Flow scheduling in optical flow switched (OFS) networks under transient conditions

Rosberg, Zvi; Li, Ji; Li, Fan; Zukerman, Moshe

Published in:
Journal of Lightwave Technology

Published: 01/01/2011

Document Version:
Post-print, also known as Accepted Author Manuscript, Peer-reviewed or Author Final version

Published version (DOI):
10.1109/JLT.2011.2167498

Publication details:
Rosberg, Z., Li, J., Li, F., & Zukerman, M. (2011). Flow scheduling in optical flow switched (OFS) networks under transient conditions. *Journal of Lightwave Technology*, *29*(21), 3250-3264. Article 6015515. https://doi.org/10.1109/JLT.2011.2167498

Rosberg, Z., Li, J., Li, F., & Zukerman, M. (2011). Flow scheduling in optical flow switched (OFS) networks under transient conditions. *Journal of Lightwave Technology*, 29(21), 3250-3264. [6015515]. https://doi.org/10.1109/JLT.2011.2167498.

Flow Scheduling in Optical Flow Switched (OFS) Networks under Transient Conditions

Zvi Rosberg, Ji Li, Fan Li and Moshe Zukerman, *Fellow, IEEE*

***Abstract*—Optical flow switching (OFS) has been recently introduced as a potential "green" architecture addressing the power issue of store-and-forward packet switching in future MAN-WAN Terabit networks. One key architectural component of OFS differentiating it from other "green" WAN architectures such as optical circuit switching (OCS), optical packet switching (OPS) and optical burst switching (OBS), is its centralized flow scheduling. Comparing the theoretical network capacity regions of OFS, OCS, OPS and OBS has revealed that the dominating theoretical capacity depends on the hardware as well as on the port configuration. The dominating actual capacity (throughput) that can be achieved also depends on the flow schedulers supported by each architecture. Since centralized scheduling incorporated in OFS is the least restricting between all scheduling methods, OFS is a promising "green" architecture option for future MAN-WAN Terabit networks. For better understanding the actual potential throughput of OFS, we study its scheduling problem in a realistic traffic model where lightpath requests arrive as a time-dependent Poisson process with Pareto distributed lightpath service times. Lightpath schedules are taken at fixed time intervals (larger than 100 ms) in a central node and flows that have already been scheduled cannot be interrupted before their completion. The scheduling problem is represented as a discrete-time Markov decision process where the objective function is given by the flow blocking probability over a finite time horizon. We derive three lower bounds to the objective function and propose several schedulers, with and without fairness requirements. The performance of our OFS schedulers are evaluated under both static and limited dynamic routing, by emulating the algorithms on random network topologies for two hours. The main result is that our proposed max-min fair scheduler with limited dynamic routing significantly outperforms all other schedulers with static routing. Furthermore, its blocking probability is close to the lower bound for static routing.**

***Index Terms*—Optical Flow Switching, Transient Blocking Approximation, Flow Scheduling.**

I. Introduction

IT is recognized that one of the main power-hungry functions of current communication networks is the store-and-forward electronic packet switching (EPS) [1]. Thus, a "green" architecture is sought where store-and-forward packet switching is minimized.

Previously well studied wide area network (WAN) architecture proposals are optical circuit switched (OCS) networks [2], [3], optical packet switched (OPS) networks [4]–[6] and optical burst switched (OBS) networks [7]–[9]. Recently, a new intriguing architecture combining metropolitan and wide area networks (MAN-WAN), called optical flow switched (OFS) networks, has been proposed in [10]–[12]. For completeness, the main distinctions between the various architectures are briefly given in Section II below.

A key concept is a *lightpath* defined as an all optical connection routed in the optical domain along one or more fiber links [13], [14]. Since wavelength conversion is feasible, the wavelengths of a given lightpath are not necessarily the same. With OFS, users reserve end-to-end lightpaths for long duration transactions of at least 100 ms. The lightpaths are scheduled by a centralized scheduler coordinated through an electronic control plane akin to the classical OCS. For efficient implementation, the centralized scheduler could be executed by several coordinated nodes located at network focal points. For hardware and management simplicity, it is assumed that the smallest granularity of bandwidth that can be reserved across each link of the core network is a full wavelength. Similarly, it is further assumed that wavelength conversion is supported by all routers and the required lightpaths can be setup with the available wavelengths and convertors. The scheduling problem without wavelength conversion is fundamentally similar, but it results in substantially lower link utilization [15].

Transactions that cannot utilize the full bandwidth of a wavelength on their own are multiplexed onto the same lightpath for the transmission across the core network. The multiplexing and demultiplexing are done at the edge nodes. Hungry bandwidth connections can be supported by allocating them multiple lightpaths. For simplicity of presentation, we limit our model to the case of single lightpath connections; however, the model is extendable to multiple lightpath connections as well.

The set of user transactions traversing the same lightpath is referred to as an *OFS flow* (briefly, a flow). Multiple flows between the same source-destination, whether or not using the same physical links, are possible.

Unlike EPS networks, all queueing of OFS data occur at the edge nodes, thereby obviating the need for buffering in the core. The core nodes of an OFS network are equipped with bufferless optical-cross-connect switches (OXC).

OFS is a centralized transport architecture in that coordination is required for logical topology reconfiguration. A high level topology layout is depicted in Fig. 1 in which three MANs along with their attached access networks are interconnected by a single WAN. For illustration, we also de-

Manuscript received March 03, 2011; revised August 04, 2011; accepted August 27, 2011.

The authors are with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong. (e-mail: rosberg@fairflows.com, woshiliji@yahoo.com, fanli3@cityu.edu.hk, m.zu@cityu.edu.hk).

The work described in this paper was supported by a grant from City University of Hong Kong (Project No. SRG-7008089).

pict a centralized scheduler located in three nodes (connected by a permanent control channel) as well as two end-to-end lightpaths.

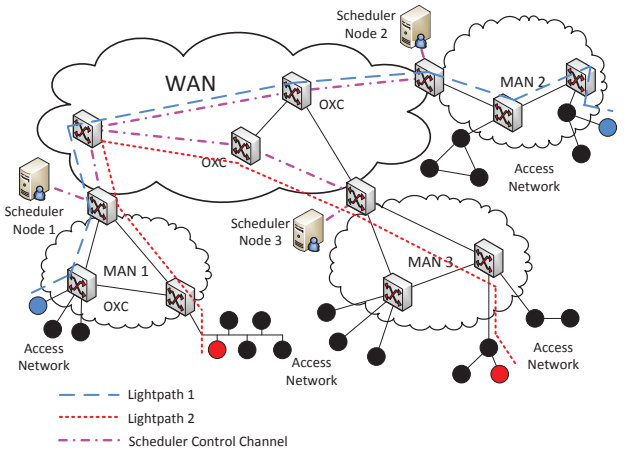

Fig. 1. An OFS topology layout.

In future Terabit Internet, traffic in the core network will likely be efficiently aggregated and sufficiently intense to warrant a slowly-varying quasi-static logical topology. Hence, the centralized management and control required for OFS is not expected to be onerous from the capacity perspective; however, it could present the following concern to applications requiring a short set-up delay, e.g., VoIP. The concern of set-up delay with OFS is further discussed below. Note that set-up delay is not the same as packet delay and bears no impact on it. Also note that packet delay in all-optical transport architectures is usually not an issue except for OBS and OPS, where flow bandwidth is not reserved for the entire connection time.

The OCS and OFS architectures are essentially similar in the abstract level. Unlike OCS which is designed for a WAN topology layout, OFS aims at extending the optical bypass of GMPLS to the end users attached to interconnected MANs [11].

The flow scheduling problem of OFS is different from the traditional routing and wavelength assignment (RWA) problem [16]–[20] applied to optical circuit switching (OCS). Indeed, on the one hand, OFS scheduling is centralized and is carried out in discrete points of time (at least 100 ms apart) where multiple flow candidates are already waiting in the edge buffers. However, on the other hand, routes and wavelengths in RWA are assigned on-the-fly upon individual flow arrivals. Consequently, OFS scheduling lends itself to a combinatorial problem, whereas the traditional RWA can be viewed as an on-the-fly admission control problem.

Now we further discuss the concern of the set-up delay in OFS. Note that the minimum time between two consecutive centralized lightpath scheduling epochs is determined by two variables: (i) the time to deliver the bandwidth requests from the source nodes to the central scheduler, A, and (ii) the scheduler processing time, B. Since A and B can be pipelined in time, the minimum time between scheduling epochs is given by A. Since the maximum time to deliver an optical signal around the world is 200 ms, the time between consecutive scheduling epochs (in the worst case layout) cannot be less than 200 ms.

However, with 200 ms between scheduling epochs, the application set-up delay, defined as the time between a new application arrival and its first access to the bandwidth, could be longer than 200 ms. In the case where lightpath allocations also reserve spare bandwidth for future arrivals, the start of some applications could be delayed for at most 200 ms and others (which cannot fit into the spare capacity) would wait for an additional period of B ms. Thus, the computational complexity of B is important for designing the amount of spare capacity and for determining the limitations of applications requiring short set-up time.

Apparently, the best performing scheduler analyzed in this paper, the max-min fair scheduler with limited dynamic routing (implemented with a round-robin (RR) rule) has a computational complexity of $\kappa \cdot N^2$, where N is the number source-destination edge pairs and κ is a constant. Timing our implemented code with $N = 100$ reveals that the processing time of max-min fair scheduler is 0.9 ms implying $\kappa = \frac{0.9}{100^2}$. Thus, the computational time for scheduling N source-destination edge pairs is expected to take $\frac{0.9}{100^2} \cdot N^2$ ms. Particularly, for $N = 100, 2000$ and $10,000$ source-destination edge pairs, the scheduling times are expected to take $0.0009, 0.36$ and 9 sec, respectively. It is arguable what should the maximum application set-up delay be. It clearly depends on the application and we suggest using VoIP as a reference application. Moreover, we argue that the number of phone rings one would tolerate until s/he hangs up is a reasonable subjective estimate. Our own tolerance level is 6 rings (about 6 seconds).

