has been accepted by the research community for years. However, by deriving a stability condition for HiPeR-ℓ [11], our study was the first to quantify the effect of load balancing on the performance of broadcast optical networks.

mission time at the high-speed rates at which optical networks are expected to operate. Slowly tunable devices, on the other hand, have tuning times that can be significantly longer. As a result, these devices cannot be assumed "tunable" at the media access level (i.e., for the purposes of scheduling packet transmissions), as this requires fast tunability. Motivation for the use of slowly tunable lasers or filters is provided by two factors. First, they can be significantly less expensive than rapidly tunable devices, making it possible to design lightwave network architectures that can be realized cost effectively. Second, the variation in traffic demands is expected to take place over larger time scales (several orders of magnitude larger than a single packet transmission time). Hence, even very slow tunable devices will be adequate for updating the WLA over time to accommodate varying traffic demands.

Assuming an existing WLA and some information about the new traffic demands, a new WLA, optimized for the new traffic pattern, must be determined. We considered this problem in [1], and we proposed an approach to reconfiguring the network that is minimally disruptive to existing traffic. Specifically, we devised the *GLPT* algorithm for obtaining a new WLA such that a) the new traffic load is balanced across the channels and b) the number of receivers that need to be retuned to take the network from the old to the new WLA is minimized. The specifications of GLPT include a *knob* parameter, which provides for tradeoff selection between load balancing and number of retunings. In terms of load balancing, the WLA obtained by GLPT is guaranteed to be no more than two times away from the optimal one, in the worst case, regardless of the knob value used. GLPT also leads to a scalable approach to reconfiguring the network, since it tends to select the less utilized receivers for retuning, and since for certain values of the knob parameter the expected number of retunings scales with the number of channels, *not* the number of nodes in the network. For more details on the operation and performance of the GLPT algorithm, the reader is referred to [1].

During the reconfiguration phase, while the network makes a transition from one WLA to another, some cost is incurred in terms of packet delay, packet loss, packet desequencing, and the control resources involved in receiver retuning. Clearly, receiver retunings should not be very frequent, since unnecessary retunings affect the performance encountered by the users. Hence, it is desirable to minimize the number of network reconfigurations. However, postponing a necessary reconfiguration also has adverse effects on the overall performance. Since the network does not operate at an optimal point in terms of load balancing, it takes longer to clear a given set of traffic demands, causing longer delays and/or buffer overflows, as well as a decrease in the network's traffic carrying capacity (refer also to the stability condition in [11]). Similarly, if the decisions are made merely by considering the degree of load balancing, even tiny changes in the traffic demands can lead to constant reconfiguration, thereby significantly hurting network performance. Consequently, it is important to have a performance criterion that can capture the above tradeoffs in an appropriate manner and allow their simultaneous optimization.

In this paper, we develop a novel, systematic, and flexible framework in which to view and contrast reconfiguration policies. Specifically, we formulate the problem as a Markovian decision process and show how an appropriate selection of reward and cost functions can achieve the desired balance between various performance criteria of interest. However, because of the huge state space of the underlying Markov process, it is impossible to directly apply appropriate numerical methods to obtain an optimal policy. We therefore develop an approximate model with a manageable state space, which captures the pertinent properties of the original model. We also study the issues that arise during the transition phase and we present a class of retuning strategies. Finally, we present a comprehensive evaluation of reconfiguration policies and retuning strategies through both analytical and simulation techniques.

In Section II, we present a model of the broadcast WDM network under study. In Section III, we formulate the reconfiguration problem as a Markovian decision process, and we discuss the issues involved in obtaining an optimal policy. In Section IV, we describe a class of retuning strategies for taking the network from the old to the new WLA. We present numerical results in Section V and conclude this paper in Section VI.

II. The Broadcast WDM Network

We consider a packet-switched single-hop lightwave network with N nodes, and one transmitter-receiver pair per node. The nodes are physically connected to a passive broadcast optical medium that supports $C < N$ wavelengths, $\lambda_1, \ldots, \lambda_C$. Both the transmitter and the receiver at each node are tunable over the entire range of available wavelengths. However, the transmitters are *rapidly tunable* while the receivers are *slowly tunable*. We will refer to this tunability configuration, shown in Fig. 1, as *rapidly tunable transmitter, slowly tunable receiver* (RTT-STR). Although we will only consider RTT-STR networks in this paper, we note that all our results can be easily adapted to the dual configuration, STT-RTR. Further, the results can be applied to any WDM network where it is necessary to multiplex traffic from a large user population into a number of wavelengths, and they are not limited to broadcast networks.

We represent the current traffic conditions in the network by an $N \times N$ traffic demand matrix $\mathbf{T} = [t_{ij}]$. Quantity t_{ij} could be a measure of the average traffic originating at node i and terminating at node j, or it could be the effective bandwidth [8] of the traffic from i to j. As traffic varies over time, the elements of matrix $\mathbf{T}$will change. This variation in traffic takes place at larger scales in time; for instance, we assume that changes in the traffic matrix $\mathbf{T}$occur at connection request arrival or termination instants. We also assume that the current matrix $\mathbf{T}$ completely summarizes the entire history of traffic changes, so that future changes only depend on the current values of the elements of $\mathbf{T}$.

During normal operation, each of the slowly tunable receivers is assumed to be fixed to a particular wavelength. Let $\lambda(j) \in \{\lambda_1, \ldots, \lambda_C\}$ be the wavelength currently assigned to receiver j. A WLA is a partition $\mathcal{R} = \{R_c,\ c = 1, \ldots, C\}$ of the set $\mathcal{N} = \{1, \ldots, N\}$ of nodes, such that $R_c = \{j \mid \lambda(j) = \lambda_c\},\ c = 1, \ldots, C$, is the subset of nodes currently receiving on wavelength λ_c. This WLA is known to the network nodes, and it is used to determine the target channel for a packet, given the packet's destination. The network operates by having each node

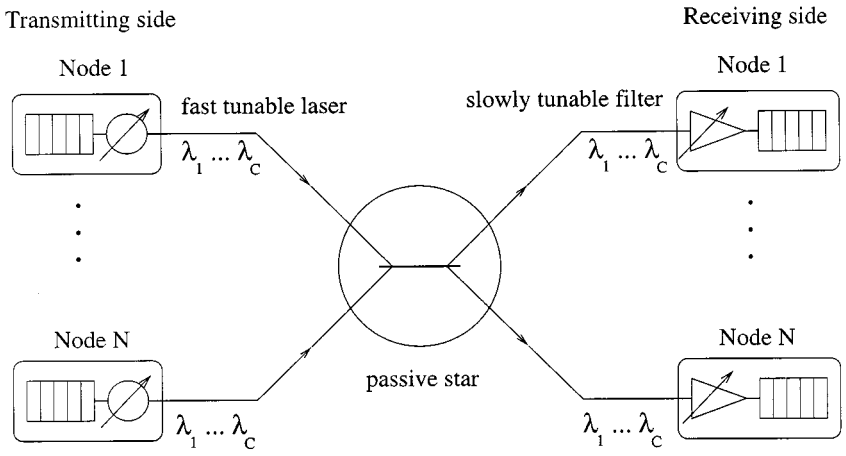

Fig. 1. A broadcast WDM network with N nodes and C channels.

employ a media access protocol, such as HiPeR-ℓ, that requires tunability only at the transmitting end. Nodes use HiPeR-ℓ to make reservations, and can schedule packets for transmission using algorithms that can effectively mask the (relatively short) latency of tunable transmitters [10].

We now define the *degree of load balancing* (DLB) $\phi(\mathcal{R}, \mathbf{T})$ for a network with traffic matrix $\mathbf{T}$ operating under WLA $\mathcal{R}$ as

$$(1+\phi(\mathcal{R},\mathbf{T}))\frac{\sum_{i=1}^{N}\sum_{j=1}^{N} t_{ij}}{C} = \max_{c=1,\ldots,C}\left\{\sum_{i=1}^{N}\sum_{j\in R_c} t_{ij}\right\}. \tag{1}$$

The right-hand side of (1) represents the bandwidth requirement of the dominant (i.e., most loaded) channel, while the second term in the left-hand side of (1) represents the lower bound, with respect to load balancing, for any WLA for traffic matrix $\mathbf{T}$. Thus, the DLB is a measure of how far away WLA $\mathcal{R}$ is from the lower bound. If $\phi = 0$, then the load is perfectly balanced and each channel carries an equal portion of the offered traffic, while when $\phi > 0$, the channels are not equally loaded. In other words, the DLB characterizes the efficiency of the network in meeting the traffic demands denoted by matrix $\mathbf{T}$ while operating under WLA $\mathcal{R}$: the higher the value of ϕ, the less efficient the WLA.

In order to more efficiently utilize the bandwidth of the optical medium as traffic varies over time, a new WLA may be sought that distributes the new load more equally among the channels. We will refer to the transition of the network from one WLA to another as *reconfiguration*. In general, we assume that reconfiguration is triggered by changes in the traffic matrix $\mathbf{T}$. When such a change occurs, the following actions must be taken.

1) A new WLA for the new traffic matrix must be determined.
2) A decision must be made on whether or not to reconfigure the network by adopting the new WLA.
3) If the decision is to reconfigure, the actual retuning of receivers must take place.

The first issue was addressed in [1], where we developed the GLPT algorithm. It takes as input the current WLA $\mathcal{R}$ and the new traffic matrix $\mathbf{T}'$, and determines the new WLA. Section III addresses the problem of determining whether the changes in traffic conditions warrant the reconfiguration of the network to the new WLA, and Section IV studies the issue of receiver retuning.

III. Markov Decision Process Formulation

A. Reconfiguration Policies

We define the state of the network as a tuple $(\mathcal{R}, \mathbf{T})$. $\mathcal{R}$ is the current WLA, and $\mathbf{T}$is a matrix representing the prevailing traffic conditions. Changes in the network state occur at instants when the matrix $\mathbf{T}$ is updated. Recall that we have assumed that future traffic changes only depend on the current values of the elements of $\mathbf{T}$, a reasonable assumption when the traffic is not characterized by long-range dependencies. Therefore, the process $(\mathcal{R}, \mathbf{T})$ is a semi-Markov process. Let $\mathcal{M}$ be the process embedded at instants when the traffic matrix changes. Then, $\mathcal{M}$ is a discrete-time Markov process. Our formulation is in terms of the Markov process $\mathcal{M}$.

A network in state $(\mathcal{R}, \mathbf{T})$ will enter state $(\mathcal{R}', \mathbf{T}')$ if the traffic matrix changes to $\mathbf{T}'$. Implicit in the state transition is that the system makes a decision to reconfigure to WLA $\mathcal{R}'$. In order to completely define the Markovian state transitions associated with our model, we need to establish *next WLA* decisions. The decision is a function of the current state and is denoted by $d[(\mathcal{R}, \mathbf{T})]$. Setting $d[(\mathcal{R}, \mathbf{T})] = \mathcal{R}_{\text{next}}$ implies that if the system is in state $(\mathcal{R}, \mathbf{T})$ and the traffic demands change, the network should be reconfigured into WLA $\mathcal{R}_{\text{next}}$. Note that WLA $\mathcal{R}_{\text{next}}$ can be the same as $\mathcal{R}$, in which case the decision is not to reconfigure. Therefore, for each state $(\mathcal{R}, \mathbf{T})$, there are two alternatives: either the network reconfigures to WLA $\mathcal{R}'$ obtained by the GLPT algorithm with $\mathcal{R}$ and $\mathbf{T}'$ as inputs (in which case the new state will be $(\mathcal{R}', \mathbf{T}')$) or it maintains the current WLA (in which case the new state will be $(\mathcal{R}, \mathbf{T}')$). The set of decisions for all network states defines a *reconfiguration policy*.

To formulate the problem as a Markov decision process, we need to specify reward and cost functions associated with each transition. Consider a network in state $(\mathcal{R}, \mathbf{T})$ that makes a tran-

sition to state $(\mathcal{R}', \mathbf{T}')$. The network acquires an *immediate expected reward* equal to $\alpha[\phi(\mathcal{R}', \mathbf{T}')]$, where $\alpha(\cdot)$ is a nonincreasing function of $\phi(\mathcal{R}', \mathbf{T}')$, the DLB of WLA $\mathcal{R}'$ with respect to the new traffic matrix $\mathbf{T}'$. Also, if $\mathcal{R}' \neq \mathcal{R}$, a *reconfiguration cost* equal to $\beta[\mathcal{D}(\mathcal{R}, \mathcal{R}')]$ is incurred, where $\beta(\cdot)$ is a nondecreasing function of the number of receivers that have to be retuned to take the network to the new WLA $\mathcal{R}'$. In other words, a switching cost is incurred each time the network makes a decision to reconfigure. We assume that the rewards and costs are bounded, i.e.,

$$\begin{aligned} \alpha_{\min} \leq & \alpha[\phi(\mathcal{R}', \mathbf{T}')] \leq \alpha_{\max} \\ 0 \leq & \beta_{\min} \leq \beta[\mathcal{D}(\mathcal{R}, \mathcal{R}')] \leq \beta_{\max} \end{aligned} \tag{2}$$

where $\alpha_{\min}$, $\alpha_{\max}$, $\beta_{\min}$, and $\beta_{\max}$ are real numbers.

The problem is how to reconfigure the network sequentially in time, so as to maximize the expected reward minus the reconfiguration cost over an infinite horizon. Let $(\mathcal{R}^{(k)}, \mathbf{T}^{(k)})$ denote the state of the network immediately after the k-th transition, $k = 1, 2, \ldots$ Let also Z be the set of admissible policies. The network reconfiguration problem can then be formally stated as follows (note that $\mathcal{D}(\mathcal{R}, \mathcal{R}) = 0$).

Problem 3.1: Find an optimal policy $z^\star \in Z$ that maximizes the expected reward

$$F = \lim_{k \to \infty} \frac{1}{k} E \left\{ \sum_{l=1}^{k} \alpha[\phi(\mathcal{R}^{(l)}, \mathbf{T}^{(l)})] - \beta[\mathcal{D}(\mathcal{R}^{(l-1)}, \mathcal{R}^{(l)})] \right\}. \tag{3}$$

The first term on the right-hand side of (3) is the reward obtained by using a particular WLA, and the second term is the cost incurred at each instant of time that reconfiguration is performed. The presence of a reward that increases as the DLB ϕ decreases (i.e., as the load is better balanced across the channels) provides the network with an incentive to associate with a WLA that performs well for the current traffic load. On the other hand, the introduction of a cost incurred at each reconfiguration instant discourages frequent reconfigurations. Thus, the overall reward function captures the fundamental tradeoff between the DLB and frequent retunings involved in the reconfiguration problem. In Section V, we motivate the above formulation by showing how an appropriate selection of reward and cost functions yields various performance criteria of interest. Typically, such selection can be based on either measurements of an existing network or simulations.

For the case $\beta_{\max} = 0$, the problem of finding an optimal policy is trivial, since it is optimal for the network to associate with the WLA that best balances the offered load at each instant in time. This is because the evolution of the traffic matrix $\mathbf{T}$ is not affected by the network's actions and reconfigurations are free. However, when $\beta_{\max} > 0$, there is a conflict between *future reconfiguration costs incurred* and *current reward obtained*, and it is not obvious as to what constitutes an optimal policy. We also note that as $\beta_{\min} \to \infty$, the optimal policy would be to never reconfigure, since this is the only policy for which the expected reward in (3) would be nonnegative. Again, however, the point (i.e., the smallest value of $\beta_{\min}$) at which this policy becomes optimal is not easy to determine, as it depends on the transition probabilities of the underlying Markov chain.

Consider an ergodic, discrete-space discrete-time Markov process with rewards and a set of alternatives per state that affect the probabilities and rewards governing the process. Howard's *policy-iteration* algorithm [3] can be used to obtain a policy that maximizes the long-term reward in (3) for such a process. Initially, an arbitrary policy is specified from which all state transition rates are determined. The algorithm then enters its basic iteration cycle, which consists of two stages. The first stage is the *value-determination operation*, which evaluates the current policy. In the second stage, the *policy-improvement routine* uses a set of criteria to modify the decisions at each state and obtain a new policy with a higher reward than the original policy. This new policy is used as the starting point for the next iteration. The cycle continues until the policies in two successive iterations are identical. At this point, the algorithm has converged, and the final policy is guaranteed to be optimal with respect to maximizing the reward in (3).

A difficulty in applying the policy-iteration algorithm to the Markov process $\mathcal{M}$ is that its running time per iteration is dominated by the complexity of solving a number of linear equations on the order of the number of states in the Markov chain. Even if we restrict the elements of traffic matrix $\mathbf{T}$ to be integers [2] and impose an upper bound on the values they can take, the potential number of states $(\mathcal{R}, \mathbf{T})$ is so large that the policy-iteration algorithm cannot be directly applied to anything but networks of trivial size. In the next section, we show how to overcome this problem by making some simplifying assumptions that will allow us to set up a new Markov process whose state space is manageable.

B. Alternative Formulation

Consider a network in state $(\mathcal{R}, \mathbf{T})$, and a new traffic matrix $\mathbf{T}'$ for which the WLA obtained with the GLPT algorithm is $\mathcal{R}'$. A closer examination of the reward function in (3) reveals that the immediate reward acquired when the network makes a transition does not depend on the actual values of the traffic elements or the actual WLAs involved, but only on the values of the DLBs $\phi(\mathcal{R}, \mathbf{T}')$ and $\phi(\mathcal{R}', \mathbf{T}')$ and the distance $\mathcal{D}(\mathcal{R}, \mathcal{R}')$. Thus, we make the simplifying assumption that the decision to reconfigure will also depend on the DLBs and the distance only. This is a reasonable assumption, since it is the DLB, not the actual traffic matrix or WLA, that determine the efficiency of the network in satisfying the offered load. Similarly, it is the number of retunings that determines the reconfiguration cost, not the actual WLAs involved.

Based on these observations, we now introduce a new process embedded, as Markov process $\mathcal{M}$, at instants when the traffic matrix changes, as illustrated in Fig. 2. The state of this process is defined to be the tuple (ϕ, D), where ϕ is the DLB achieved by the current WLA with respect to the current traffic matrix and D is the number of retunings required if the network were to reconfigure. Transitions in the new process have the Markovian property, since they are due to changes in the traffic matrix that, in turn, are Markovian. However, as defined, the process is a continuous-state process since, in general, the DLB ϕ is a real

[2] If the elements of $\mathbf{T}$ are real numbers, then $\mathcal{M}$ becomes a continuous-state process and the policy-iteration algorithm cannot be applied.

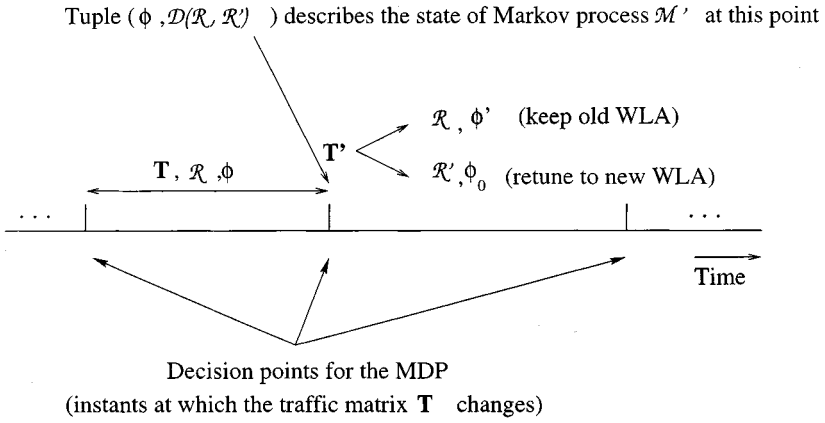

Fig. 2. State of the new Markov process $\mathcal{M}'$.

number. In order to apply Howard's policy-iteration algorithm, we need a discrete-state process. We obtain such a process by using discrete values for random variable ϕ as follows.

By definition [refer to (1)], the DLB ϕ can take any real value between zero and $C-1$, where C is the number of channels in the network.[3] We now divide the interval $[0, C-1]$ into a number $K+1$ of nonoverlapping intervals $[\phi_0^{(l)}, \phi_0^{(u)}), [\phi_1^{(l)}, \phi_1^{(u)}), \ldots, [\phi_K^{(l)}, \phi_K^{(u)}]$, where $\phi_k^{(l)}$ and $\phi_k^{(u)}$ are the lower and upper values of interval k, $k = 0, \ldots, K$, and $\phi_k^{(l)} < \phi_k^{(u)}$, $\phi_0^{(l)} = 0$, $\phi_k^{(u)} = \phi_{k+1}^{(l)}$, and $\phi_K^{(u)} = C-1$. Let ϕ_k denote the midpoint of interval k. We now define a new discrete-state process $\mathcal{M}'$ with state (ϕ_k, D). We will use state (ϕ_k, D) to represent any state (ϕ, D) of the continuous-state process such that $\phi_k^{(l)} \leq \phi < \phi_k^{(u)}$. Clearly, the larger the number K of intervals, the better the approximation.

Before we proceed, we make one further refinement to the new discrete-state process $\mathcal{M}'$. We note that the GLPT algorithm in [1] is an approximation algorithm for the load balancing problem, and it guarantees that the DLB of the WLA obtained using the algorithm will never be more than 50% away from the degree of load balancing of the optimal WLA. The importance of this result is as follows. Consider a network in which the traffic matrix changes in such a way that the current WLA provides a DLB ϕ for the new traffic matrix such that $\phi < 0.5$. Based on the guarantee provided by algorithm GLPT, we can safely assume that the load is well balanced and avoid a reconfiguration. This is because the network will incur a cost for reconfiguring, without any assurance that the new DLB will be less than ϕ. Therefore, we choose to let $\phi_0^{(u)} = 0.5$, and therefore the midpoint for the first interval is $\phi_0 = 0.25$. We will call any state (ϕ_0, D) a *balanced* state since the offered load is balanced within the guarantees of the GLPT algorithm.

We now specify decision alternatives, as well as reward and cost functions, associated with each transition in the new process $\mathcal{M}'$. Consider a network in state (ϕ_k, D). At the instant

[3]The value $\phi = 0$ is achieved when the load is perfectly balanced across the C channels, in which case the expression on the right-hand side of (1) becomes equal to $\sum_{i=1}^{N}\sum_{j=1}^{N} t_{ij}/C$. The value $\phi = C-1$ corresponds to the worst case scenario where one channel carries all the traffic; in this case, the right-hand side of (1) becomes equal to $\sum_{i=1}^{N}\sum_{j=1}^{N} t_{ij}$.

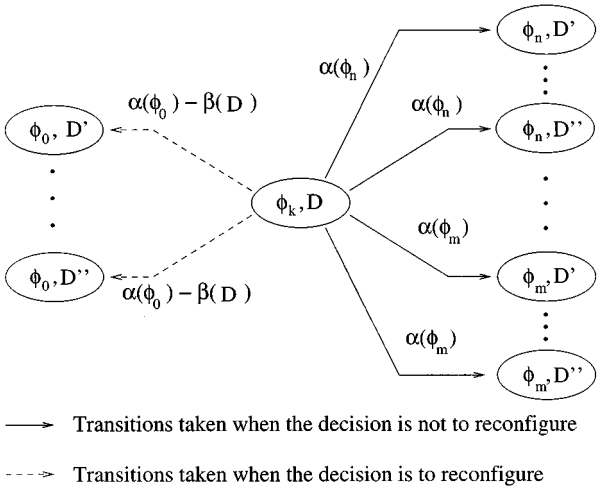

Fig. 3. Transitions and rewards out of state (ϕ_k, D) of process $\mathcal{M}'$ under the two decision alternatives (Note: the labels along the transitions represent rewards, *not* transition probabilities).

the traffic matrix changes, the network has two options. It may maintain the current WLA, in which case it will make a transition into state (ϕ_l, D'), where ϕ_l is the DLB of the current WLA with respect to the new traffic matrix and D' is the new distance. Or, it will reconfigure into a new WLA. In the latter case, the network will move into state (ϕ_0, D''), since its new DLB is guaranteed to be less than 0.5. When the network makes a transition into state $(\phi_l, D'), l \geq 0$, it acquires an immediate expected reward, which is equal to $\alpha(\phi_l)$. In addition, if (ϕ_l, D') is a balanced state (i.e., if $l = 0$), a reconfiguration cost equal to $\beta(D)$ is incurred.

The transitions out of state (ϕ_k, D) and the corresponding rewards are illustrated in Fig. 3. If the decision of the policy is not to reconfigure, then the process will take one of the transitions indicated by the solid arrows in Fig. 3. Since the network does not incur any reconfiguration cost, the immediate reward acquired is a function of the new DLB in the new state. If, on the other hand, the decision is to reconfigure, the transition out of state (ϕ_k, D) will always take the network to a balanced state

with a DLB equal to ϕ_0. These transitions are shown in dotted lines in Fig. 3. A reconfiguration cost is incurred in this case, making the immediate reward equal to $\alpha(\phi_0) - \beta(D)$.

The new process $\mathcal{M}'$ is a discrete-space discrete-time Markov process with rewards and two alternatives per state, and we can use the policy-iteration algorithm [3] to obtain an optimal policy off-line and cache its decisions. The optimal policy decisions can then be applied to a real network environment in the following way (refer also to Fig. 2). Consider a network with traffic matrix $\mathbf{T}$ operating under WLA $\mathcal{R}$. Let $\mathbf{T}'$ be the new traffic matrix and $\mathcal{R}'$ be the WLA constructed by algorithm GLPT [1], with $\mathcal{R}$ and $\mathbf{T}'$ as inputs. Let also $D = \mathcal{D}(\mathcal{R}, \mathcal{R}')$ be the number of receivers that need to be retuned to obtain WLA $\mathcal{R}'$ from WLA $\mathcal{R}$. To determine whether the network should reconfigure to the new WLA $\mathcal{R}'$, let $\phi(\mathcal{R}, \mathbf{T})$ be the current DLB for the network, and suppose that $\phi(\mathcal{R}, \mathbf{T})$ falls within the kth interval, $0 \leq k \leq K$. By definition of the Markov process $\mathcal{M}'$, the current network state is modeled by state (ϕ_k, D) of this process. If, under the optimal policy, the decision associated with this state is to reconfigure, then the network must make a transition to the new WLA $\mathcal{R}'$; otherwise, the network will continue operating under the current WLA $\mathcal{R}$.

We note that the discrete-space Markov process (ϕ_k, D) is an approximation of the continuous-space process (ϕ, D), since, as discussed above, in general the DLB ϕ is a real number between zero and $C-1$. We also note that as the number of intervals $K \rightarrow \infty$, the discrete-state process approaches the continuous-state one. Therefore, we expect that as the number of intervals K increases, the accuracy of the approximation will also increase, and the decisions of the optimal policy obtained through the process (ϕ_k, D) will "converge." This issue will be discussed in more detail in Section V, where numerical results to be presented will show that the decisions of the optimal policy "converge" for relatively small values of K. This is an important observation since the size of the state space of Markov process $\mathcal{M}'$ increases exponentially with K. By using a relatively small value for K, we can keep the state space of the process to a reasonable size, making it possible to apply the policy-iteration algorithm [3].

IV. The Transition Phase and Retuning Strategies

In this section, we assume that a decision to reconfigure the network has been made according to the optimal policy described in the previous section, and that the new WLA $\mathcal{R}'$ has been obtained through the GLPT algorithm [1]. We are, therefore, concerned with the *transition phase*, during which the actual retuning of receivers that takes the network from the current WLA $\mathcal{R}$ to the new WLA $\mathcal{R}'$ takes place. Clearly, in order to complete the reconfiguration, a number of receivers equal to $D(\mathcal{R}, \mathcal{R}')$ must be retuned. A *retuning strategy* determines when, and in what sequence, these receivers are taken off-line for retuning.

While the receiver of, say, node j is being retuned to a new wavelength, it cannot receive data, and thus, any packets sent to node j are lost. If, on the other hand, the network nodes are aware that node j is in the process of retuning its receiver, they can refrain from transmitting packets to it. In this case, packets destined to node j will experience longer delays while waiting for the node to become ready for receiving again. Moreover, packets for j arriving to the various transmitters during this time cannot be serviced and may cause buffer overflows. This increase in delay and/or packet loss during the transition phase is the penalty incurred for reconfiguring the network. Once all receivers have been retuned to their new wavelengths, the network has completed its reconfiguration and normal operation will resume until a new reconfiguration is required.

There is a wide range of strategies for retuning the receivers, mainly differing in the tradeoff between the length of the transition period and the portion of the network that becomes unavailable during this period (see [6] for a discussion of similar issues in multihop networks). One extreme approach would be to simultaneously retune all the receivers that are assigned new channels under $\mathcal{R}'$. The duration of the transition phase is minimized under this approach (it becomes equal to the receiver tuning latency), but a significant fraction of the network may be unusable during this time. At the other extreme, a strategy that retunes one receiver at a time minimizes the portion of the network unavailable at any given instant during the transition phase, but it maximizes the length of this phase (which now becomes equal to the receiver tuning latency times the distance $\mathcal{D}(\mathcal{R}, \mathcal{R}')$). Between these two ends of the spectrum lie a range of strategies in which groups of two or more receivers are retuned simultaneously.