The remainder of the paper is organized as follows. In Section II we briefly review the macroscopic distinctions between optical transport architectures. In Section III we define our OFS transient model and in Section IV we explain how to approximate transient blocking probabilities. The Markov decision process and lower bounds are derived in Section V and our heuristic schedulers without fairness consideration are defined in Section VI. The schedulers with the max-min fairness consideration are given in Section VII and numerical examples are presented in Section VIII. Finally, conclusions are drawn in Section IX.

II. Macroscopic Distinctions Between All-Optical Network Architectures

Relevant optical transport architectures, which can qualify for a "green tag", are those that reduce buffering and table lookup. These are OPS, OCS, OBS and OFS due to the fact that data switching is mainly done in the optical domain. For completeness, we briefly describe the main distinctions between their bandwidth schedulers.

A. OPS

Proposals for OPS envision a network comprising optical fibers and interconnecting WDM OPS nodes capable of switching variable length packets in the optical domain. Data

packets are individually and independently transported along optical fibers using WDM or DWDM and the wavelength capacity assignment, routing and switching are done in a distributed way. The analysis of [11] suggests that OPS has an optimal theoretical network capacity region.

Currently, packet switching is mainly performed in the electronic domain. However, electronic switching does not qualify as a "green" architecture and is expected to present a great concern in the future as traffic and capacity increase.

B. OCS and OFS

OFS is a network architecture where ingress nodes assemble data packets into very long bursts and employ a centralized scheduler to set up lightpaths (circuits) for those bursts. Lightpath setting with OCS is similar but usually performed off line. While OCS is currently used only in the WAN, OFS is aimed at setting up lightpaths between end users attached to MANs interconnected by WANs. Both architectures do not buffer data at the core network and switching is optical (no table lookup). Thus, they both qualify as "green" architectures.

The analysis of [11] suggests that OFS and OCS have a theoretical network capacity region determined by the optimal solution of the maximum number of flows that can be transmitted concurrently (shown to be in NP-hard [21]).

In light of the NP-hard computational complexity, the focus of this paper is to explore computational efficient schedulers needed for the OFS implementation.

C. OBS

OBS is a network architecture where ingress core edge nodes assemble data packets into bursts of a variable length (usually short, in contrast to the very long bursts of OFS or OCS) destined to the same egress core node. Uniquely to OBS, the lightpath for each burst is reserved and pre-configured by a corresponding setup control packet traversing ahead of the burst. Each core switch that can accommodate the burst for the next link hop, pre-configures the switch to avoid data buffering at the core. The first switch along the burst path, that cannot accommodate the burst, dumps the burst. Aiming to reduce burst loss, an option that allows burst deflection routing has been studied in [22].

Similarly to OCS/OFS, OBS does not use electronic buffering (for contention reduction it can use fiber delay lines (FDL) for short time buffering) and also qualifies as a "green" architecture. Unlike OCS and OFS which employ centralized lightpath schedulers, OBS employs a distributed scheduler usually using a random channel access method. One exception is wavelength-routed OBS [23] which is practically an OCS for short bursts.

To estimate the network capacity region of OBS and relate it to that of OCS/OFS one can draw conclusions from a single channel with random access, on one hand, and reserved access (e.g., TDM), on the other hand. It is well known that random access has a capacity region substantially lower than that of a reserved access. Similar arguments are made in [11] to assert that the network capacity region of OBS is dominated by that of OFS/OCS. It is worth noting that under low load, the set-up delay with OBS outperforms all other architectures.

To obtain a fair idea on the expected throughput of OBS in our traffic model and compare it with OFS, OBS JET [7] is modeled by a random scheduler defined and justified in Section VIII.

III. The OFS Model

To derive flow schedulers and to evaluate their performance in a core network operating under the OFS transport architecture when flow arrivals are time dependent, we model it as follows. Let $1,2,\ldots,L$ denote the network link labels connecting the OXC switches where every link l supports $W(l)$ wavelengths, each with the same capacity c b/s. We assume that wavelength conversion is supported, hence, each wavelength can be switched by the OXC from any input port and lambda (color) to any other output port and lambda. Setting up a lightpath requires reconfiguration and setup time. A new arriving OFS flow is buffered until the next scheduling epoch where it is either assigned to a predetermined route associated with one or more lightpaths (if available) or dropped if none is available. Since under OFS architecture [12], routing and scheduling epochs are performed centrally at time granularity of about 100 ms, we can consider either a combined RWA, or separate routing and wavelength assignment problems. In this study we focus on the wavelength assignment problem given that the routes have been determined in the first phase. We refer to this problem as a *flow scheduling problem*.

Thus, we assume that at the scheduling epoch the routes are given. A fixed route $r=(l_1,l_2,\ldots,l_n)$ of length n is an ordered set of links connecting a source edge node to a destination edge node. Each lightpath corresponds to a route by associating available wavelengths along each link. Let $\mathcal{R}$ be the set of all predetermined routes. The capacity of a route r is simply the capacity of a single wavelength denoted by c.

As described above, in the OFS architecture, a single flow could be an aggregation of multiple user transactions arriving at the same source edge node from the access networks and are destined to the same destination edge node. The aggregation is performed at the source edge node packing all user transactions with the same destination edge node into the same aggregate flow so as to minimize the number of required wavelengths. The bandwidth of each flow is at most c b/s and requires a single lightpath.

Flows manifest themselves (i.e., arrive) at the network edge nodes and are buffered there until the next scheduling epoch. OFS flow arrival processes can be modeled as independent processes, each of which follows a time dependent Poisson process with time dependent rates $\{\lambda_r'(t);\ t\geq 0,\ r\in\mathcal{R}\}$. The Poisson law is well regarded as a good law for modeling arrivals from a large number of independent sources. Since the arrival rates change during the day we use time dependent Poisson arrivals rather than the conventional homogeneous Poisson process.

According to the OFS transport architecture, (aggregated) flows are scheduled to their routes by a centralized reservation scheduler at the beginning of each synchronized time-slot of

duration T. Consequently, the relevant time-dependent flow arrival rate at route r are given by

$$\lambda_r(k) = \int_{t \le kT}^{t \le (k+1)T} \lambda'_r(t)dt, \quad k = 0,1,2,\ldots$$

Each flow is associated with a variable service time, S, which we assume to be Pareto distributed with scale $\beta > 0$ and shape $\alpha > 0$, i.e.,

$$P(S > x) = \left(\frac{\beta}{x}\right)^{\alpha}, \quad x \ge \beta. \tag{1}$$

We assume that all service times are independent. Note that Pareto distribution became a standard convention to model long range dependent (LRD) heavy tailed service times observed in the current Internet [24], [25].

Relevant to our discussion on the transient blocking probability given below, are the mean value of the flow service time S

$$E(S) = \frac{\alpha\beta}{\alpha - 1}, \ \alpha > 1 \tag{2}$$

and its *stationary excess distribution*, S_e, whose complementary c.d.f. is given by (see e.g., [25])

$$P(S_e > x) = \begin{cases} \frac{1}{\alpha}\left(\frac{\beta}{x}\right)^{\alpha-1}, & x \ge \beta, \\ \frac{\alpha-1}{\alpha}\left(1 - \frac{x}{\beta}\right) + \frac{1}{\alpha}, & \text{otherwise.} \end{cases} \tag{3}$$

At every multiple of T, kT, (briefly, every k), the scheduler assigns a subset of the flows waiting in the queues of all edge nodes to their corresponding routes for the next time period T based on the available lightpaths.

Since scheduler decisions are made at times $\{kT;\ k = 1,2,\ldots\}$, all transactions of a flow arriving before the flow is scheduled are buffered at the edge node. We consider a loss system where flows that cannot be scheduled at time kT are dropped and lost. The reasoning for this rule is that the data packets of a buffered flow have already waited for an average of $T/2$ (i.e., more than 50 ms), which justifies their dropping.

If a scheduled flow does not complete its required service time until the next decision epoch, it remains in the system and keep holding its lightpath for the next time interval. Consequently, the set of new scheduled flows at time k comprise only those which have arrived during $((k-1)T, kT]$. We refer to such scheduling policies as *persistent flow schedulers*.

The advantage of persistent schedulers (over nonpersistent ones) is that the reserved lightpaths of the applications are guaranteed for the entire connection time (very important for TCP connections). The disadvantage of persistent schedulers is that lightpath allocations are more restrictive compared to allocation with nonpersistent schedulers.

To evaluate the flow blocking probability under a time-dependent arrival process and compute one of our lower bounds to the optimal blocking probability, we provide an approximation in the next section for the transient *flow blocking probability* at time k.

For clarity, we summarize below the main notions used in this paper.

Notation Summary

- $W(l)$: *No. of wavelength in link l.*
- $\mathbf{r}$: *A route of n links, where $\mathbf{r} = (l_1, l_2, \ldots, l_n)$.*
- $\lambda_{\mathbf{r}}(k)$: *The Poisson arrival rate at route $\mathbf{r}$ in time step k.*
- S: *A flow service time (Pareto distributed).*
- $\mathcal{N}_{\mathbf{r}}(k)$: *The set of flows at time k queued for route $\mathbf{r}$.*
- $N_{\mathbf{r}}(k)$: *No. of flows at time k queued for route $\mathbf{r}$.*
- $\mathcal{N}'_{\mathbf{r}}(k)$: *The set of new flows at time k queued for route $\mathbf{r}$.*
- $N'_{\mathbf{r}}(k)$: *No. of new flows at time k queued for route $\mathbf{r}$.*
- $X_{\mathbf{r},n}$: *No. of steps flow n at route $\mathbf{r}$ is residing in the system.*
- $\mathbf{X}(k)$: *The Markov decision process state of all $\{X_{\mathbf{r},n}(k)\}$.*
- $\pi(k)$: *The scheduling decision of scheduler π at step k.*
- $W_{\mathbf{r}}(k)$: *No. of lightpaths allocated to the flows queued for route $\mathbf{r}$ by a scheduler at step k.*
- $C_{\pi}(k)$: *Immediate cost of action $\pi(k)$.*
- $V_K^{\pi}(\mathbf{x})$: *Total expected cost for K steps under scheduler π given the initial state is $\mathbf{x}$.*
- $V_K^{*}(\mathbf{x})$: *Optimal cost for K steps given the initial state is $\mathbf{x}$.*

IV. Transient Blocking Probability Approximation

We use the following standard approximation for an $M_t/G/L/L$ queue known as the *modified offered load (MOL)* [26]. According to this model, M_t represents the time dependent Poissonian flow arrivals with rates $\lambda(t)$, G represents the Pareto distributed flow service time and L represents a number of lightpaths that flows arriving at a particular route can use. Note that in our network scheduling model, the available number of lightpaths for one route is affected by traffic on other routes.