Let $S, \mid S \mid = \mathcal{D}(\mathcal{R}, \mathcal{R}')$, be the set of receivers that need to be switched to new wavelengths in order to reconfigure the network to WLA $\mathcal{R}'$. In the most general sense, a retuning strategy is defined as a partition of S into M, $1 \leq M \leq \mid S \mid$, subsets of receivers g_m, such that time t_m is associated with subset g_m. Under this definition, a subset g_m represents a group of receivers that are simultaneously taken off-line for retuning. The retuning of group g_m starts at time t_m and lasts for a period of time equal to the receiver retuning latency, after which the receivers in the group become operational again. Without loss of generality, we assume that $t_m < t_{m+1}$.

We distinguish between *overlapping* and *nonoverlapping* strategies. In the former, the retuning periods of two distinct groups of receivers are allowed to overlap, while in the latter a group g_m does not start retuning until after group g_{m-1} has completed its retuning. Since it does not appear that overlapping strategies offer any advantages over nonoverlapping ones (in fact, they may complicate the management of the transition phase), we only consider nonoverlapping strategies here. However, the number of partitions of the set S can potentially be very large, making it impractical to study all possible nonoverlapping strategies. Hence, we restrict ourselves to the class of parameterized nonoverlapping strategies described in Fig. 4. Specifically, a family of strategies is characterized by a specific value for parameter L, which represents the maximum number of receivers in each group g_m. When a group of L receivers has completed its retuning, another group of L receivers is immediately scheduled for retuning (unless there are less than L receivers left to be retuned); this process is repeated until all receivers in S have switched to their new wavelengths under WLA $\mathcal{R}'$. We note that a wide range of strategies can be obtained for the different values of L, $1 \leq L \leq N$. Furthermore, by varying the value of parameter L, we hope to

Parameterized non-overlapping greedy retuning strategy
Input: Parameter $1 \leq L \leq N$, set $S(\mathcal{R}, \mathcal{R}')$ of receivers that need to be retuned

1. Repeat while set S is non-empty
2. At most L receivers from set S are selected (e.g., the ones with the lowest addresses) and taken off-line for retuning
3. Each node in the network removes packets addressed to these receivers from its buffers (these packets are lost since they are queued for the wrong (old) wavelength)
4. Each node in the network buffers new packets destined to the receivers being retuned into the queue corresponding to the new wavelength, but does not transmit these packets yet
5. While the L receivers are being retuned, the network nodes transmit traffic only to the remaining $N - L$ receivers
6. When retuning is complete, the L receivers are removed from set S
7. end

Fig. 4. A class of parameterized retuning strategies.

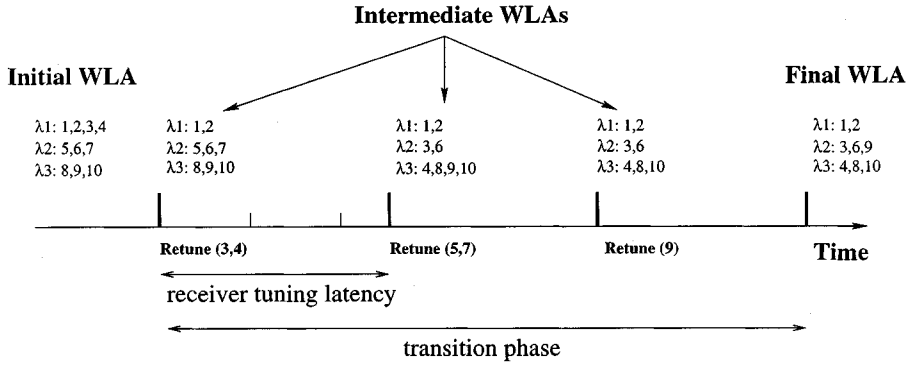

Fig. 5. Steps of a nonoverlapping retuning strategy with $L = 2$ for a network with $N = 10, C = 3$.

determine whether there exists a tradeoff between the number of receivers that can be scheduled to retune at the same time and the associated packet losses during the transition phase.

The steps taken by such a strategy during the transition phase are illustrated in Fig. 5. It is assumed that a network with $N = 10$ nodes and $C = 3$ channels is to be reconfigured from an initial to a final WLA (as shown in the figure) and that $L = 2$. In this case, five receivers need to be retuned, and $S = \{3, 4, 5, 7, 9\}$. In the first step, receivers 3 and 4 are taken off-line for retuning. When retuning is complete, receivers 3 and 4 become operational again, and receivers 5 and 7 are selected for retuning. Finally, receiver 9 is retuned, at which time the transition to the new WLA is complete. Thus, in this example, the transition phase takes time equal to three times the receiver retuning latency. Note that whenever $L < \mid S \mid$, i.e., for any strategy that does not permit all receivers in S to retune simultaneously, the network's WLA undergoes a series of transformations that begins with the initial WLA $\mathcal{R}$ and ends with the new balanced WLA $\mathcal{R}'$. The WLAs between the initial and final ones (called *intermediate* WLAs in Fig. 5) do not contain all the receivers in the network, just those that are not retuning in the current step. By using small, incremental steps during the transition phase, rather than shutting down a large part of the network to simultaneously retune all the receivers in S, we hope to minimize the negative effects on network performance while at the same time keeping the length of the transition phase relatively small.

V. Numerical Results

We now study the reconfiguration policies and retuning strategies introduced in previous sections using both analytical and simulation techniques.

A. Optimal Reconfiguration Policies

In this section, we demonstrate the properties of the optimal policies obtained by applying the policy-iteration algorithm [3] to the Markov decision process developed in the previous section. We also show how the optimal policy is affected by the choice of reward and cost functions, and we compare the long-term reward acquired by the network when the optimal policy is employed to the reward acquired by other policies. All the results presented in this section are for the approximate Markov process $\mathcal{M}'$ with state space (ϕ_k, D).

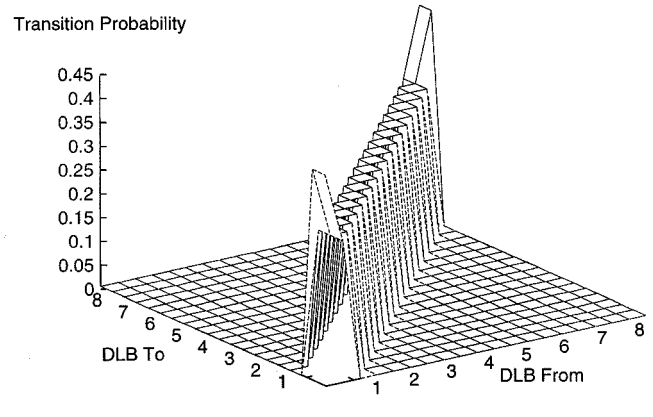

Fig. 6. Near-neighbor model.

In this study, we consider a *near-neighbor* traffic model. In other words, we make the assumption that if the network currently operates with a DLB equal to ϕ_k and no reconfiguration occurs, the next transition is more likely to take the network to a state with the same DLB or its two nearest neighbors ϕ_{k-1} and ϕ_{k+1} than to a DLB further away from ϕ_k. Specifically, we assume (4), shown at the bottom of the page. This traffic model is illustrated in Fig. 6, which plots the conditional probability $P[\phi_l \mid \phi_k]$ that the next DLB will be ϕ_l, given that the current DLB is ϕ_k, for $K = 20$ intervals. The near-neighbor model captures the behavior of networks in which the traffic matrix $\mathbf{T}$changes slowly over time and abrupt changes in the traffic pattern have a low probability of occurring. As a result, a network that is perfectly balanced will not immediately become highly unbalanced, and vice versa. We consider a different client-server model of communication in the next section (refer to Fig. 21) to illustrate that our approach can be used under a wide range of traffic patterns.

Given the probabilities in (4), we let the conditional transition probability, *when no reconfiguration occurs*, from state (ϕ_k, D) to state (ϕ_l, D') be equal to

$$P[(\phi_l, D') \mid (\phi_k, D)] = P[\phi_l \mid \phi_k] p_{D'} \tag{5}$$

where $p_{D'}$ is the probability that D' retunings will be required in the next reconfiguration. The probabilities p_D were measured experimentally, by running a large number of simulations using the near-neighbor model and recording the number of retunings needed at each reconfiguration instant. We also observed that the probability that random variable D takes on a particular value is independent of the DLB ϕ_k; thus (5).

We note that we need to obtain two different transition probabilities out of each state [3], one for each of the two possible options: the do-not-reconfigure option and the reconfigure option. The above discussion explains how to obtain the transition probability matrix for the do-not-reconfigure option. The transition probability matrix for the reconfigure option is easy to determine since we know that regardless of the value ϕ_k of the current state, the next state will always be a balanced state, i.e., its DLB will be ϕ_0. The individual transition probabilities from a state (ϕ_k, D) to a state (ϕ_l, D') are then obtained by making the same assumption that the distribution of D is independent of the DLB ϕ_k. Therefore, the transition probabilities under the reconfigure option are

$$P[(\phi_l, D') \mid (\phi_k, D)] = \begin{cases} p_{D'}, & l = 0 \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

1) Convergence of the Optimal Policy: Let us first consider the following reward and cost functions:

$$\alpha[(\phi_k, D)] = \frac{A}{1+\phi_k}, \quad \beta(D) = BD \tag{7}$$

where A and B are weights assigned to the rewards and costs. These reward and cost functions can reflect performance measures such as throughput, delay, packet loss, or the control resources involved in receiver retuning. For example, a reward function of the form $A/(1+\phi)$ may, depending on the value of parameter A, capture either the throughput or average packet delay experienced while the network operates with a DLB equal to ϕ. On the other hand, using a cost that is proportional to the number $D = \mathcal{D}(\mathcal{R}, \mathcal{R}')$ of retunings (i.e., $\beta(\mathcal{D}(\mathcal{R}, \mathcal{R}')) = BD$) can account for the control requirements for retuning the receivers, or for the data loss incurred during reconfiguration. Furthermore, parameter B can be chosen based on which of the retuning strategies discussed in Section IV is employed. Thus, network designers can select in a unified fashion appropriate rewards and costs to achieve the desired balance among the various performance criteria of interest.

We apply Howard's algorithm [3] to a network with $N = 20$ nodes and $C = 5$ wavelengths with a near-neighbor traffic model similar to the one shown in Fig. 6. Our objective is to study the effect that the number of intervals K in the range [0, $C-1$] of possible values of DLB ϕ has on the decisions of the optimal policy. As we mentioned in Section III-B, we expect the decisions of the optimal policy to "converge" as $K \to \infty$. More formally, let φ be a real number such that $0 \le \varphi \le C-1$, and let k_K be the interval in which φ falls when the total number of intervals is K. Also let $d^{(K)}[(\phi_{k_K}, D)]$ be the decision of the optimal policy for state (ϕ_{k_K}, D) of Markov process $\mathcal{M}'$ when

$$P[\phi_l \mid \phi_k] = \begin{cases} 0.3, & k = 1, \ldots, K-1, \ \ l = k-1, k, k+1 \\ \dfrac{0.1}{(K-2)}, & k = 1, \ldots, K-1, \ \ l \neq k-1, k, k+1 \\ 0.45, & k = 0, \ \ l = 1 \text{ or } \ k = K, \ \ l = K-1 \\ \dfrac{0.1}{(K-2)}, & k = 0, \ \ l = 2, \ldots, K \text{ or } \ k = K, \ \ l = 0, \ldots, K-2. \end{cases} \tag{4}$$

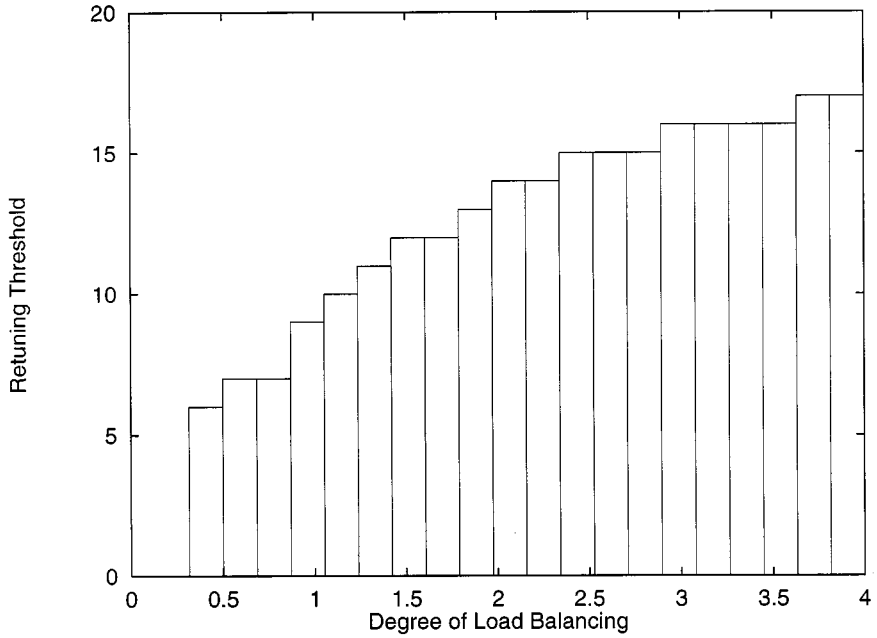

Fig. 7. Optimal policy decisions for $N = 20$, $C = 5$, $K = 20$, $A = 30$, and $B = 1$.

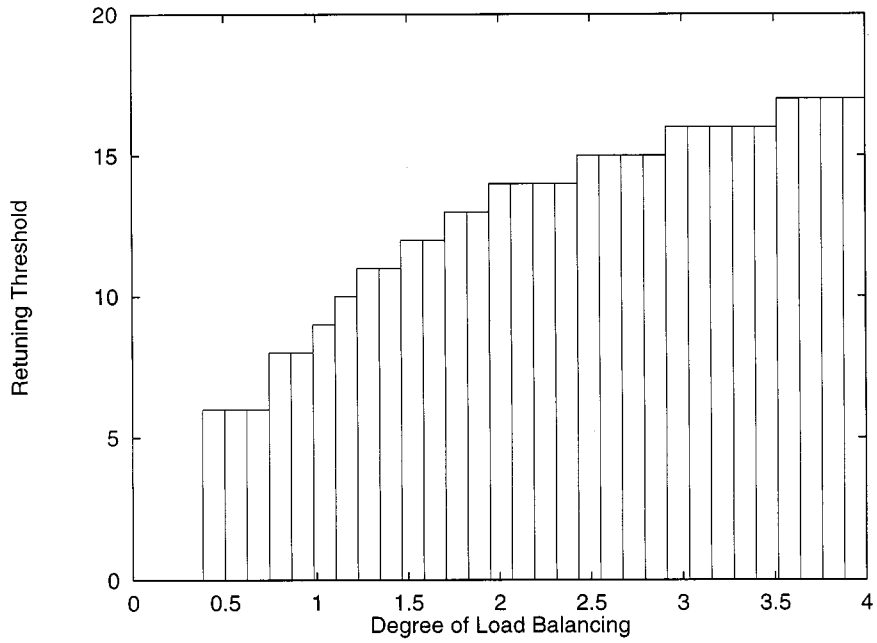

Fig. 8. Optimal policy decisions for $N = 20$, $C = 5$, $K = 30$, $A = 30$, and $B = 1$.

the number of intervals is K. We will say that the decisions of the optimal policy converge if

$$\lim_{K\to\infty} d^{(K)}[(\phi_{k_K}, D)] = d[(\varphi, D)] \quad \forall \varphi, D. \tag{8}$$

In Figs. 7–9, we plot the decisions of the optimal policy for the 20-node five-wavelength network with a near-neighbor traffic model and for three different values of K; the weights used in the functions (7) were set to $A = 30$ and $B = 1$. Fig. 7 corresponds to the optimal policy for $K = 20$ intervals, while in Figs. 8 and 9, we increase K to 30 and 40, respectively. The histograms shown in Figs. 7–9, as well as in other figures in this section, should be interpreted as follows. In each figure, the x-axis represents the DLB ϕ_k (with a number of intervals equal to the corresponding value of K), while the y-axis represents the possible values of D. The vertical bar at a particular DLB value ϕ_k has a height equal to D_k^{thr} such that

$$d^{(K)}[(\phi_k, D)] = \begin{cases} \text{reconfigure}, & D \le D_k^{\text{thr}} \\ \text{do not reconfigure}, & D > D_k^{\text{thr}} \end{cases} \tag{9}$$

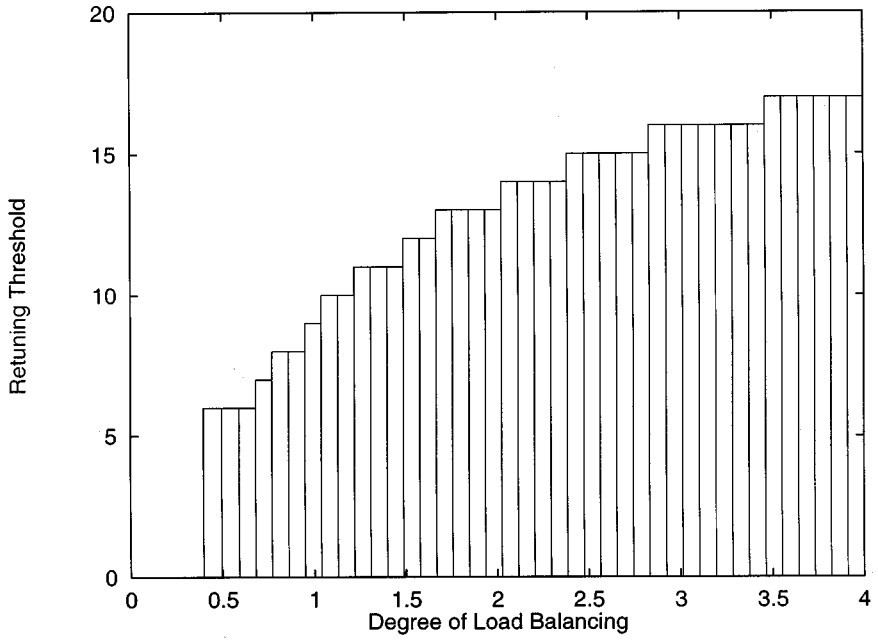

Fig. 9. Optimal policy decisions for $N = 20, C = 5, K = 40, A = 30, B = 1$.

In other words, for each value of ϕ_k, there exists a *retuning threshold* value D_k^{thr} such that the decision is to reconfigure when the number of receivers to be retuned is less than D_k^{thr}, and not to reconfigure if it is greater than D_k^{thr}. Since the optimal policy had similar behavior for all the different reward and cost functions we considered, its decisions will be plotted as a histogram similar to those in Figs. 7–9.[4]

As we can see in Figs. 7–9, the decisions of the optimal policy do converge [in the sense of (8)] as K increases. For instance, let us consider a DLB of one, which falls in the fourth interval when $K = 20$ (in Fig. 7), the sixth interval when $K = 30$ (in Fig. 8), and the seventh interval when $K = 40$ (in Fig. 9). In all three cases, the retuning threshold is equal to nine for these intervals; therefore, the decisions of the optimal policy for the three values of K are the same. On the other hand, for a DLB of two, the retuning threshold is 14 in Fig. 7 it drops to 13 in Fig. 8, the same as in Fig. 9. In other words, for a DLB of two, the decisions of the optimal policy are different when $K = 20$ than when $K = 30$ or 40 (in the former case, the decision is to reconfigure as long as the number of retunings is at most 14, while in the latter the decision is to reconfigure only when the number of retunings is at most 13). But the important observation is that the policy decisions do not change when the number K of intervals increases from 30 to 40, indicating convergence. In fact, there are no changes in the optimal policy for values of K greater than 40 (not shown here). We have observed similar behavior for a wide range of values for weights A and B, for different network sizes, and for other reward and cost functions. These results indicate that a relatively small number of intervals is sufficient for obtaining an optimal policy.

Another important observation from Figs. 7–9 is that the retuning threshold increases with the DLB values. This behavior can be explained by noting that because of the near-neighbor distribution (refer to Fig. 6), when the network operates at states with high DLB values, it will tend to remain at states with high DLB values. Since the reward is inversely proportional to the DLB value, the network incurs small rewards by making transitions between such states. Therefore, the optimal policy is such that the network decides to reconfigure even when there is a large number of receivers to be retuned. By doing so, the network pays a high cost, which, however, is offset by the fact that the network makes a transition to the balanced state with a low DLB, reaping a high reward. On the other hand, when the network is at states with low DLB, it also tends to remain at such states where it obtains high rewards. Therefore, the network is less inclined to incur a high reconfiguration cost, and the retuning threshold for these states is lower.

2) The Effect of Reward and Cost Functions: In Figs. 10–12, we apply Howard's algorithm to a network with $N = 100$ nodes and $C = 10$ wavelengths, operating under a near-neighbor model similar to the one shown in Fig. 6. For this network we used $K = 20$ intervals, and we varied weights A and B in the reward and cost functions in (7) to study their effect on the optimal policy. Specifically, we let $B = 1$, and we varied A from 20 (in Fig. 10) to 35 (in Fig. 11) to 50 (in Fig. 12). We first observe that the optimal policy is again a threshold policy for each value ϕ_k of the DLB. However, as A increases, we see that the retuning threshold associated with each DLB value also increases. This behavior of the optimal policy is in agreement with intuition since, by increasing A, we increase the reward obtained by taking the

[4]That the optimal policy was found to be a threshold policy (with a possibly different retuning threshold) for each value of ϕ_k can be explained by the fact that we only consider cost functions that are nondecreasing functions of random variable D. As a result, if the decision of the optimal policy for a state (ϕ_k, D_1) is not to reconfigure, intuitively one expects the decision for state (ϕ_k, D_2), where $D_2 > D_1$, to also be not to reconfigure, since the reconfiguration cost $\beta(D_2)$ for the latter state would be at least as large as the reconfiguration cost $\beta(D_1)$ for the former

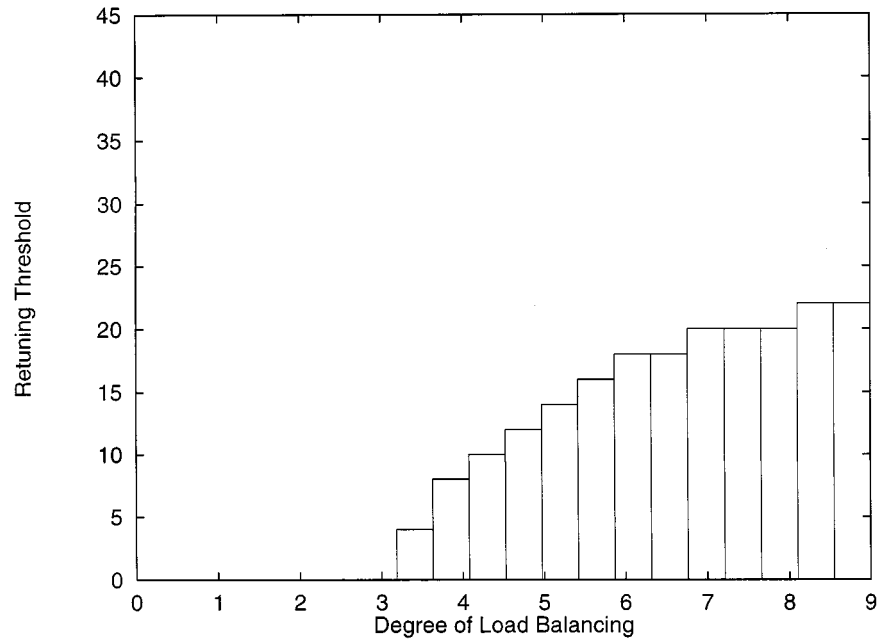

Fig. 10. Optimal policy decisions for $N = 100, C = 10, K = 20, A = 20$, and $B = 1$.

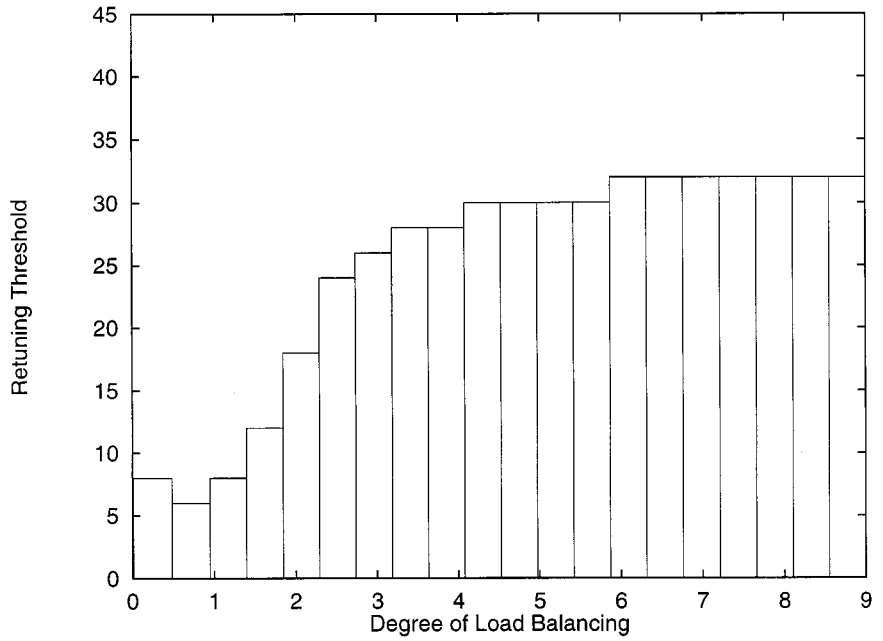

Fig. 11. Optimal policy decisions for $N = 100, C = 10, K = 20, A = 35$, and $B = 1$.

network to a balanced state relative to the cost of reconfiguration, making reconfigurations more attractive. Similarly, if we keep A constant and increase B (a case not shown here), reconfiguring the network becomes less desirable, and thus the retuning threshold associated with each DLB value decreases. Overall, in our study, we have found that one can obtain a wide variety of policies by varying the values of weights A and B. It is up to the network operator to decide what values to use, and thus to make the network more or less sensitive to traffic changes (i.e., more or less likely to reconfigure).

We now proceed to study the effect of different reward and cost functions. One performance measure of interest is the probability of unnecessary reconfigurations. By making $\beta_{\min}$ and $\beta_{\max}$ large, and letting $\alpha(\cdot)$ be a slowly decreasing function as ϕ increases, minimizing the probability of unnecessary reconfigurations becomes equivalent to maximizing (3). Similarly, the objective to minimize the probability that the portion of the network that becomes unavailable due to reconfiguration is greater than a certain threshold $D_{\max}$ can be achieved by letting $\beta_{\min}$ be small, letting $\beta_{\max}$ be large, and selecting $\beta(\cdot)$

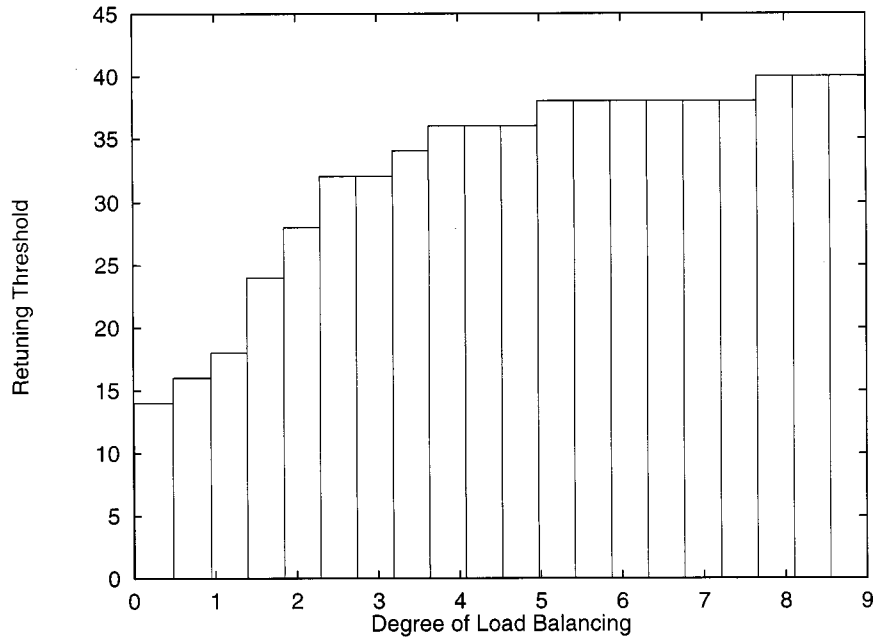

Fig. 12. Optimal policy decisions for $N = 100$, $C = 10$, $K = 20$, $A = 50$, and $B = 1$.