A flow is blocked at time k if the number of flows requesting lightpaths at time $t = kT$ is larger than L. Denote this probability by $B(k)$ and define a function

$$\beta_L(x) = \frac{x^L/L!}{\sum_{j=0}^{L}(x^j/j!)}.$$

Following [26], we approximate $B(k)$ by

$$B(k) = \beta_L\Big(E\big[Q_{\infty}(kT)\big]\Big), \tag{4}$$

where

$$\begin{aligned} E\big[Q_{\infty}(kT)\big] &= E\big[\lambda(t - S_e)\big]E[S] \\ &= E\big[\lambda(t - S_e)\big]\tfrac{\alpha\beta}{(\alpha-1)} \end{aligned} \tag{5}$$

and the distribution of S_e is given by (3).

The already scheduled flows may have arrived during any time prior to kT, and therefore the current blocking probability is affected by earlier arrival rates. This consideration is especially important for the case where flow sizes follow a heavy tailed distribution. Thus, by (3) and (5),

$$E\big[Q_{\infty}(kT)\big] = \frac{\alpha\beta}{(\alpha-1)c}\int_0^{\infty}\lambda(kT - x)dP(S_e \le x). \tag{6}$$

For discussion and comparison with other approximations see [27], [28].

In the next section we regard the scheduling problem as a Markov decision process (MDP). Readers who prefer fluent reading may skip the next section and return to it for more thorough understanding of the heuristic schedulers derived in Section VI.

V. A Markov Decision Process

In the context of OFS [10], [11], a flow is an aggregated set of user transactions associated with a core route r requiring a capacity of a whole wavelength.

Since the flow service time, S, is assumed to be Pareto distributed, the distribution of the remaining time of an incomplete flow, S_e, is different from the distribution of a new flow (see (3)). In general, the hazard function of S_e increases with its sojourn time. Therefore, to define a discrete MDP, the state at time k should comprise not only the number of flows of each route, but also their elapsed service times. Since each new wavelength allocation requires configuration time, scheduling decisions are limited to the discrete points of time, $k = 1,2,\ldots$, which are T time units apart.

Let $\mathcal{N}_r(k)$ and $N_r(k)$ be the set of all flows at time k requiring route r, and its number of flows, respectively. Additionally, let $X_{r,n}$ be the number of time intervals each flow $n \in \mathcal{N}_r(k)$ has been residing in the system. The system state at time k is given by

$$X(k) = \{X_{r,n};\ n \in \mathcal{N}_r(k),\ r \in \mathcal{R}\}.$$

Note that flow n with $X_{r,n} = 0$ is a new flow that has arrived during interval $\big((k-1)T, kT\big]$. To differentiate between new and ongoing flows, let $\mathcal{N}'_r(k)$ and $N'_r(k)$ be the set and its corresponding number of the new flows only at time k, respectively. The information available to a centralized scheduler, π, at time k, is the state $X(k)$ upon which it makes a decision.

Our main focus here are schedulers that maintain existing incomplete flows with $X_{r,n} > 0$, and drop only new flows that cannot be scheduled. Recall that such schedulers are referred to as persistent schedulers. Also, let $W_r(k)$ denote the number of wavelengths (lightpaths) scheduled to flows requiring route r at time k. A feasible persistent scheduling decision at time k , $\pi(k)$, is a mapping

$$\pi(k) : \big\{X(k)\big\} \to \{W_r(k)\},$$

satisfying constraints

$$\begin{aligned} &N_r(k) - N'_r(k) \le W_r(k) \le N_r(k),\ \forall\ r; \\ &\sum_{\{r:\ l \in r\}} W_r(k) \le W(l),\ \forall \text{ link } l. \end{aligned} \tag{7}$$

That is, a feasible scheduler does not interrupt flow transmissions and preserves the capacity constraints (number of wavelengths) of each link in the network. For every state $X(k) = x$, let $\mathcal{A}(x)$ be the set of feasible scheduler actions satisfying (7).

Clearly, to maximize the throughput a schedule may comprise multiple concurrent flows. A set of concurrent scheduled flows is referred to as a *transmission set*. With wavelength conversion, the constraints of (7) characterize a feasible scheduler decision. Also, in our immediate (one step) cost function, we account only for flow dropping, but not for flow delay time. Notice that both the delay until the flow is scheduled and its transmission time have a constant expected value invariant of the scheduler.

Consider a feasible scheduler π. Given that the state at time k is $X(k)$, the *immediate cost* of an action $\pi(k) \in \mathcal{A}(X(k))$ is therefore defined as

$$C_\pi(k) = \sum_{r \in \mathcal{R}} \big[N_r(k) - W_r(k)\big]^+. \tag{8}$$

For every given scheduler π, characterized by $\{W_r(k);\ k = 1,2\ldots\}$, the process $\{X(k);\ k = 1,2,\ldots\}$ is a Markov chain evolving as follows.

Let $S(y)$ denote the remaining service time given $S > y$. Suppose that $T \ge \beta$. Using the conditional probability formula of the Pareto distribution given by (1), a scheduled flow with an elapsed time of $X_{r,n} \ge \beta$ completes its service and leaves the system with probability

$$P\Big(S\big(X_{r,n}\big) \le X_{r,n} + T\Big) \stackrel{def}{=} P\big(X_{r,n}\big) =$$

$$= \begin{cases} 1 - \left(\frac{X_{r,n}}{X_{r,n}+T}\right)^\alpha, & X_{r,n} = 1,2,\ldots, \\ 1 - \left(\frac{\beta}{T}\right)^\alpha, & X_{r,n} = 0. \end{cases}$$

Denote by $D(k)$ the number of serviced flows leaving the system during step k. Clearly, $D(k)$ is Multinomially distributed with success probabilities $\{P(X_{r,n})\}$.

Additionally, $A_r(k)$ new flows with $X_{r,l} = 0$, $l = 0,\ldots,A_r(k)$, arrive at the system, where $A_r(k)$ is Poissonian distributed with rate $\lambda_r(k)$. Thus, under every policy π, the MDP evolves as

$$N'_r(k+1) = A_r(k),\ \forall\ r \in \mathcal{R};$$

$$X_{r,n}(k+1) = 0,\ \forall\ n \in \mathcal{N}'_r(k+1) \text{ and } r \in \mathcal{R};$$

$$N_r(k+1) = N'_r(k+1) + W(k) - D(k),\ \forall\ r \in \mathcal{R};$$

$$X_{r,n}(k+1) = X_{r,n}(k+1) + T,$$

$$\forall\ n \in \mathcal{N}_r(k+1) \setminus \mathcal{N}'_r(k+1) \text{ and } r \in \mathcal{R}.$$

We consider the following discrete MDP with a finite horizon K. For every initial state x, the cost (total number of dropped flows) of a feasible scheduler π is defined by

$$V_K^\pi(x) = \sum_{k=1}^{K} E\big[C_\pi(k)\big],$$

and the optimal feasible scheduler π^*, is the one attaining

$$V_K^*(z) \stackrel{def}{=} V_K^{\pi^*}(z) \le V_K^\pi(z), \quad \forall\ \ \pi \text{ and } z.$$

The function $V_K^*(z)$ is called the *value function* of the problem.

Since finding the optimal scheduler is mathematically intractable, we derive several lower bounds and advise several heuristic schedulers.

A. A lower bound via policy relaxation

Here, we derive a lower bound on the value function (i.e, flow blocking probability) by relaxing the scheduling rules. Note that a feasible scheduler, π, is constrained to persist the service of incomplete flows. This persistence is expressed by the left-hand side of the first constraint of (7).

To bound $V_K^*(z)$ from below, we consider a new set of schedulers, $\tilde{\pi}$, where the left-hand side of the first constraint of (7) is replaced by 0. This set is referred to as *nonpersistent schedulers* since they are not constrained to persist previously lightpath allocations. Its set of actions at state $X(k) = x$ is denoted by $\tilde{\mathcal{A}}(x)$. We remind the reader that the focus of this paper is on persistent schedulers and that certain nonpersistent schedulers are used as bounds and approximations for our persistent schedulers. We do not intend here to find the "best" nonpersistent scheduler.

In the next theorem, we show that the value function of the nonpersistent schedulers, $\tilde{V}_K^*(x)$, bounds $V_K^*(x)$ from below.

Theorem 1: For every horizon K and initial state x, $\tilde{V}_K^*(x) \leq V_K^*(x)$.

Proof: We prove the assertion by induction on K. First note that

$$\mathcal{A}(x) \subseteq \tilde{\mathcal{A}}(x), \quad \forall\ x. \tag{9}$$

For $K = 1$, (9) implies that

$$\tilde{V}_1^*(x) = \min_{\tilde{\pi}(1)\in\tilde{\mathcal{A}}(x)} C_{\tilde{\pi}}(1) \leq \min_{\pi(1)\in\mathcal{A}(x)} C_{\pi}(1) = V_1^*(z).$$

In the following, $E_{x,\pi(1)}[V(X(2))]$ denotes the expected value of a random function $V(X(2))$ whose probability distribution is induced by the one-step transition probability of the MDP $\{X(k), k = 1,2,\ldots\}$, given that $X(1) = x$ and the first scheduler action is $\pi(1)$.