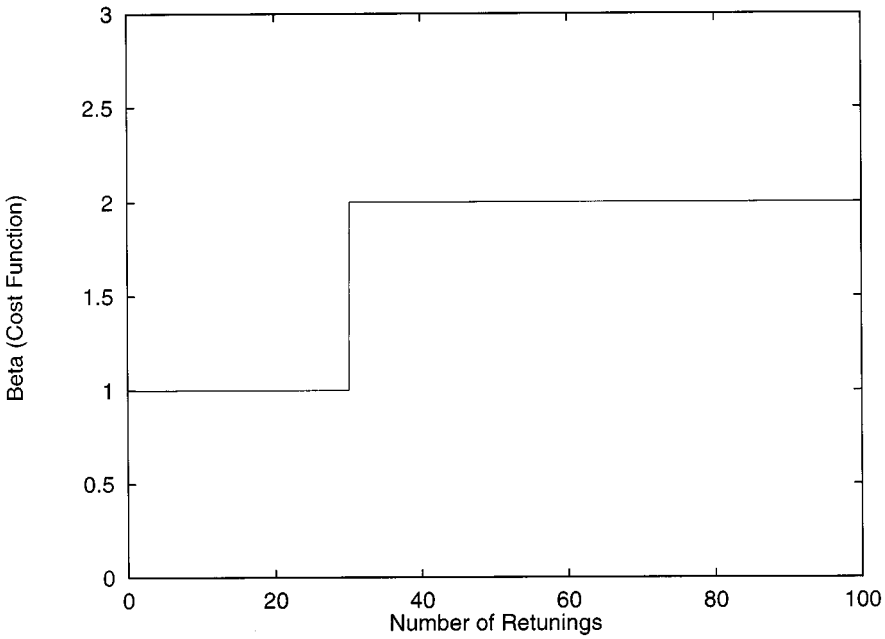

Fig. 13. Cost function $\beta(D)$ used for the policy shown in Fig. 14.

to be a function as shown in Fig. 13 (where $D_{\max} = 30$). We now consider the latter cost function $\beta(D)$ plotted in Fig. 13, while the reward function is as in (7) with $A = 50$. In Fig. 14, we show the decisions of the optimal policy for a network with $N = 100$, $C = 10$, and a near-neighbor traffic model when $K = 20$. As we can see, the retuning threshold never exceeds the value $D_{\max} = 30$. Therefore, the network will never reconfigure when the number of retunings is greater than 30, as expected.

Another important performance objective is to minimize the probability that the network will not be able to handle the offered traffic load. This is equivalent to minimizing the probability that the DLB increases beyond a maximum value $\phi_{\max}$. Let $\gamma_{\max}$ be the maximum traffic load (in packets per packet transmission time) that will ever be allowed into the network. By definition of the DLB in (1), the load offered to the dominant channel when the DLB is ϕ will be $(1+\phi)\gamma_{\max}/C$. Since each channel can clear at most one packet per packet time, we

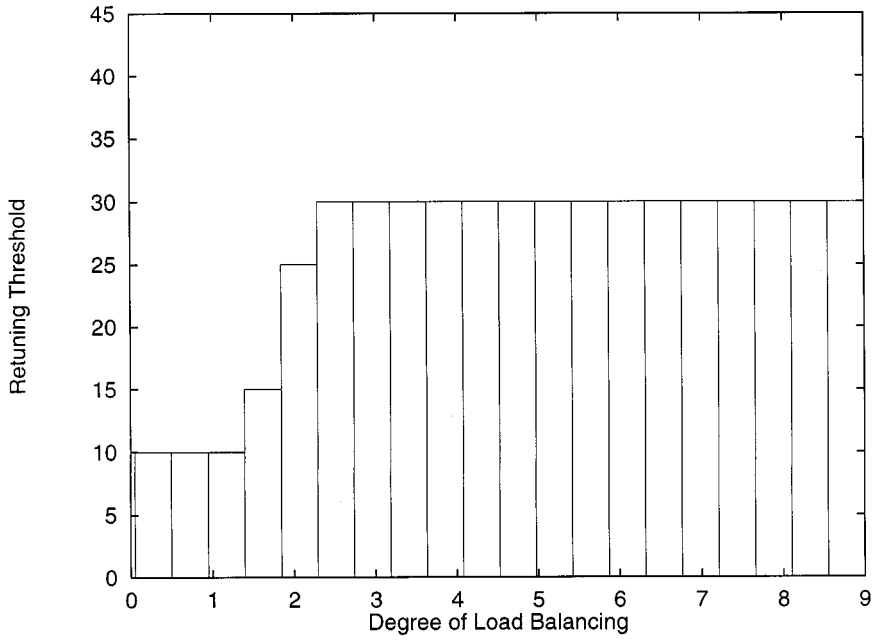

Fig. 14. Optimal policy decisions for $N = 100$, $C = 10$, $K = 20$, and $A = 50$.

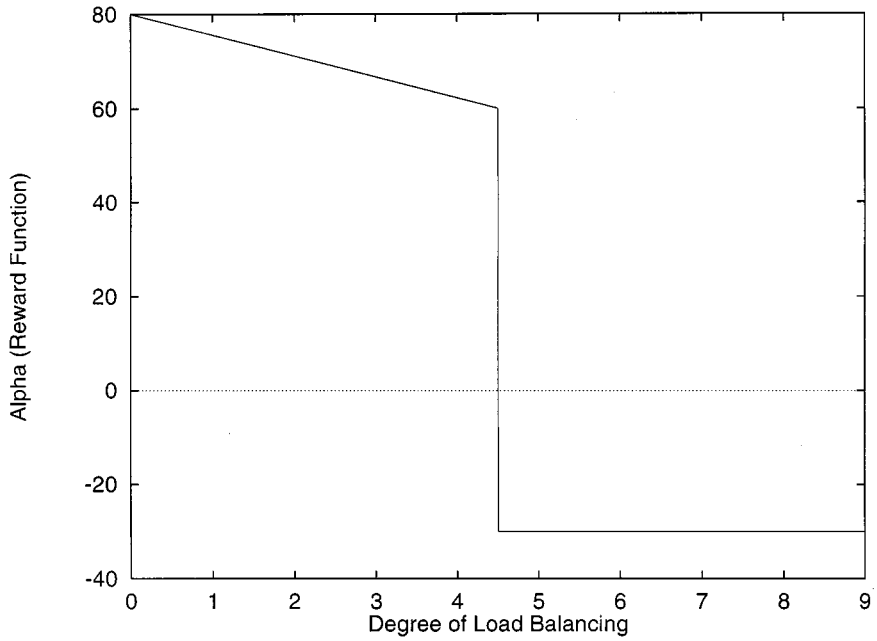

Fig. 15. Reward function $\alpha(\phi)$ used for the policy shown in Fig. 16.

have that $1 + \phi_{\max} \leq C/\gamma_{\max}$. Therefore, this objective can be achieved by selecting $\alpha(\cdot)$ a function, as shown in Fig. 15 (where $\phi_{\max} = 4.5$) and letting $\beta_{\max}$ be small. Thus, we also obtained the optimal policy for the same network as above, but with the reward function shown in Fig. 15; and a cost function $\beta(D) = BD$, with $B = 1$. The resulting policy is shown in Fig. 16, where we can see that the retuning threshold is 100 for DLB values greater than 4.5. Since the maximum number of receivers that will ever need to be retuned is $N - C = 90$[1], a retuning threshold equal to 100 means that the network will always reconfigure when the DLB becomes greater than 4.5. Thus, although the network is not prohibited from entering a state with a DLB value greater than 4.5, once doing so, in the very next transition the network will reconfigure and will enter the balanced state. Subsequently, because of the nature of the near-neighbor traffic model, the network will tend to stay at states with low DLB values. In effect, therefore, the probability that the network will be operating at states with DLB values

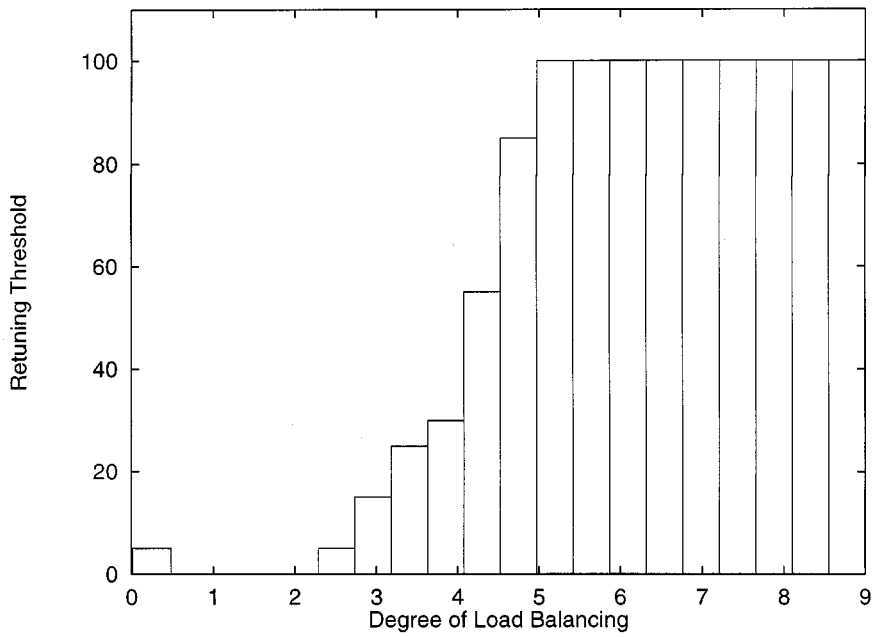

Fig. 16. Optimal policy decisions for $N = 100, C = 10, K = 20$, and $B = 1$.

greater than 4.5 is very small when the reward function in Fig. 15 is used.

3) Comparison to Threshold Policies: In this section, we compare the optimal policy against three classes of threshold-based policies.

1) *DLB-threshold policies.* There exists a threshold DLB value $\phi_{\max}$ such that if the system is about to make a transition into a state $(\phi_k, D), \phi_k > \phi_{\max}$, then the network will reconfigure and make a transition to a state with DLB ϕ_0, regardless of the reconfiguration cost involved. Otherwise, no reconfiguration occurs. This class of policies is not concerned with the reconfiguration cost incurred. Instead, it ensures that the traffic-carrying capacity of the network will never fall below the value $\gamma_{\min} = C/(1 + \phi_{\max})$.
2) *Retuning-threshold policies.* This class of policies is in a sense a "dual" of the previous one, in that decisions are based solely on the number of retunings involved in the reconfiguration, not on the DLB. Specifically, if the network is about to make a transition, then the network will reconfigure only if the number D of receivers that must be retuned is less than or equal to a threshold $D_{\max}$. If $D > D_{\max}$, no reconfiguration takes place. This class of policies ensures that the portion of the network that becomes unavailable due to reconfiguration never exceeds $D_{\max}$.
3) *Two-threshold policies.* This class of policies attempts to combine the objectives of the two classes of policies above. Specifically, there are two thresholds $\phi_{\max}$ and $D_{\max}$. If the system is about to make a transition into a state (ϕ_k, D), then the network will reconfigure if $\phi_k > \phi_{\max}$. Otherwise, if $\phi_k \leq \phi_{\max}$, the network will reconfigure if the number D of receivers that must be retuned is less than or equal to $D_{\max}$, and it will not reconfigure if $D > D_{\max}$. We note that if we let $D_{\max} = N - C$ (i.e., the maximum number of receiver that will ever need to be retuned [1]), these policies reduce to the class of DLB-threshold policies. Similarly, if we let $\phi_{\max} = C - 1$ (i.e., the DLB threshold is equal to the maximum DLB value), these policies become simple retuning threshold policies. Therefore, the two-threshold policies are the most general class of policies and include the DLB-threshold and retuning-threshold policies as special cases.

The DLB-threshold and the general two-threshold policies above define Markov processes that are *outside* the class of Markovian decision processes considered in Section III. In a Markovian decision process, there are several alternatives per state. But once an alternative has been selected for a state, then transitions from this state are always governed by the chosen alternative (refer also to Fig. 3). In a DLB-threshold policy, on the other hand, the alternative selected does not depend on the *current* state, but rather on the *next* state. Therefore, the system may select different alternatives when at a particular state, depending on what the next state is,[5] and similarly for the two-threshold policies. Since Howard's algorithm [3] is optimal only within the class of Markovian decision processes, it is possible that these threshold policies obtain rewards higher than the optimal policy determined by the algorithm. Retuning-threshold policies, however, are such that there is a unique alternative per state, so we expect them to perform no better than the optimal policy.[6]

[5]If the next state is one with a DLB less than the threshold, the alternative selected is not to reconfigure; otherwise the alternative selected is to reconfigure.

[6]As we have seen, the optimal policies are in fact threshold policies with a different retuning threshold for each DLB value. Therefore, the optimal policy will in general perform better than a retuning policy with the same threshold for all DLB values.

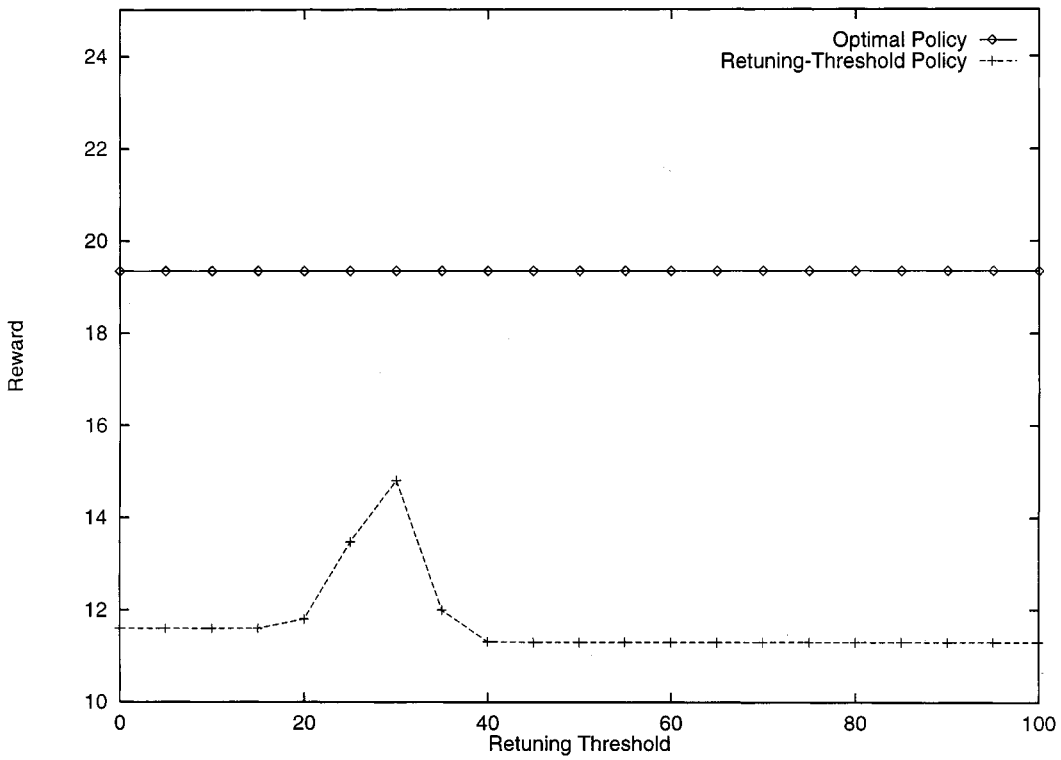

Fig. 17. Policy comparison, $N = 100, C = 10, K = 20, A = 50$, and $B = 1$.

All the results presented in this section are for a network with $N = 100$ nodes, $C = 10$ wavelengths, a near-neighbor traffic model, and $K = 20$ intervals. The reward and cost functions considered are those in (7). In Fig. 17, we compare the optimal policy obtained by Howard's algorithm [3] to a number of retuning-threshold policies. The figure plots the average long-term reward acquired by each of the policies against the retuning threshold $D_{\max}$. The horizontal line corresponds to the reward of the optimal policy, which, clearly, is independent of the retuning threshold. Each point of the second line in the figure corresponds to the reward of a retuning-threshold policy with the stated threshold value. As we can see, retuning-threshold policies obtain a reward that is significantly less than that of the optimal policy, as expected. Furthermore, the reward of retuning-threshold policies varies depending on the actual threshold used. Since the best threshold depends on system parameters such as the traffic patterns and the reward and cost functions and the associated weights, it is impossible to know the best threshold to use unless one experiments with a large number of threshold values.

In Fig. 18, we compare the optimal policy to a DLB-threshold policy and a number of two-threshold policies. For these results, we used $A = 50$ and $B = 1$ as the values for the weights in the reward and cost functions, respectively, in (7). This time, we plot the reward of each policy against the DLB threshold value; similar to Fig. 17, the optimal policy is independent of the DLB threshold, resulting in a horizontal line in Fig. 18. We also plot the reward of DLB-threshold policies with varying DLB thresholds and of a family of two-threshold policies. Each of the three plots of two-threshold policies corresponds to a different retuning threshold (namely, $D_{\max} = 40$, 32, and 24) and varying DLB thresholds. Also, recall that the DLB-threshold policy is equivalent to a two-threshold policy with a retuning threshold equal to $N - C = 90$.

The most interesting observation from Fig. 18 is that, for certain values of the DLB-threshold, the DLB-threshold policy and the two-threshold policy with retuning threshold $D_{\max} = 40$ achieve a higher reward than the optimal policy obtained through Howard's algorithm. This result is possible because, as we discussed earlier, the class of two-threshold policies is more general than the class of policies for which Howard's algorithm is optimal. On the other hand, we note that the reward of the DLB-threshold policy depends strongly on the DLB threshold used and that the reward of the two-threshold policies depends on the values of both thresholds. Although within a certain range of these values the threshold policies perform better than the optimal policy, the latter outperforms the former for most threshold values. Therefore, threshold selection is of crucial importance for the threshold policies, but searching through the threshold space can be expensive. The optimal policy, however, guarantees a high overall reward and is also simpler to implement since the network does not need to *look ahead* to the next state to decide whether or not to reconfigure.

Fig. 19 is similar to Fig. 18 in that we again compare the optimal policy against a DLB-threshold and two-threshold policies. For these experiments, however, we have used $A = 20$ and $B = 1$ in the reward and cost functions, respectively, of (7). As we can see, the reward of the optimal policy is strictly higher

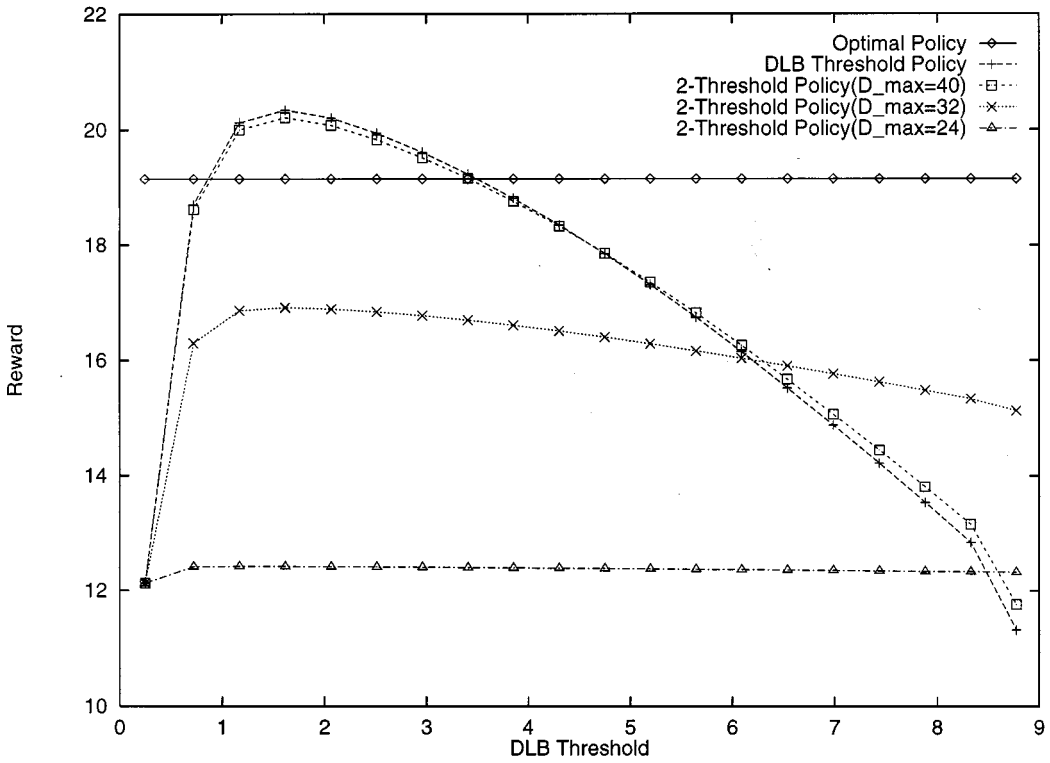

Fig. 18. Policy comparison, $N = 100$, $C = 10$, $K = 20$, $A = 50$, and $B = 1$.

than that of threshold policies across all possible threshold values. These results demonstrate that DLB- or two-threshold policies do not always perform better than the optimal policy, and their performance depends on the system parameters and/or the reward and cost functions. Furthermore, it is not possible to know ahead of time under what circumstances the threshold policies will achieve a high reward. Equally important, if the network's operating parameters change, threshold selection must be performed anew, since, for instance, the DLB threshold that maximizes the reward of the DLB-threshold policy in Fig. 18 results in very poor performance in Fig. 19, and vice versa.

Overall, the results presented in this section demonstrate that the optimal policy obtained through Howard's algorithm can successfully balance the two conflicting objectives, namely, the DLB and the number of retunings, and always achieves a high reward across the whole range of the network's operating parameters. We have also shown that by appropriately selecting the reward and cost functions, the optimal policy can be tailored to specific requirements set by the network designer. On the other hand, pure threshold policies, although they can sometimes achieve high reward, are less flexible, and they introduce an additional degree of complexity, namely, the problem of threshold selection.

B. Simulation Study

The objective of our simulation study is twofold. First, we want to demonstrate the benefits of dynamic reconfiguration in terms of specific performance measures such as packet delay and packet loss. Second, we want to evaluate the various retuning strategies of Section IV by studying the effect of parameter L in the algorithm shown in Fig. 4. To this end, we have developed a simulator of a WDM network that implements the reconfiguration policies and retuning strategies described earlier. In the following, we present the most important features of our simulator; for a detailed description, the reader is referred to [2].

Each channel in the network is modeled as an OC-48 (2.48 Gb/s) link. The traffic between nodes in the network follows a client-server model of communication. Each node is either a server or a client, but the number of servers is considerably smaller than the number of clients. Each client has two types of traffic sources. The first source creates the background traffic in the network by generating packets to other client nodes; the arrival rate of this traffic is relatively low. The second source models the communication of this client with its current server, and its arrival rate is significantly higher than that of the background source. A source, regardless of its type, is implemented following the bursty source model recommended by the ATM Forum for simulating VBR-rt traffic. The packet size in the network was taken to be equal to the length of an ATM cell. The nodes access the various channels using the HiPeR-ℓ MAC protocol [11], which in turn uses the algorithms in [10] to schedule packets for transmission.

We created variations in the traffic pattern by using the following technique. Initially, each client node is associated with one particular server. Periodically, the assignment of clients to

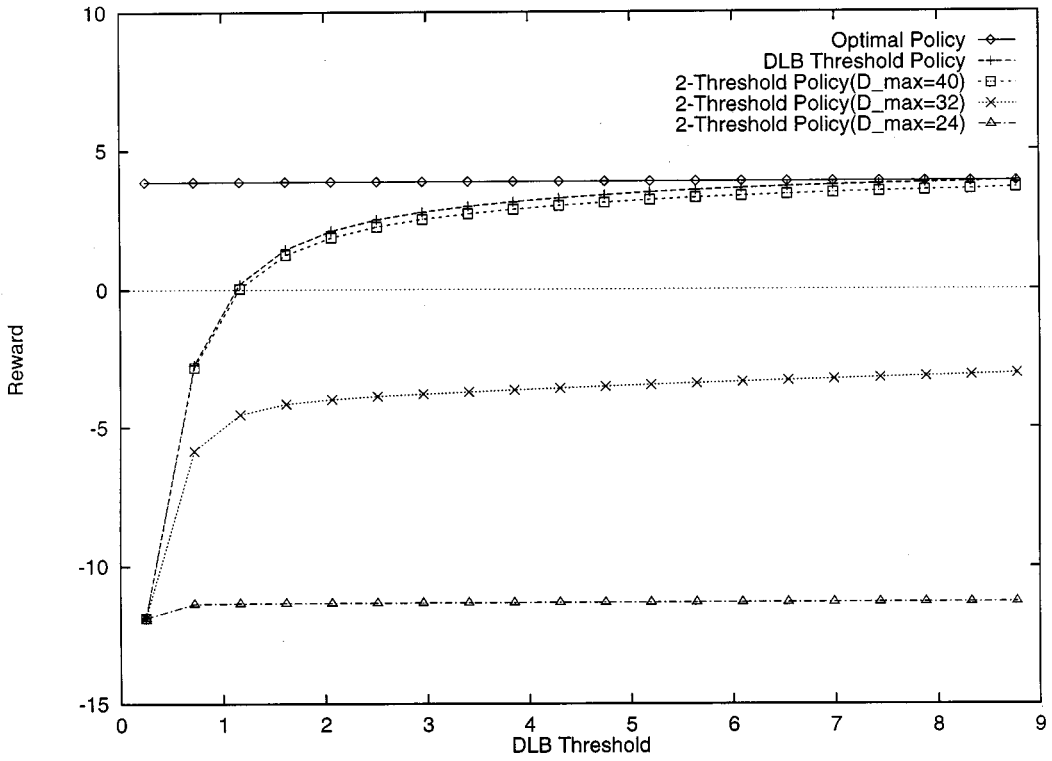

Fig. 19. Policy comparison, $N = 100$, $C = 10$, $K = 20$, $A = 20$, and $B = 1$.

servers is modified to reflect changes in the needs of the applications running at the client (for instance, a client may access a file server to edit a file and then a Web server to fetch a Web page). This change in client-server assignments creates a shift in the traffic pattern and may result in an unbalanced network where one or more channels carry a large fraction of the overall traffic. Network nodes accumulate traffic data for each source–destination pair over 2 ms intervals, using the in-band control packets of HiPeR-ℓ [11]. These data are used to compute the current DLB of the network and to determine whether reconfiguration is needed. As a result, successive reconfigurations are always spaced at least 2 ms apart. However, no necessary reconfigurations were ever delayed, since in our model each client-server assignment lasts for at least 20 ms.[7]

We collected experimental traffic data for the client-server communication model, from which we built a conditional distribution of DLB values for $K = 20$ intervals. This conditional distribution $P[\phi_k \mid \phi_l]$ is plotted in Fig. 20. As we can see, this distribution is quite different from the near-neighbor distribution of Fig. 6. Using the reward and cost functions in (7), the optimal policy obtained by Howard's algorithm is shown in Fig. 21. This optimal policy was used throughout this section.

For the purposes of this study, we consider a WDM network with $N = 70$ nodes, of which seven are servers and 63 are clients. There are $C = 5$ channels in the network with an aggregate capacity of 12.4 Gb/s. The fast tunable transmitters are assumed to take two packet times (342 ns for 53-byte packets and OC-48 speeds) to switch wavelengths. Each simulation run terminates when a total of 10^7 packets are successfully received by their respective destinations. In the following, we describe two sets of experiments. First, we compare the average packet delay and packet loss in networks with and without dynamic reconfiguration. Second, we compare various retuning strategies in order to determine the effect of several important parameters on network performance.

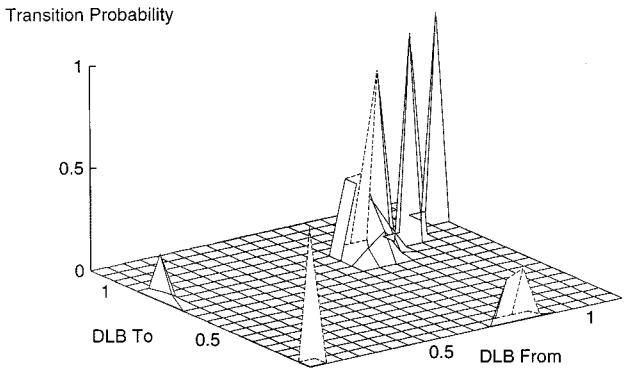

Fig. 20. Conditional distribution of DLB values in the client-server model.

[7]While a change in the client-server assignment is likely to create a completely new traffic pattern and lead to reconfiguration, the behavior of individual sources within a given assignment also causes (smaller) changes in the overall traffic. The accumulation of a large number of these changes may create the need for a reconfiguration within a given client-server assignment.

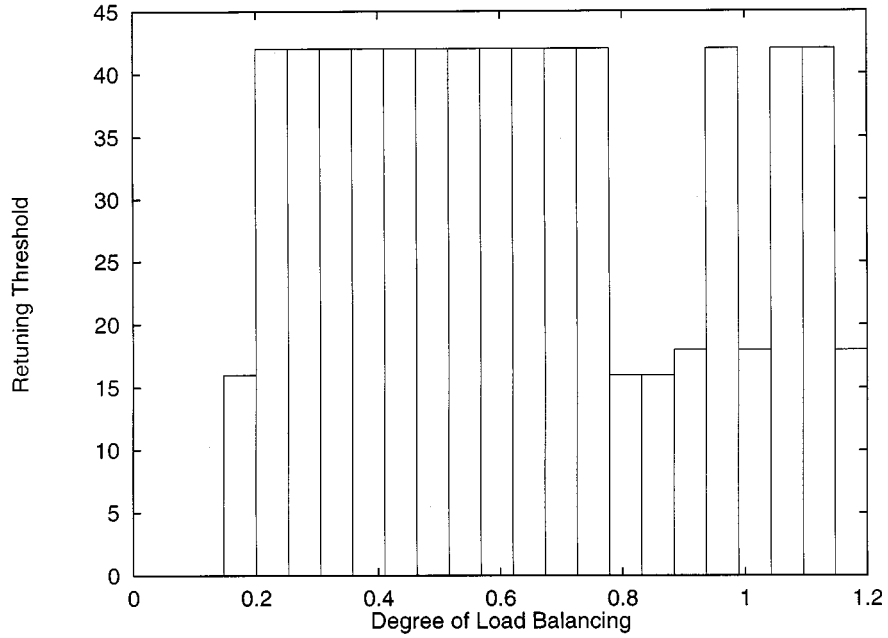

Fig. 21. Optimal reconfiguration policy calculated for the client-server traffic pattern.