Suppose that the assertion holds true for every $k \leq K$ and we will show that it is also true for $K+1$. Using Bellman optimality equations [29], we have

$$\begin{aligned}
&\tilde{V}_{K+1}^*(x) \\
&= \min_{\tilde{\pi}(1)\in\tilde{\mathcal{A}}(x)} \left\{C_{\tilde{\pi}}(1) + E_{x,\tilde{\pi}(1)}\left[\tilde{V}_K^*(X(2))\right]\right\} \\
&\leq \min_{\pi(1)\in\mathcal{A}(x)} \left\{C_{\pi}(1) + E_{x,\pi(1)}\left[\tilde{V}_K^*(X(2))\right]\right\} \qquad (10)\\
&\leq \min_{\pi(1)\in\mathcal{A}(x)} \left\{C_{\pi}(1) + E_{x,\pi(1)}\left[V_K^*(X(2))\right]\right\} \\
&= V_{K+1}^*(x).
\end{aligned}$$

The first equality is Bellman equation for the nonpersistent scheduler problem. The next inequality follows from restricting the minimum to a subset of the first scheduler action and continuing with the optimal scheduler for the nonpersistent scheduler problem. The following inequality is implied by the induction assumption and the last equality is Bellman equation for the persistent scheduler problem.

Note that by reducing the first scheduler action to $\mathcal{A}(x)$, we impose a one-step transition to states of the persistent scheduler. ■

The lower bound, $\tilde{V}_K^*(x)$, as well as the optimal nonpersistent scheduler can be derived numerically via *value or policy iteration* for MDP. However, due to the continuous state space and the derivation of $D(k)$, it is a quite complex task.

To guide our investigation of persistent schedulers, we explore further the set of nonpersistent schedulers. Consider the nonpersistent scheduler $\tilde{\pi}$ that at every state $X(k)$ schedules a transmission set $W(k)$ satisfying

$$\begin{aligned}
&0 \leq W_r(k) \leq N_r(k),\ \forall\ r; \\
&\sum_{\{r:\ l\in r\}} W_r(k) \leq W(l),\ \forall \text{ link } l.
\end{aligned} \tag{11}$$

The following concept of maximal transmission set is useful for our investigation. A transmission set of flows is *maximal*, if it cannot be extended with another flow without violating constraint (11). Finding the maximum number of flows that can be transmitted concurrently is a special case of the *disjoint paths problem* [30] shown to be in NP-hard [21]. Polynomial approximation algorithms for the disjoint paths problems have been extensively studied in [31] and in references therein.

The solution of the maximal concurrent transmitted flows is clearly given by the NP-hard integer linear program of

$$\max_{\{W_r\}} \sum_r W_r(k), \tag{12}$$

subject to constraints (11).

Finding maximal transmission sets at every state $X(k)$ can be formulated as finding cliques[1] of a graph where the vertices comprise the set of flows at that state. Two vertices are connected by an edge if their corresponding flow can transmit concurrently.

Confining ourselves to nonpersistent schedulers, it is clear that any scheduler $\tilde{\pi}$ that schedules a non-maximal transmission set at step k is not optimal. The reason is that it can strictly be improved by a policy $\tilde{\pi}'$ that does schedule a maximal transmission set at k and continues from $k+1$ and on as $\tilde{\pi}$.

Formally, it is given by the following inequalities. Suppose that $\tilde{\pi}(k)$ is the first time where a non-maximal transmission is scheduled by $\tilde{\pi}$. Shifting the initial time by k (k becomes 1 and $K+1-k$ becomes K), we have

$$\begin{aligned}
V_K^{\tilde{\pi}}(x) &= C_{\tilde{\pi}}(1) + E_{x,\tilde{\pi}(1)}\left[V_{K-1}^{\tilde{\pi}}(X(2))\right] \\
&> C_{\tilde{\pi}'}(1) + E_{x,\tilde{\pi}'(1)}\left[V_{K-1}^{\tilde{\pi}}(X(2))\right] \qquad (13)\\
&= V_K^{\tilde{\pi}'}(x).
\end{aligned}$$

Note that the inequality holds true since the set of scheduled flows $\tilde{\pi}'(1)$ strictly contains the set of scheduled flows $\tilde{\pi}(1)$. Moreover, at time 2, $\tilde{\pi}'$ can drop the additional scheduled flows, if not completed their service, and exactly imitate $\tilde{\pi}$ until the end. The inequality remains strict since there is a positive probability that at least one additional scheduled flow will complete its service at step one. Thus, the following conclusion for the optimal nonpersistent scheduler follows.

[1] In graph theory, a clique in an undirected graph is a subset of its vertices such that every two vertices in the subset are connected by an edge.

Corollary 1: Any flow allocation of the optimal nonpersistent scheduler at time k, $\tilde{\pi}^*(k)$, schedules a maximal transmission set. ■

It should be noted that Corollary 1 does not imply that if the maximal transmission set of $\tilde{\pi}'(k)$ is larger than the maximal transmission set of $\tilde{\pi}(k)$, then its decision at time k can be used to improve $\tilde{\pi}$. Therefore, allocating the maximal transmission set, i.e., the set with the largest number of flows at any time k is not necessarily optimal.

To derive a computational attractive lower bound based on Corollary 1, let $\mathcal{F}^0$ be the solution of the *max-flow* problem, i.e., the maximal transmission set with the largest number of possible flows determined irrespectively of the number of flows in the network. The set $\mathcal{F}^0$ is given by the routes $\{r\}$ corresponding to the solution of

$$\max_{\{W_r \geq 0\}} \sum_r W_r \tag{14}$$

subject to

$$\sum_{\{r:\ l \in r\}} W_r \leq W(l),\ \forall \text{ link } l. \tag{15}$$

Inspired by Corollary 1, the following heuristic nonpersistent policy, denoted by $\tilde{\pi}^0$, could be reasonably close to optimal when the horizon K is large or when the load is heavy. Thus, we use its **blocking probability to compute one of our lower bounds** to the optimal blocking probability, referred to as the *lower via policy relaxation*.

Definition 1: At every step k and state $X(k)$, the nonpersistent policy $\tilde{\pi}^0$ schedules the following flows:

1) $\tilde{\pi}^0(k) = \max\left\{\mathcal{S};\ \mathcal{S} \subseteq \left\{\bigcup_{r\in\mathcal{R}} \mathcal{N}_r(k)\right\} \bigcap \mathcal{F}^0\right\}$;
2) Extend $\tilde{\pi}^0(k)$ to a maximal transmission set with flows from $\left\{\bigcup_{r\in\mathcal{R}} \mathcal{N}_r(k)\right\} \setminus \tilde{\pi}^0(k)$.

B. A lower bound via routing relaxation

Here, we derive a lower bound on the flow blocking probability by relaxing the network routing constraints as given by the second constraint set of (7). To this end, we use the approximation of the transient blocking probability presented in Section IV.

Denote by $L = |\mathcal{F}^0|$ the cardinal number of the max-flow set of the original network. Consider the $M_k/G/L/L$ queueing system with L servers and waiting positions, a time dependent Poissonian flow arrival process $\lambda(k)$, $k = 1,2,\ldots,K$, where

$$\lambda(k) = \sum_{r\in\mathcal{R}} \lambda_r(k), \tag{16}$$

and independent Pareto flow holding times.

Since L is the maximum number of concurrent lightpaths, it follows that for every arrival and service realization, the number of flows serviced by the time dependent $M_t/G/L/L$ system is larger than that of the original system. Thus, the following lower bound follows.

Let $B(k)$ be the transient blocking probability of the $M_k/G/L/L$ queueing system at time k. Recalling that $V_K^*(x)$ is the total expected number of blocked flows in the original system, we have

Theorem 2: For every horizon K and initial state x, $\sum_{k=1}^K \lambda(k)B(k) \leq V_K^*(x)$. ■

For computational purpose, the transient blocking probabilities, $\{B(k)\}$, can be approximated using the MOL method as given by (4) and (5). In our case, j is superfluous and the rates $\lambda(t)$ are given by the sum of all route rates.

C. A lower bound via cost relaxation

In Subsection V-A, we relaxed the persistent schedulers to nonpersistent schedulers. However, the computational complexity of computing the lower bound to $V_K^*(x)$ as given by $\tilde{V}_K^*(x)$ is NP-complete.

To alleviate the computational complexity, we relax the immediate cost function and account only for the new flow dropping. That is, discontinued flows are not penalized. This relaxation of the problem is equivalent to the *complement MDP* where schedulers aim at maximizing the total number of accepted flows. The value function of this complement MDP is denoted by $\hat{V}_K^*(x)$.

Since all unaccepted arriving flows are dropped, the minimum expected total number of blocked flows is given by

$$\hat{B}_K^* \stackrel{def}{=} \sum_{k=1}^K \lambda(k) - \hat{V}_K^*(0). \tag{17}$$

The optimal policy that attains $\hat{V}_K^*(x)$ is clearly the *nonpersistent myopic scheduling* which at every time k and state $X(k)$ schedules the maximum transmission set with the largest number of flows.

To evaluate the lower bound, $\hat{B}_K^*$, consider the following time dependent model $M_t/G/L_k/L_k$ of Section IV with time dependent L_k servers and waiting positions and time dependent Poissonian flow arrival process $\lambda(t)$, $t = 1,2,\ldots,K$.

The values of L_k are pre-computed and stored in a lookup table by solving the following problems for each possible state of number of flows. That is, for each k, let $\mathcal{F}^0(k)$ be the solution of the following *max-flow* problem at phase k,

$$\max_{\{W_r \geq 0\}} \sum_r W_r \tag{18}$$

subject to

$$\begin{aligned} &0 \leq W_r \leq N_r(k),\ \forall\ r; \\ &\sum_{\{r:\ l\in r\}} W_r \leq W(l),\ \forall \text{ link } l, \end{aligned} \tag{19}$$

for all $0 \leq N_r(k) \leq \max_l W(l)$.

A computational procedure to evaluate the lower bound (17) is to simulate the process for K steps. At each step k, the values of $\{N_r(k)\}$ are determined by the simulation and $L_k = |\mathcal{F}^0(k)|$ is computed by solving (18)–(19).

VI. Persistent Schedulers

In this section we derive several heuristic persistent schedulers. Note that our flow scheduling is a centralized problem done at discrete points of time, where multiple flow candidates are available, rather than upon each flow arrival. This is different from the traditional distributed routing and wavelength assignment (RWA) problem [16]–[18] performed upon each flow arrival. Also, our scheduling problem lends itself into a combinatorial problem whereas the traditional RWA problems are admission control problems.

The derivations of the lower bounds suggest the following two heuristic persistent schedulers, the *max-flow persistent scheduler* and the *max current set persistent scheduler*.