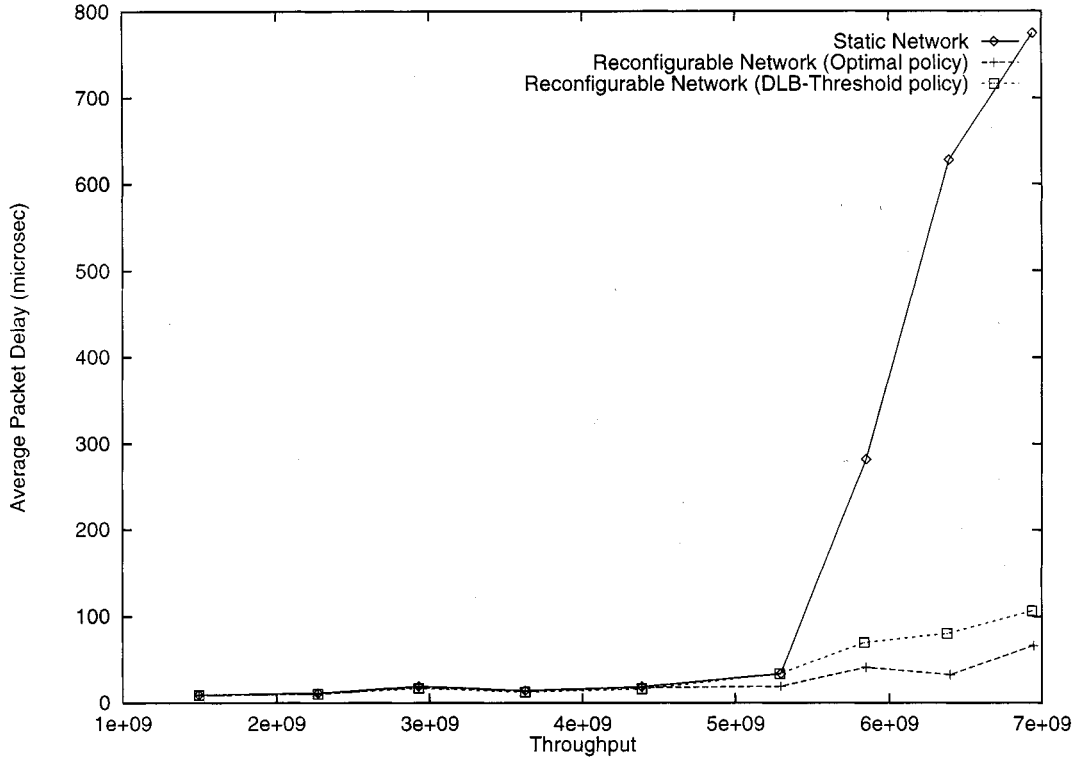

Fig. 22. Average packet delay in networks with and without reconfiguration capabilities.

1) Benefits of Dynamic Network Reconfiguration: We study the performance of the network with and without reconfiguration capabilities. When no reconfiguration is allowed, the assignment of receivers to wavelengths (WLA) remains fixed throughout the simulation (a static network). In a dynamic network, reconfigurations are governed by one of two policies: the optimal MDP policy depicted in Fig. 21 or a DLB-threshold policy (see Section V-A3) with a DLB threshold equal to 0.8. For either policy, the retuning strategy in Fig. 4 with $L = N$ is used, i.e., all receivers needing retuning are simultaneously

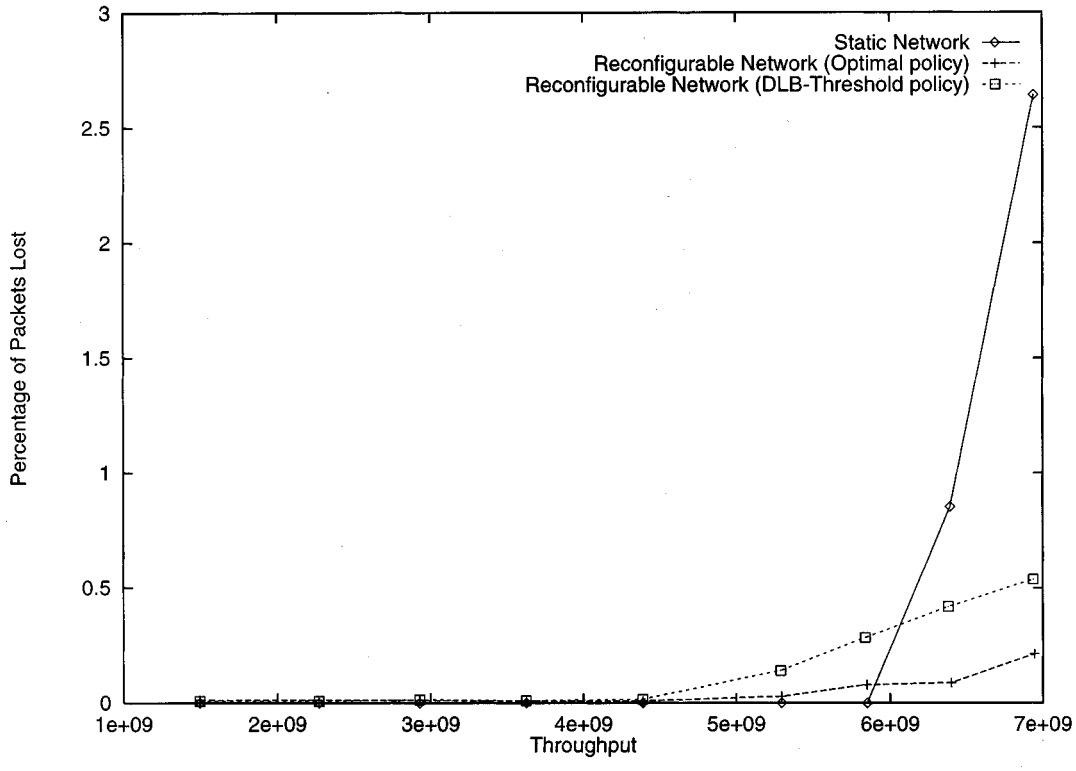

Fig. 23. Packet loss in networks with and without reconfiguration capabilities.

retuned in one step. The receiver tuning latency is assumed to be equal to 1000 packet times (i.e., three orders of magnitude greater than the transmitter latency), while the buffer capacity at each network node is fixed at 200 packets per channel.

In Fig. 22, we plot the average packet delay for each network as a function of total network throughput. As we can see, the networks with a dynamic reconfiguration capability clearly outperform the one without such capability. As the load approaches and exceeds 50% of the total channel capacity (equal to 12.4 Gb/s), the delay in the static network increases very rapidly to nearly 800 μs. On the other hand, the average packet delay in the dynamically reconfigurable networks grows much slower and remains under 100 μs for the loads shown. Also, the network using the optimal reconfiguration policy achieves lower average packet delays than those of the network using the DLB-threshold policy. These results can be explained by the stability condition derived in [11], which states that the maximum sustained traffic load is inversely proportional to the average DLB of the network. By periodically rebalancing the traffic load across the channels, reconfigurable networks maintain a significantly lower average DLB than static networks, allowing them to accommodate higher traffic loads. Furthermore, the optimal policy achieves a lower average DLB than the DLB-threshold policy, resulting in lower delays.

In Fig. 23, we plot the percentage of packets lost in the three networks. As we can see, at low loads, the static network incurs no losses, while the reconfigurable networks do experience some packet loss. This loss is not due to buffer overflows; instead, it takes place during the transition phase due to reconfiguration. Recall that when a receiver is to be retuned, packets currently buffered for it at the various nodes are discarded since they are buffered for transmission on the wrong (old) wavelength. Thus, this loss represents the penalty incurred for having a network that is traffic-adaptive. However, at higher loads, the static network experiences quite high losses due to buffer overflows. Since the WLA is fixed in a static network, when the traffic patterns change such that one channel carries a large part of the overall traffic, buffers for this channel quickly fill up in all network nodes. Reconfigurable networks, however, periodically rebalance the traffic load, preventing unnecessary buffer overflows. As a result, at higher loads, packet losses are significantly lower than in the static network. Again, packet loss is lowest when the optimal policy is used; at the highest load in Fig. 23, the packet loss with the optimal policy is one order of magnitude lower than the loss in the static network.

Fig. 24, which shows the maximum buffer occupancy for the various traffic loads, provides further support to these conclusions. At low loads, buffer occupancy levels for the static network are low, but at high loads maximum buffer occupancy equals the buffer capacity, indicating packet loss due to overflows. On the other hand, buffer occupancy for the reconfigurable networks is low at low loads, confirming that the losses shown in Fig. 23 are indeed due to reconfiguration. Furthermore, the maximum buffer occupancy when the optimal policy is used is quite low and never equals the buffer capacity for the range of loads shown. This fact explains the low delays in Fig. 22 and indicates that this network never loses any packets due to buffer overflows and can accommodate even

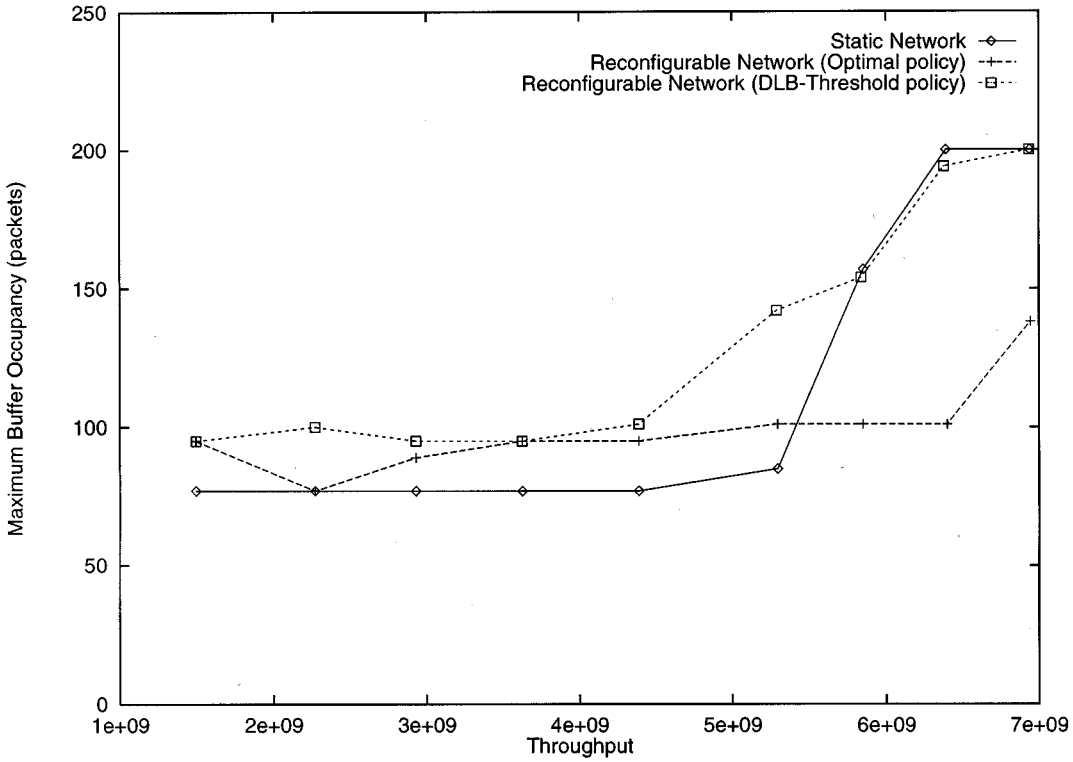

Fig. 24. Maximum buffer occupancy in networks with and without reconfiguration capabilities.

higher loads. When the DLB-threshold policy is used, the maximum buffer occupancy is higher overall and equals the capacity for the highest load shown.

We conclude that under moderate-to-high traffic loads, the performance of a WDM network, in terms of both average packet delay and packet loss, would significantly benefit from having a dynamic reconfiguration capability. Furthermore, this improvement in performance is achieved with a decrease in the amount of buffer space required. These results clearly demonstrate that by adapting itself to prevailing traffic conditions, the novel RTT-STR architecture introduced in this paper can provide a cost-effective solution to building high-performance WDM networks. They also illustrate the importance of employing an optimal reconfiguration policy, since the network using the optimal policy clearly outperforms the one with the DLB-threshold policy.

2) Comparison of Retuning Strategies: We now compare the various retuning strategies in the class shown in Fig. 4 in terms of the same performance metrics considered in the previous section, namely, average packet delay, packet loss, and maximum buffer occupancy. In our study, we only consider a reconfigurable network employing the optimal policy of Fig. 21 and operating under an offered load of 7 Gb/s (the maximum load shown in Figs. 22–24). We have obtained results for three values of the receiver tuning latency, corresponding to 1000, 2000, and 5000 packet transmission times, in order to study the effect of this important parameter on the performance of the various retuning strategies. For the experiments presented in this section, the buffer capacity at each node was set to such a high value that there were never any buffer overflows. Therefore, all the losses shown were entirely due to the transition phase during reconfiguration.