A. Max-flow persistent scheduling

The *max-flow persistent scheduler*, π^0, is the persistent version of the nonpersistent policy, $\tilde{\pi}^0$, as defined by Definition 1. Unlike $\tilde{\pi}^0$ aiming at building up the max-flow scheduling state and may discontinue ongoing flows, π^0 does not schedule new flows not included in the max-flow set $\mathcal{F}^0$.

Definition 2: At every step k and state $X(k)$, max-flow persistent scheduler schedules the maximum number of ongoing flows along with the new flows from the set $\mathcal{F}^0$.

Note that the max-flow persistent scheduler translates into a static fixed set of lightpath allocation for all flows in the set $\mathcal{F}^0$. Namely, W_r^0 lightpaths are statically allocated to flows using route r, where W_r^0 is the solution of

$$\max_{\{W_r\}} \sum_r W_r$$

subject to

$$\sum_{\{r:\ l\in r\}} W_r \le W(l),\ \forall \text{ link } l.$$

At every step k, all new flows are assigned to the available lightpaths and the rest are blocked.

Remark 1: Clearly, max-flow persistent scheduler starves the flows not in the max-flow set. This can be partially corrected as follows. Flows not in the max-flow set are FIFO scheduled to any available lightpath until one of its wavelengths is required by a flow in the max-flow set. However, under heavy load when each route is likely to have several waiting flows, this correction will not help.

B. Max current set persistent scheduling

The *max current set persistent scheduler* is the persistent version of the nonpersistent myopic scheduler which attempts at building the most likely maximum transmission set at any given state, as defined below.

Definition 3: At every step k and state $X(k)$, the max current set persistent scheduler schedules the ongoing flows along with new flows comprising the largest subset of any maximal transmission set.

That is, the lightpaths assigned to new flows at step k are $W_r(k)$ solving

$$\max_{\{W_r\}} \sum_r W_r(k)$$

subject to

$$N_r(k) - N_r'(k) \le W_r(k) \le N_r(k),\ \forall\ r;$$

$$\sum_{\{r:\ l\in r\}} W_r(k) \le W(l),\ \forall \text{ link } l.$$

Unlike the max-flow persistent scheduler, which is static, the max current set persistent scheduler is adaptive.

C. Anticipating max current set persistent scheduling

Note that the adaptive *max current set persistent scheduler* is a *non-anticipating* in the sense that it does not account for future flow blocking - only for the current flow blocking. On the other hand, the static max-flow persistent scheduler, is somewhat anticipating by accommodating only flows that could minimize the long-run average flow blocking. However, max-flow persistent scheduler is not adaptive.

The idea of the next scheduler is to convert the *max current set persistent scheduler* into an anticipating one by adding to the objective function the following penalty to the next-step blocking rate.

For every lightpath r, let $P_r(k)$ be the total arrival rate of all other flows colliding with at least one link used by r. That is,

$$P_r(k) = \sum_{r'\ne r:\ r'\cap r\ne\phi} \lambda_{r'}(k). \tag{20}$$

The *anticipating max current set persistent scheduler* assigns lightpaths $W_r(k)$ to new flows at step k which solve

$$\max_{\{W_r\}} \sum_r \left[W_r(k) - A\cdot P_r(k+1)\right]$$

subject to

$$N_r(k) - N_r'(k) \le W_r(k) \le N_r(k),\ \forall\ r;$$

$$\sum_{\{r:\ l\in r\}} W_r(k) \le W(l),\ \forall \text{ link } l.$$

Note that $\{P_r(k+1)\}$ are known constants; hence the objective function remains linear. Additionally, long hop flows and flows using routes comprising links shared by many flows ("bottleneck" flows) are penalized more than short and "non-bottleneck" flows.

The schedulers derived so far are based on a total performance criteria without considering fairness between the flows. In the next section we also consider fairness.

VII. Max-Min Discrete Fair Schedulers

Observe that schedulers attempting to minimize the flow dropping for a given horizon K are unfair in the sense that some of the flows may not be scheduled at all *(starve)*. The reason is that the cost function and the constraints at every step k are linear; consequently, optimal wavelength assignments under constraint (7) are attained at extreme points determining lightpaths only for a subset of flows at each step k.

Therefore, a more practical scheduler should also assign the lightpaths at every step k in a fair manner. A common fairness notion for network bandwidth allocation is *max-min fairness*, [32], where bandwidth is allocated to flows so

as to maximize the minimum allocated bandwidth per flow. Without link capacity constraints, the allocated bandwidth to each flow is the same. With link capacity constraints, max-min fair allocation maximizes the allocated bandwidth of the i-th minimum allocated bandwidth. Additionally, unlike schedulers attempting to minimize the flow dropping, max-min fair schedulers are starvation-safe.

Note that max-min fairness is one of the two most common fairness criteria, where the other one is *proportional fairness* [33]. Fairness criteria is a controversial issue and max-min and proportional fairness are placed in the two ends of a general fairness spectrum introduced in [34]. Both criteria have merit; we selected max-min fairness since its computational complexity is polynomial (even for discrete variables) whereas that of proportional fairness is exponential (solving a convex program).

Max-min fairness is defined as follows.

Definition 4: The continuous max-min fair rate allocation is a feasible rate vector where the rate of each flow i cannot be increased, while maintaining feasibility, without decreasing the rate of some other flow j which has a smaller or equal rate than flow i.

A simple way to illustrate the max-min fair rates is by visualizing the links as one way water pipes with given diameters connected by taps, resembling switches, and generating a network of pipes connecting sources and destinations. Max-min rates are attained by filling each source with water at a constant rate (which is the same for all sources) and gradually increasing the rate until it cannot be increased anymore. This *water-filling* algorithm can be used to implement the continuous max-min fair bandwidth allocation.

Unlike the continuous case, our lightpath allocation problem is a discrete problem where the capacity variables are the number of lightpaths allocated to route r, W_r. The discrete domain necessitates a change in the max-min fair definition, as well as in the algorithm [35]. A simple example given in [35] illustrates the required change in the definition. Consider a single link with 4 lightpaths and 3 routes (flows). There are 3 lightpath max-min allocation vectors, (1,1,2), (1,2,1) and (2,1,1). Each one of these max-min allocations, e.g., (1,1,2), does not satisfy definition 4. Indeed, the rate of flow 1 can be increased from 1 to 2 and the rate of flow 3 be decreased to 1 (the rate of flow 1 before the increase). Thus, we need to modify the max-min definition so as to accommodate discrete unit allocations.

Definition 5: The discrete max-min fair rate allocation is a feasible rate vector where the discrete rate of each flow associate with route r, W_r, cannot be increased to $W'_r > W_r$, while maintaining feasibility, without decreasing the rate of some other flow j which has a smaller or equal rate than the new rate of flow r, W'_r.

Note that unlike the continuous max-min fair rate allocation, the discrete max-min fair rate allocation, a flow associated with route r may be allocated zero lightpaths. This could only occur when other flows sharing a link with r are allocated exactly one lightpath. Also, as shown by the example above, the discrete max-min fair rate vector is not unique and some max-min fair rates can utilize the network substantially better than others. Indeed, consider a network comprising h link hops in tandem labeled as $1,\ldots,h$. Suppose that the route of flow 0 comprises all links and for each $1 \le i \le h$, the route of flow i comprises only link i. Both allocation rates, $(1,0,\ldots,0)$ and $(0,1,\ldots,1)$ are max-min fair rates illustrating the potential difference in their network utilization.

As shown in [35], the following *discrete bottleneck condition* is an alternative definition of discrete max-min fair rates.

Definition 6: A feasible lightpath allocation (rate) vector $\{W_r\}$ is discrete max-min fair if and only if one of the following is satisfied for every flow r.

1) If $W_r \ge 1$, flow r has either a discrete bottleneck link $l \in r$ where $\sum_{\{r':\ l\in r'\}} W_{r'} = W(l)$ and $W_r \ge W_{r'} - 1,\ \forall\ r':\ l \in r'$; or $W_r = N_r$ (each flow is allocated a lightpath).
2) If $W_r = 0$, either there is a bottleneck link $l \in r$ where $\sum_{\{r':\ l\in r'\}} W_{r'} = W(l)$ and $W_{r'} \le 1,\ \forall\ r':\ l \in r'$; or $N_r = 0$.

Using the discrete bottleneck condition of definition 6, it is straightforward to verify that the algorithm defined in the following Subsection VII-A yields a max-min fair rate vector.

A. A nonpersistent max-min discrete fair scheduler

The following *round-robin (RR)* scheduler is a natural discrete version of the water-filling algorithm. Suppose that at the beginning of step k there are $\{N_r(k),\ r \in \mathcal{R}\}$ waiting flows. The RR allocation rule allocates lightpaths iteratively. Let $N_r(k,i)$ denote the number of flows waiting for route r after iteration i is complete; also denote $N_r(k,0) = N_r(k)$. At every iteration, $i \ge 1$, the RR allocates a single lightpath (if available) in an arbitrary order to every route r with $N_r(k,i-1) > 0$. Thus, we define:

Definition 7: At each step k, the nonpersistent max-min discrete fair scheduler allocates the lightpaths in a RR fashion.

It can easily be verified that the lightpaths allocated by the RR scheduler satisfy the discrete bottleneck condition, regardless of the allocation order.

To obtain a discrete max-min fair allocation with the maximal number of allocated flows, one can solve the following max-flow problem at every iteration i.

$$\max_{\{W_r \in \{0,1\}\}} \sum_r W_r$$

subject to

$$\sum_{\{r:\ l\in r\}} W_r \le W(l,i),\ \forall \text{ link } l,$$

where $W(l,i)$ is the number of unallocated lightpaths of link l after iteration i is complete.

B. A persistent max-min discrete fair scheduler

The persistent max-min discrete fair scheduler is similar to the nonpersistent one, except that at every RR iteration, the lightpaths being held by the flows remaining from the previous step are also accounted.

Let $W_r(k,i)$ denote the number of lightpaths used or newly allocated at step k to flows associated with route r after iteration i is complete.

Definition 8: At each step k, the persistent max-min discrete fair scheduler allocates the lightpaths in the following myopic optimal RR fashion.

1) Initialize

$$W_r(k,0) = N_r(k) - N'_r(k),\ \forall\ r,$$

$$W(l,0) = W(l) - \sum_{r:\ l\in r} W_r(k,0),\ \forall\ l.$$

2) At every iteration $i \geq 1$, compute

$$\max_{\{x_r\in\{0,1\}\}} \sum_r x_r \cdot I\{i > W_r(k,0)\}$$

subject to

$$\sum_{\{r:\ l\in r\}} x_r \leq W(l,i),\ \forall \text{ link } l,$$

where $I\{E\}$ is the indicator function of event E.

3) Update $W_r(k,i) = W_r(k,i-1) + x_r$.
4) Stop if $\sum_r x_r = 0$.

Note that the myopic optimality does not imply that the number of rejected flows is minimized since each iteration does not account for remaining wavelengths left for the next iteration.

Since the max-min discrete scheduler with RR turned out to be the most efficient scheduler, we note that its computational complexity is $|\mathcal{R}|^2$. Indeed, it is proportional to $(\max_r W_r) \times |\mathcal{R}|^2$ and $\max_r W_r$ is bounded by the constant $\max_l W(l)$.

VIII. Numerical Examples of Random Networks

The performance of our proposed OFS scheduling algorithms are demonstrated in a random topology comprising 100 source destination edge node pairs (S-D) and 10 physical core links each comprising of 27 wavelengths. We use a random topology rather than specific topologies so as to eliminate the bias introduced by the specific ones. We advocate that it is more appropriate when seeking for a general scheduler aimed at any arbitrary topology.

Three types of network layouts were simulated: one with a symmetrical layout and two types with asymmetrical layouts. The layouts along with their corresponding routes are redrawn at the beginning of each simulation run. For the symmetrical case, a route of each S-D pair has 5 hops on the average and comprises each of the 10 network links with probability 0.2.

For the first asymmetrical layout, referred to as *asymmetric link congestion*, the links comprising each S-D route are drawn so as to generate an asymmetric distribution of the number of routes crossing a single link. Specifically, the 10 links are divided into five equal-size groups. A link from group one is drawn randomly by 10 S-D routes; a link from group two is drawn randomly by 20 S-D routes; and so forth until group five, where a link is drawn by 50 S-D routes. This procedure generates an asymmetrical layout where each S-D route is of length 5, and links are congested asymmetrically.

For the second asymmetrical layout, referred to as *asymmetric route length*, the links comprising each S-D route are drawn so as to generate an asymmetric route length distribution (in hop counts). Specifically, the 100 S-D pairs are randomly divided into five groups, each comprising 20 S-D pairs. The routes assigned to the first group are of length 1, to the second group are of length 2, and so forth until group five, where the route length is 5. The links of each route in every group are randomly drawn from the 10 network links.

Clearly, the links comprising each route are drawn without repetition and avoid loops. Also, in the case where each S-D flow can select multiple routes, each route associated with a given S-D pair is drawn following the same statistical procedure.

We investigate the performance considering two types of routing: (i) *static routing*, where each S-D flow can use only one fixed route; and (ii) *limited dynamic routing*, where each S-D flow can use any one out of four possible routes depending on the network state. For the latter type of routing see, e.g., [36], [37].

We use the traffic model described in Section III with time dependent Poisson flow arrivals with rate λ_t. We consider a transient arrival pattern for a time interval where traffic offered load is linearly increasing in discrete steps which are 0.1 seconds apart. The initial arrival rate is set to $\lambda_0 = 0.875$. Then it incrementally increases by an equal increment of 8.5×10^{-5}. We assume that the processes are independent and have the same arrival rates $\lambda(t)$. Also, the flow service time is Pareto distributed with $\alpha = 2.1$ and $\beta = 0.07$. This is a typical load pattern in the first two hours of a working day.

For each experiment, we generate 24 independent network layouts and ran each layout as to simulate 7200 seconds (2 hours) of the physical network time. The performance measures depicted in the graphs are given by the averages over the 24 runs along with their 95% confidence intervals. The scheduling decisions are taken at 100 ms apart.

A. Performance with a single static route

Fig. 2 depicts the cumulative flow blocking probabilities in the symmetrical network for the following five scheduling algorithms (also depicted in other figures):

1) persistent max current set [MCSP] (see VI-B),
2) nonpersistent opt. max-min fair [MMNP(opt)] (see VII-A),
3) persistent opt. max-min fair [MMP(opt)] (see VII-B),
4) nonpersistent max-min fair with RR [MMNP(rr)] and
5) persistent max-min fair with RR [MMP(rr)].

In this figure (as well as in the rest of the figures) the bottom x-axis is the time, t [s], and the top x-axis is the average offered data load per physical link at time t.

First note the misleading scale artifact of the confidence intervals in Fig. 2. Although they appear larger than the ones in the other figures, they are actually similar in their absolute size. Also, for clarity, we display only the confidence intervals for MCSP; the rest are of a similar size.

Also note that since the arrival process is transient (time dependent), the stationary (steady state) blocking probability does not exist. Therefore, we consider here the ratio of the total number of blocked flows to the total number of flow arrivals until time t, which we call *cumulative blocking probability* depicted in Figs. 2–8. When the system is stationary, the cumulative blocking probability converges to the standard

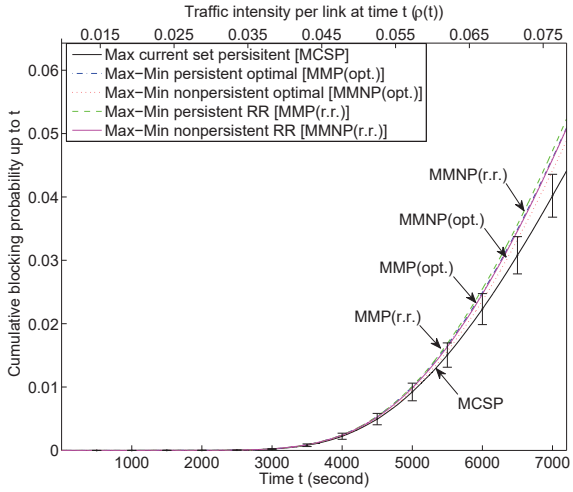

Fig. 2. Cumulative blocking probabilities for symmetrical networks.

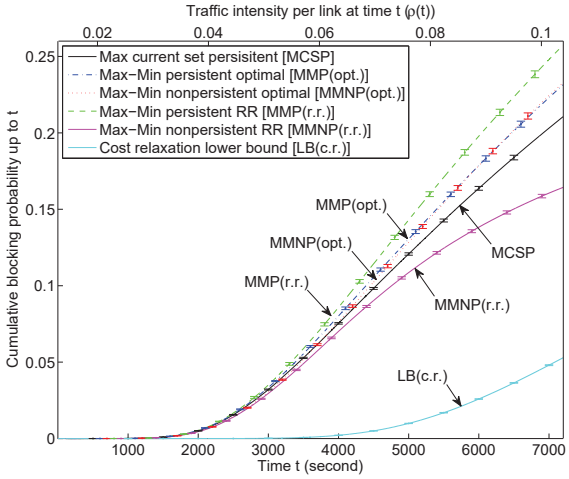

Fig. 3. Cumulative blocking probabilities for asymmetrical networks with different link congestion.

stationary blocking probability. Since our numerical examples concern with the transient case where the flow arrival rate, $\lambda_r(t)$, increases with t, the cumulative blocking probability also increases with t.

It can be observed that the persistent max current set scheduler outperforms all other schedulers and that the non-persistent optimal max-min fair scheduler is the second best. The cumulative blocking probabilities of all max-min fair schedulers are very similar and differ only by about 0.01 and by about 0.03 from MCSP at the largest traffic intensity. Thus, although fairness has a performance penalty it is more desirable in practice *(i.e., starvation safe)* and therefore presenting a reasonable alternative to the persistent max current set scheduler (which is unfair). It is worth noting that the nonpersistent max-min fair scheduler does not necessarily outperform the persistent max-min fair scheduler. The reason is that each iteration selects a flow using a myopic optimal rule rather than a long-term optimal rule.

Fig. 3 depicts the cumulative flow blocking probabilities in a network with the *asymmetric link congestion*. The presented graphs are of the same five scheduling algorithms used in the symmetrical layout. Similarly, Fig. 4 presents the results for a network with the *asymmetric route length*. The cost relaxation lower bound is also presented.

It can be observed that in the case of the asymmetric link congestion (Fig. 3), the nonpersistent RR max-min fair scheduler outperforms all other schedulers. However, amongst the persistent schedulers, the persistent max current set scheduler is dominating for $t > 4000$; and below 4000, all schedulers are similar. It can also be noticed that the blocking probabilities of the persistent optimal max-min fair scheduler is quite close to that of the best persistent scheduler with an cumulative blocking probability only slightly larger, i.e., 0.02 at the highest traffic load.

For the case of the asymmetric route length (Fig. 4), the persistent max current set scheduler outperforms all others $t > 4000$; and below 4000, all schedulers are similar. As in the other asymmetrical case, the cumulative blocking probabilities of the persistent optimal max-min fair scheduler is also quite close to that of the best persistent scheduler with a cumulative blocking probability only slightly larger, i.e., 0.025 at the highest traffic load.

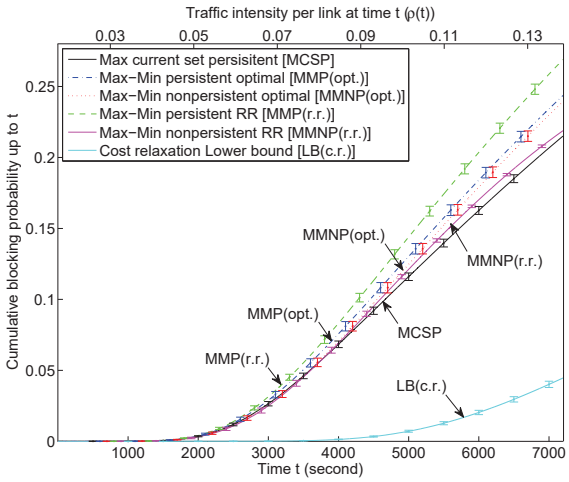

Fig. 4. Cumulative blocking probabilities for asymmetrical networks with different route length.