In Figs. 25–27, we plot the average packet delay, packet loss, and maximum buffer occupancy, respectively, as a function of parameter L (recall that in Fig. 4, L corresponds to the maximum number of receivers that can be simultaneously retuned in one step of the transition phase). As we can see, as the receiver tuning latency increases, all three performance measures are negatively affected. This behavior can be explained by noting that the longer it takes each receiver to retune, the longer a group of receivers will be off-line during the transition phase and the longer it will take to reconfigure the network for a given value of L. While a receiver is not operational, newly arriving packets destined for it have to be buffered. Thus, for higher retuning latency values, the maximum buffer occupancy increases (Fig. 27) leading to higher losses (Fig. 26) as well as higher delays (Fig. 25). We also note, however, that within the range of values for L for which the best performance is observed over all three measures (i.e., $5 \leq L \leq 15$), the curves corresponding to a tuning latency of 2000 are very close to those for a latency of 1000, and even the curves for a latency of 5000 are not significantly different. Furthermore, the average packet delay, the packet loss, and the maximum buffer occupancy for a latency of 5000 and $L = 10$ are lower than the corresponding values for a DLB-threshold policy in Figs. 22–24. This result indicates that if an appropriate value for L is used, then even very slow (and,

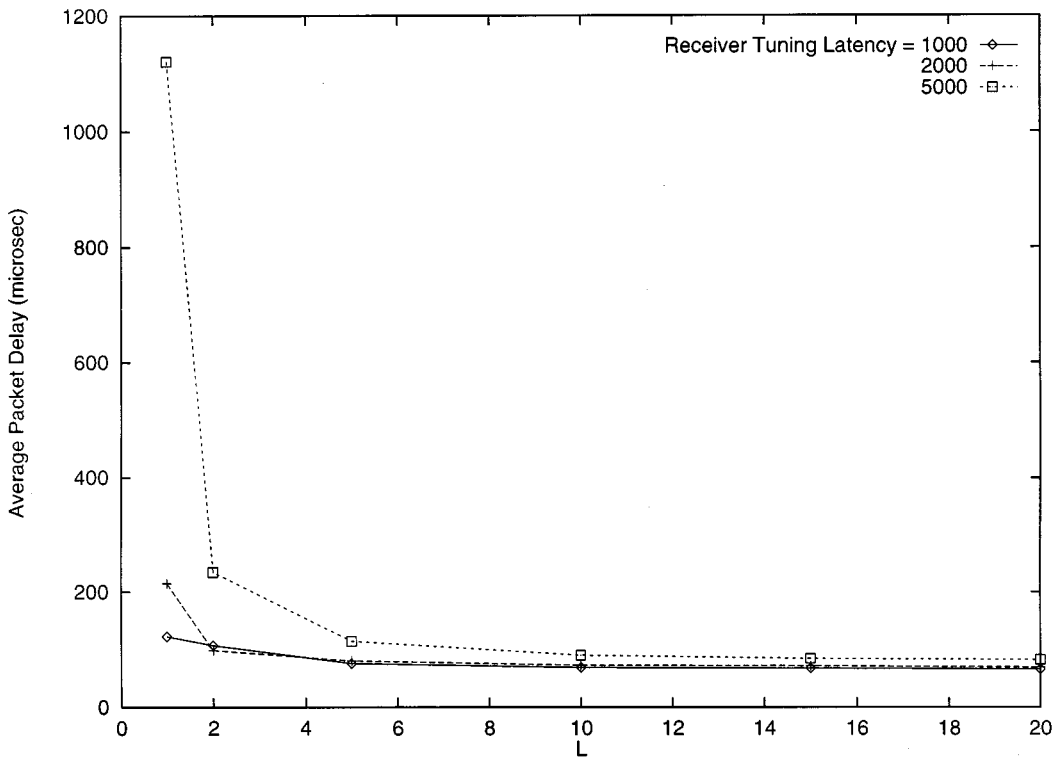

Fig. 25. Average packet delay comparison of retuning strategies.

consequently, inexpensive) tunable devices can be adequate for reaping the benefits of reconfiguration.

Finally, from the three figures, we observe that the two extreme strategies obtained for large values of L (i.e., simultaneously tuning all required receivers at once) and $L = 1$ (i.e., tuning one receiver at a time) do not perform as well as other strategies obtained for values of L between one and N. We can explain this result by considering each of the two extreme strategies separately. When all of the receivers are allowed to retune at the same time, the network takes the shortest possible time to reconfigure to an optimal WLA; however, during this time a large number of receivers are not operational. As a result, even though average packet delay is very low due to a short transition phase, packet losses are relatively high. When the network is only allowed to retune the receivers one at a time, the transition phase is the longest among all strategies. Consequently, the network operates under a suboptimal WLA for a long time, and performance suffers in terms of both packet delay and loss. The results in the figures indicate that strategies with medium values for L (specifically, $L = 5, 10, 15$) achieve the best overall performance.

VI. Concluding Remarks

We have conducted an in-depth study of the reconfiguration problem in traffic-adaptive WDM networks. Our objective was to investigate three open issues: how frequently to reconfigure the network, how to structure the reconfiguration phase, and how to quantify the benefits of reconfiguration to the network in terms of measurable performance parameters.

In order to address the first issue, we developed a formulation based on Markov decision process theory. Given some information regarding the traffic patterns in the network (which can be obtained empirically), the formulation allows us to apply existing algorithms (specifically, Howard's policy-iteration algorithm) to obtain (off-line) optimal dynamic reconfiguration policies. These policies can then be used during the operation of the network (on-line) to determine the instants at which reconfiguration must take place. The policies can be fine-tuned to strike the desired balance between two important but conflicting objectives: the degree of load balancing (which, if considered in isolation, would be optimized by reconfiguring any time there is a slight change in the traffic pattern) and the degree of network unavailability as captured by the number of transceiver retunings (which would be optimized by never reconfiguring the network). Our formulation provides two distinct tools for fine-tuning the policies: the cost functions (which can be selected to reflect specific performance measures) and the weights applied to each objective.

For the second issue, we presented a class of parameterized retuning strategies for carrying out the reconfiguration phase. Under these strategies, the transceivers to be retuned are grouped into sets, and the transceivers in each set are simultaneously taken off-line for retuning. Using simulation, we explored the tradeoffs between the length of the reconfiguration phase (which would be kept short if all transceivers were grouped in a single set) and the portion of network resources that become unavailable at any given time (which would be optimized if

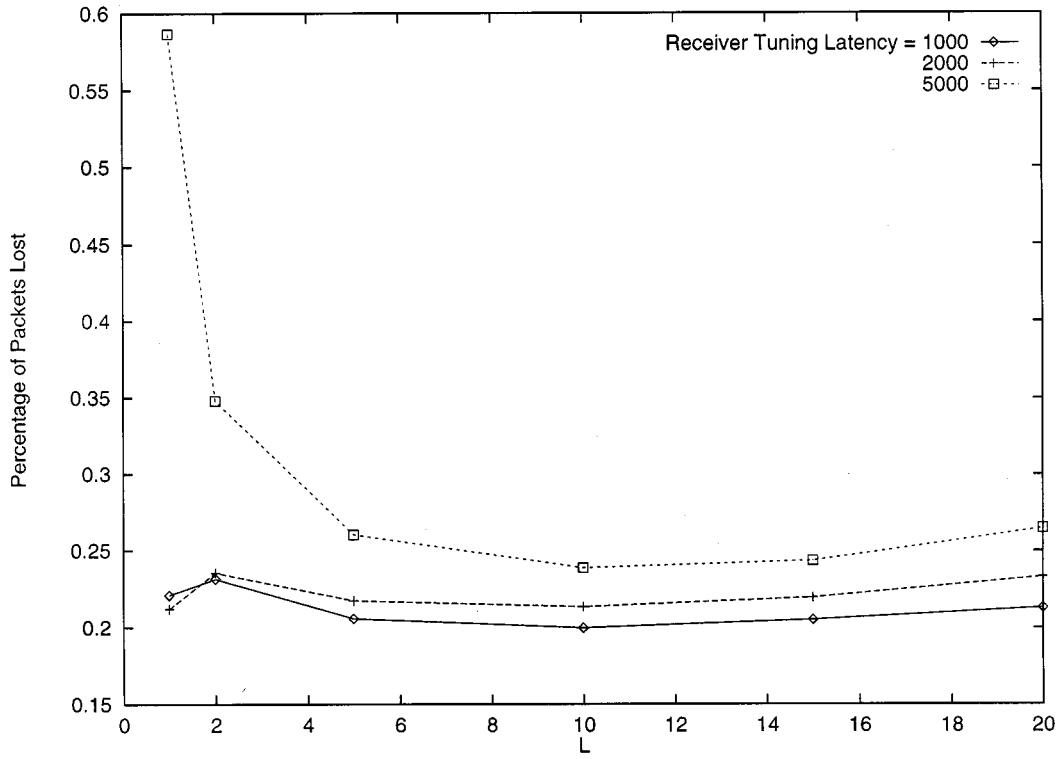

Fig. 26. Packet loss comparison of retuning strategies.

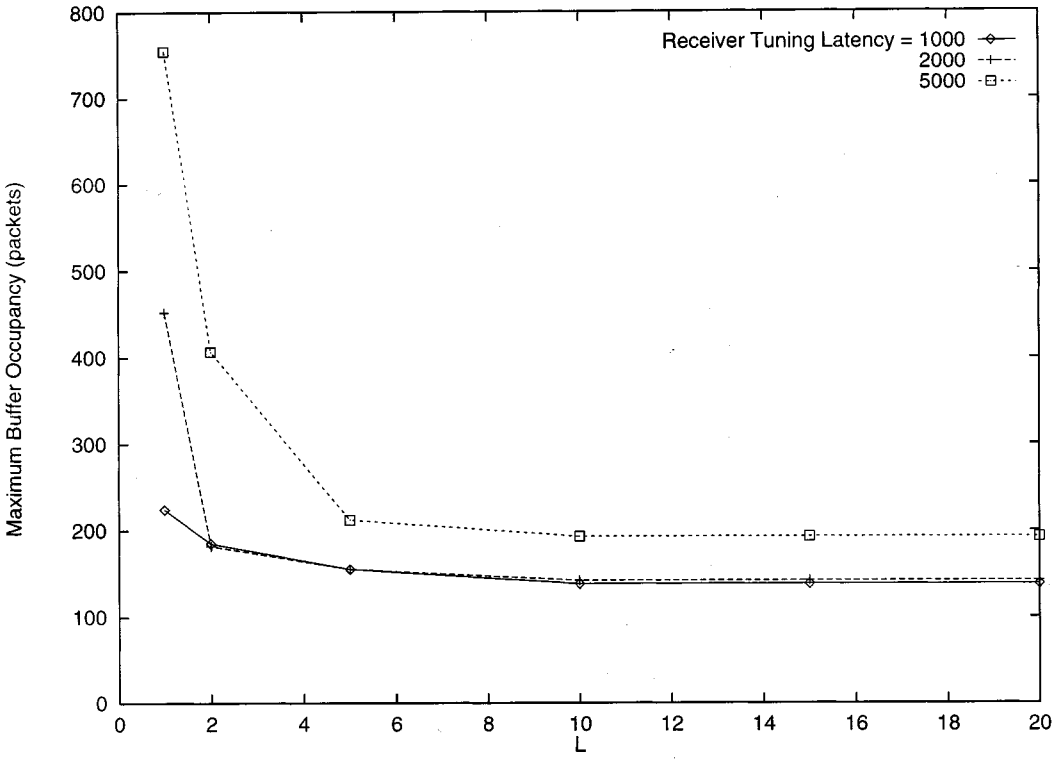

Fig. 27. Maximum buffer occupancy of retuning strategies.

each set consists of a single transceiver). While the relative performance of the strategies is affected by the transceiver tuning latency, our results demonstrate that neither of the two extreme strategies is optimal. Our findings indicate that the

strategies with the best overall performance are those that are in the middle of the spectrum between the two extremes.

Finally, using simulation, we have quantified the benefits of reconfiguration in terms of important performance measures such as average packet delay and packet loss. Specifically, we have observed that at moderate-to-high traffic loads, a network that employs our optimal reconfiguration policies significantly outperforms static networks (no reconfiguration) or networks that apply simple threshold-based reconfiguration policies.

Overall, our work demonstrates that by employing slowly tunable devices, it is possible to build traffic-adaptive high-performance multiwavelength networks cost-effectively. While in this paper we have not considered the effect of equipment failures or signal impairments, reconfiguration can also be invoked as a tool to recover from anomalous conditions. The issue of developing reconfiguration policies for recovery from failures/impairments is a topic of future research.

REFERENCES

[1] I. Baldine and G. N. Rouskas, "Dynamic load balancing in broadcast WDM networks with tuning latencies," in *Proc. INFOCOM '98*, Mar. 1998, pp. 78–85.

[2] I Baldine, "Dynamic reconfiguration in broadcast WDM networks," Ph.D. dissertation, North Carolina State University, Raleigh, 1998.

[3] R. A. Howard, *Dynamic Programming and Markov Processes*. Cambridge, MA: MIT Press, 1960.

[4] J.-F. P. Labourdette, "Traffic optimization and reconfiguration management of multiwavelength multihop broadcast lightwave networks," Comput. Networks ISDN Syst., 1998, to be published.

[5] J.-F. P. Labourdette and A. S. Acampora, "Logically rearrangeable multihop lightwave networks," *IEEE Trans. Commun.*, vol. 39, pp. 1223–1230, Aug. 1991.

[6] J.-F. P. Labourdette, F. W. Hart, and A. S. Acampora, "Branch-exchange sequences for reconfiguration of lightwave networks," *IEEE Trans. Commun.*, vol. 42, pp. 2822–2832, Oct. 1994.

[7] B. Mukherjee, "WDM-Based local lightwave networks Part I: Single-hop systems," *IEEE Network Mag.*, pp. 12–27, May 1992.

[8] H. G. Perros and K. M. Elsayed, "Call admission control schemes: A review," *IEEE Commun. Mag.*, vol. 34, no. 11, pp. 82–91, 1996.

[9] G. N. Rouskas and M. H. Ammar, "Dynamic reconfiguration in multihop WDM networks," *J. High Speed Networks*, vol. 4, no. 3, pp. 221–238, 1995.

[10] G. N. Rouskas and V. Sivaraman, "Packet scheduling in broadcast WDM networks with arbitrary transceiver tuning latencies," *IEEE/ACM Trans. Networking*, vol. 5, pp. 359–370, June 1997.

[11] V. Sivaraman and G. N. Rouskas, "A reservation protocol for broadcast WDM networks and stability analysis," *Comput. Networks*, vol. 32, no. 2, pp. 211–227, Feb. 2000.

Ilia Baldine was born in Dubna, Moscow Region, Russia in 1972. He received the B.S. degree in Computer Science from the Illinois Institute of Technology, Chicago, in 1993, and the M.S. and Ph.D. degrees in Computer Science from the North Carolina State University in 1995 and 1998, respectively.

He currently works as a Network Research Engineer with the Advanced Networking Research group at MCNC, a non-profit research corporation located in Research Triangle Park, North Carolina. His research interests include all-optical networks, ATM networks, network protocols and security.

George N. Rouskas (S'92–M'95) received the Diploma in electrical engineering from the National Technical University of Athens (NTUA), Athens, Greece, in 1989 and the M.S. and Ph.D. degrees in computer science from the College of Computing, Georgia Institute of Technology, Atlanta, in 1991 and 1994, respectively.

He joined the Department of Computer Science, North Carolina State University in August 1994 and he has been an Associate Professor since July 1999. In summer 2000, he was an Invited Professor at the University of Evry, France, and during the 2000-2001 academic year he is on sabbatical leave at Vitesse Semiconductor. His research interests include network architectures and protocols, optical networks, multicast communication, and performance evaluation.

Dr. Rouskas is a recipient of a 1997 NSF Faculty Early Career Development (CAREER) Award, and a coauthor of a paper that received the Best Paper Award at the 1998 SPIE conference on All-Optical Networking. He also received the 1995 Outstanding New Teacher Award from the Department of Computer Science, North Carolina State University, and the 1994 Graduate Research Assistant Award from the College of Computing, Georgia Tech. He was a Co-Guest Editor for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, Special Issue on Protocols and Architectures for Next Generation Optical WDM Networks, which appeared in 2000, and is on the editorial boards of the IEEE/ACM TRANSACTIONS ON NETWORKING and the *Optical Networks Magazine*. He is a member of the ACM and of the Technical Chamber of Greece.