The results presented in Figs. 2–4 can be summarized as follows.

- Except for the case of asymmetric link congestion (Fig. 3), nonpersistent schedulers bear almost no advantage over the persistent schedulers. In the case of asymmetric link congestion, the cumulative blocking probability of the nonpersistent RR max-min fair scheduler at the highest traffic load is 0.05 less than the best persistent scheduler.
- The unfair persistent max current set scheduler is outperforming all other persistent schedulers. However, the cumulative blocking probability of the nonpersistent RR max-min fair scheduler attains a quite close result.
- The cumulative blocking probabilities in the asymmetrical

layouts are higher than that of the symmetrical layout. i.e., 0.22 vs. 0.048, at the highest offered load.

B. Comparison with a random scheduler

To illustrate the performance of our proposed schedulers, we compare them with the random scheduler analyzed in [16]. A random scheduler allocates the wavelengths to the routes in a random order.

The blocking probabilities of our persistent max current set scheduler are compared with the random scheduler in the symmetric network layout (Fig. 5), the asymmetric link congestion layout (Fig. 6) and the asymmetric route length (Fig. 7). It can be observed that our persistent max current set scheduler is substantially better than the random scheduler.

It is worthwhile to note that the random scheduler fairly approximates the OBS JET scheduler [7]. Indeed, the setup control packets of all source-destination pairs are initiated upon their request arrivals and each follows its predetermined route until either a successful reservation or a contention occurs. Since arrivals at each switch are random, the induced reservation schedule is random as well.

The high cumulative blocking probabilities under the random scheduler is explained by its random wavelength allocation to the routes. A random scheduling is known to be inefficient for high offered loads (e.g., in random access channels). Since our numerical examples concern with the transient case where the flow arrival rate, $\lambda_r(t)$, increases with t, the high load effect starts at $t > 2000$ seconds (for the symmetrical case) and at $t > 1000$ seconds (for the asymmetrical cases).

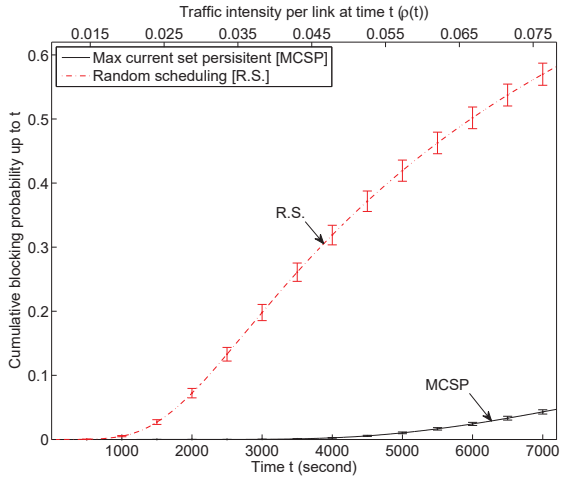

Fig. 5. Comparison of cumulative blocking probabilities with random scheduling for symmetrical networks.

C. Comparison with a limited dynamic routing

To compare the performance of our schedulers in a network where routing is static (each S-D pair has a single static route), to a scheduler where routes are selected dynamically, we consider the following scheduler which adapts to the dynamically selected routes.

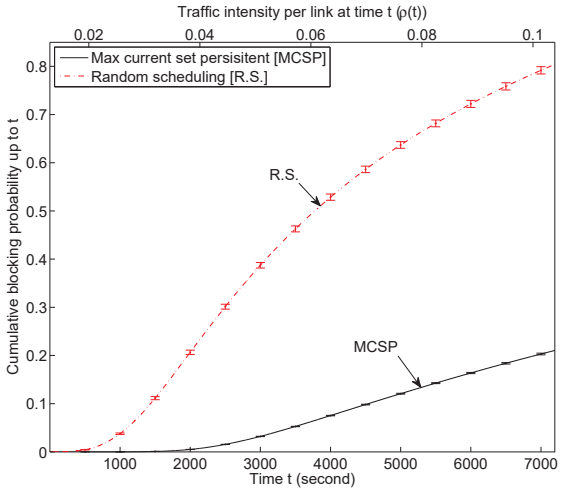

Fig. 6. Comparison of cumulative blocking probabilities with random scheduling for asymmetrical networks with different link congestion.

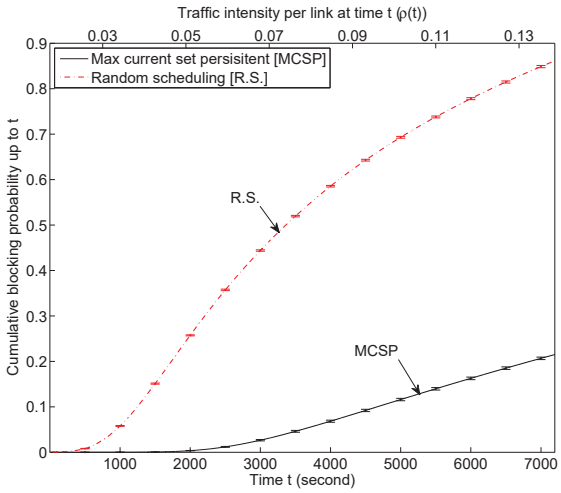

Fig. 7. Comparison of cumulative blocking probabilities with random scheduling for asymmetrical networks with different route length.

As described above, in the network with dynamic routing, each S-D has four feasible routes which are predetermined at the beginning of the simulation in a random fashion. Our *dynamic scheduler* with limited dynamic routing allocates wavelengths to the S-D pairs in a round robin fashion according to a predetermined order (selected randomly) as with the RR max-min fair scheduler). In each round and for each S-D pair, the scheduler *exhaustively searches* to find the shortest (hop count) available route from the four routes of that pair (as in [17]). The allocation preserves the persistence of the ongoing flows.

cumulative blocking probabilities of our previously persistent schedulers and that of the persistent dynamic scheduler, described above, are depicted in Fig. 8 for the network with asymmetric route length. The figures show that the freedom provided by dynamic routing (although over only four predetermined routes per S-D pair), improves the blocking probability dramatically. Moreover, comparing with the lower

bound on schedulers with static routing, depicted in Fig. 4, its cumulative blocking probability is very close to the bound.

Note that using the RR max-min fair scheduler with a set of multiple predetermined routes leaves the computational complexity of the scheduler intact, i.e., $|\mathcal{R}|^2$.

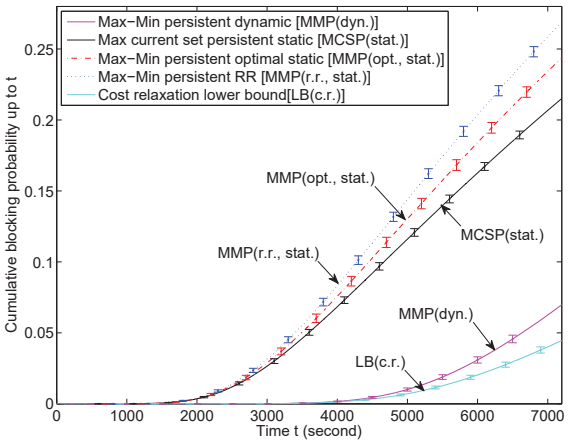

Fig. 8. Cumulative blocking probabilities with dynamic routing.

D. Fairness evaluation

Since fair lightpaths scheduling is of utmost importance in practice we examine the fairness of the most promising schedulers, i.e., the persistent max current set, the persistent optimal max-min fair and the persistent dynamic schedulers.

For every resource allocation vector $\mathbf{x}$ with n components, Jain's fairness index [38] is defined by

$$J(\mathbf{x}) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n\sum_{i=1}^{n} x_i^2}.$$

The index ranges from the worst case of $1/n$ to the best case of 1 when all allocations are equal. It is commonly perceived that the larger the index is, the fairer is the allocation.

In Fig. 9 we depict Jain's fairness index at every time t, where the vector $\mathbf{x}$ is taken as the cumulative flow blocking probabilities of all S-D pairs at time t. It can be observed that both, the persistent optimal max-min fair scheduler and the persistent dynamic scheduler are substantially fairer than the persistent max current set scheduler *for* $t > 2500$.

Before we move on to the conclusion, we provide some intuitive reasons for why one scheduler is expected to perform better than others. At the highest level, the analyzed schedulers are subdivided into persistent and nonpersistent schedulers. The advantage of persistent schedulers (over nonpersistent ones) is that the reserved lightpaths of the applications are guaranteed for the entire connection time (very important for TCP connections). The disadvantage of persistent schedulers is that lightpath allocations is more restrictive compared to allocation with nonpersistent schedulers. Thus, when comparing the cumulative blocking probabilities of the persistent and the nonpersistent versions of the same algorithm, that of the nonpersistent ones is expected to be lower (which is consistent

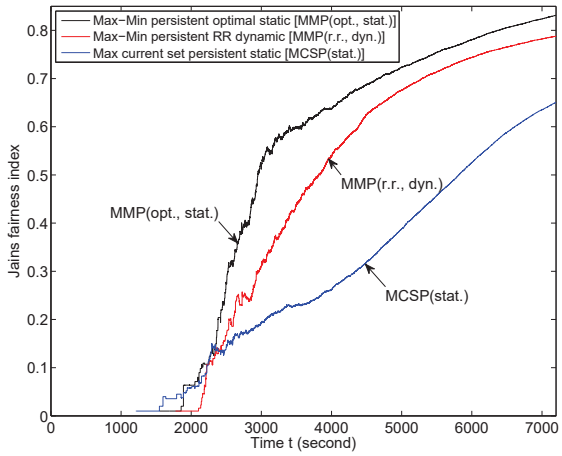

Fig. 9. Jain's fairness index for different schedulers.

with our findings except for MMNP(opt) and MMP(opt)). The exception with max-min fair scheduling using the optimal rule is that in each iteration a flow is selected using a myopic optimal rule rather than a long-term optimal rule.

Similarly to persistency, fairness also imposes a restriction on the lightpath allocation. Thus, the cumulative blocking probability of max-min schedulers are expected to be higher than that of the max current set scheduler, which allocates lightpaths as to maximize the current transmission set. However, this is only one aspect of the scheduling rule. Since both max-min fairness and max current set schedulers use myopic rules and persistency requires also long-term consideration, the relation between the max-min schedulers and max current set schedulers is hard to predict. In the symmetrical network depicted in Fig. 2, the performance shortcoming of fairness yields advantage for max current set. In the asymmetrical network depicted in Fig. 3, the performance shortcoming of persistency yields advantage for persistency max-min fair scheduler. In the asymmetrical network depicted in Fig. 4, both shortcomings impact the performance similarly.

IX. Conclusions

We have studied flow scheduling to lightpaths in an OFS architecture with time-dependent Poisson flow arrivals and Pareto distributed holding times. Scheduling decisions are taken at fixed-time intervals in a central node and scheduled flows cannot be interrupted before their completion. Unlike the OCS architecture, OFS uses a centralized scheduling node making the centralized resources allocations relevant.

The scheduling problem has been represented as a discrete-time Markov decision process with a flow blocking probability objective function over a finite horizon. We have derived three lower bounds for the objective function and proposed several schedulers, with and without fairness requirements. The schedulers performance have been evaluated under static and limited dynamic routing by emulating the algorithms on random network topologies. **The main finding** of our

performance analysis is that our proposed max-min fair scheduler with limited dynamic routing significantly outperforms all schedulers with static routing and achieves a blocking probability close to the lower bound of the schedulers whose routing is static.

The other findings are summarized as follows.

- Except for the case of asymmetric link congestion, nonpersistent schedulers bear almost no advantage compared to the persistent schedulers. In the case of asymmetric link congestion, the cumulative blocking probability of the nonpersistent RR max-min fair scheduler at the highest traffic load is 0.05 lower than the best persistent scheduler.
- The unfair persistent max current set scheduler is outperforming all other persistent schedulers. However, the cumulative blocking probability of the nonpersistent RR max-min fair scheduler is quite close to that of the unfair persistent max current set scheduler.
- The cumulative blocking probabilities in the asymmetrical layouts are higher than in the symmetrical layout, i.e., 0.22 vs. 0.048 at the highest offered load.
- The freedom provided by dynamic routing (although only four predetermined routes are available for each S-D pair), improves the cumulative blocking probability dramatically.
- Jain's fairness index shows that the persistent optimal max-min fair scheduler and the persistent dynamic scheduler are substantially fairer than the persistent max current set scheduler *for* $t > 2500$.

The OBS evaluation with the random scheduler above indicates that the throughput of OFS dominates that of OBS. However, if the applications requiring short set-up time then OBS has an advantage.

Additionally, based on the underlying architectures of OBS and OFS, the following conclusion can also be drawn. Although set-up time could present a concern in OFS for some applications, when using a persistent scheduler end-to-end packet delay is completely alleviated due to the end-to-end optical switching. On the other hand, set-up time with OBS is not a concern due its distributed algorithm; however, packet delay could be a concern since packets of a single TCP connection are possibly assembled in different bursts and when an OBS burst is lost many TCP connections are retransmitted resulting in slow start.

REFERENCES

[1] K. J. Barker, A. Benner, R. Hoare, A. Hoisie, A. K. Jones, D. K. Kerbyson, D. Li, R. Melhem, R. Rajamony, E. Schenfeld, S. Shao, C. Stunkel, and P. Walker, "On the feasibility of optical circuit switching for high performance computing systems," in *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, Seattle, Washington, USA, Nov. 2005.

[2] ITU-T Recommendation G.8080, "Architecture for the automatic switched optical networks (ASON)," Nov. 2001.

[3] A. Jaiszczyk, "Automatically switched optical networks: benefits and requirements," *IEEE Commun. Mag.*, vol. 43, no. 2, pp. S10–S15, Feb. 2005.

[4] D. G. Foursa, C. R. Davidson, M. Nissov, M. A. Mills, L. Xu, J. X. Cai, A. N. Pilipetskii, Y. Cai, C. Breverman, R. R. Cordell, T. J. Carvelli, P. C. Corbett, H. D. Kidorf, and N. S. Bergano, "2.56 Tb/s (256 times;10 Gb/s) transmission over 11,000 km using hybrid Raman/EDFAs with 80 nm of continuous bandwidth," in *Proc. OFC 2002*, Anaheim, CA, USA, Mar. 2002, paper FC3.

[5] L. Rau, S. Rangarajan, D. J. Blumenthal, H. F. Chou, Y. J. Chiu, and J. E. Bowers, "Two-hop all-optical label swapping with variable length 80 Gb/s packets and 10 Gb/s labels using nonlinear fiber wavelength converters, unicast/multicast output and a single EAM for 80- to 10Gb/s packet multiplexing," in *Proc. OFC 2002*, Anaheim, CA, USA, Mar. 2002, paper DF2.

[6] N. Wada, H. Harai, and W. Chujo, "Multi-hop 40 Gbits/s variable length photonic packet routing based on multi-wavelength label switching, waveband routing, and label swapping," in *Proc. OFC 2002*, Anaheim, CA, USA, Mar. 2002, pp. 216–217.

[7] C. Qiao and M. Yoo, "Optical burst switching (OBS) - a new paradigm for an optical Internet," *J. High Speed Networks*, vol. 8, no. 1, pp. 69–84, 1999.

[8] J. Turner, "Terabit burst switching," *J. High Speed Networks*, vol. 8, no. 1, pp. 3–16, 1999.

[9] I. Widjaja, "Performance analysis of burst admission-control protocols," *IEE Proceedings - Communications*, vol. 142, no. 1, pp. 7–14, 1995.

[10] V. W. S. Chan, G. Weichenberg, and M. Médard, "Optical flow switching," in *International Workshop on Optical Burst/Packet Switching (WOBS)*, San Jose, CA, USA, Oct. 2006.

[11] G. Weichenberg, V. W. S. Chan, and M. Médard, "On the capacity of optical networks: a framework for comparing different transport architectures," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 6, pp. 84–101, 2007.

[12] ——, "Design and analysis of optical flow-switched networks," *J. Optical Communications and Networking*, vol. 1, no. 3, pp. B81–B97, 2009.

[13] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: an approach to high bandwidth optical WANs," *IEEE Trans. Commun.*, vol. 40, no. 7, pp. 1171–1182, Jul. 1992.

[14] F. González, I. de Miguel, J. Blas, J. C. Aguago, P. Fernández, R. Durán, J. Durán, R. M. Lorenzo, E. J. Abril, and M. López, "Lightpath routing and wavelength assignment by means of ant colony optimization," in *Proceedings of Optical Design and Modelling Conference 2003 (ODMC 2003)*, Budapest, Hungary, Feb. 2003, pp. 855–864.

[15] E. W. M. Wong, L. L. H. Andrew, T. Cui, B. Moran, A. Zalesky, R. S. Tucker, and M. Zukerman, "Towards a bufferless optical Internet," *J. Lightw. Technol.*, vol. 27, no. 14, pp. 2817–2833, Jul. 2009.

[16] A. Birman, "Computing approximate blocking probabilities for a class of all-optical networks," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 5, pp. 852–857, Jun. 1996.

[17] A. Mokhtar and M. Azizoglu, "Adaptive wavelength routing in all-optical networks," *IEEE/ACM Trans. Netw.*, vol. 6, no. 2, pp. 197–206, Apr. 1998.

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

[19] Z. Rosberg, A. Zalesky, and M. Zukerman, "Packet delay in optical circuit-switched networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 341–354, Apr. 2006.

[20] Z. Rosberg, "Circuit allocation in all optical networks with average packet delay cost criterion," *IEEE Trans. Autom. Control*, vol. 51, no. 5, pp. 862–867, May 2006.

[21] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, New York, 1972, pp. 85–103.

[22] A. Zalesky, H. L. Vu, Z. Rosberg, E. W. M. Wong, and M. Zukerman, "Stabilizing deflection routing in optical burst switched networks," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 6, pp. 3–19, Aug. 2007.

[23] M. Duser and P. Bayvel, "Analysis of a dynamically wavelength-routed optical burst switched network architecture," *J. Lightw. Technol.*, vol. 20, no. 4, pp. 574–585, Apr. 2002.

[24] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 835–846, Dec. 1997.

[25] R. G. Addie, T. D. Neame, and M. Zukerman, "Performance evaluation of a queue fed by a Poisson Pareto burst process," *Computer Networks*, vol. 40, pp. 377–397, 2002.

[26] D. L. Jagerman, "Nonstationary blocking in telephone traffic," *Bell Syst. Tech. J.*, vol. 54, pp. 625–661, 1975.

[27] W. A. Massey, "The analysis of queues with time-varying rates for telecommunication models," *Telecommunication Systems*, vol. 21, no. 2, pp. 173–204, 2002.
[28] R. Stolletz, "Approximation of the non-stationary M(t)/M(t)/c(t)-queue using stationary queueing models: The stationary backlog-carryover approach," *European J. Operational Research*, vol. 190, pp. 478–493, 2008.
[29] R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.
[30] J. M. Kleinberg, "Approximation algorithms for disjoint paths problems," Ph.D. thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, USA, 1996.
[31] N. Robertson and P. D. Seymour, "Graph minors. XIII. The disjoint paths problem," *J. Combinatorial Theory*, vol. 63, no. 1, pp. 65–110, Jan. 1995, series B.
[32] D. P. Bartsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ USA: Prentice Hall, 1987.
[33] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow price proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, 1998.
[34] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.
[35] J. R. Giralt, "A theory of lexicographic optimization for computer networks," Ph.D. Thesis, Dept. EECE, University of California, Irvine, 2003.
[36] D. Eppstein, "Finding the *k* shortest paths," *SIAM Journal on Computing*, vol. 28, no. 2, pp. 652–673, 1998.
[37] I. Ouveysi, F. Shu, W. Chen, G. Shen, and M. Zukerman, "Topology and routing optimization for congestion minimization in optical wireless networks," *Optical Switching and Networking*, vol. 7, no. 3, pp. 95–107, Jul. 2010.
[38] D. M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1–14, 1989.