Double Queue Scheduler for Differentiated Setup Delay Virtual Optical Network Embedding over Elastic Optical Network

F. Mousavi Madani[1]

Received :2015\11\25 Accepted:2015\12\25

Abstract

Elastic optical network has recently raised immense research efforts to serve as a highly-flexible and scalable platform required by the emerging on-demand cloud applications. Besides, virtual network embedding on top of elastic optical network has been introduced as a promising solution to fine-grain sharing of node processing and optical bandwidth resources. Owing to the diverse delay sensitivity requirements of heterogeneous cloud applications, an efficient double queue scheduler is presented, to deliver differentiated setup delay embedding in a fair manner. Extensive numerical simulations carried out over two prototype substrate networks, demonstrated that our proposed scheme can concomitantly enhance blocking and fairness performance of the single queue scheduler over wide range of offered traffic loads.

Keywords: Elastic optical network; Fairness; Set-up delay tolerance.

Introduction

Flexible grid optical networks have recently been proposed as a promising solution to nurture insatiable bandwidth demand of applications such as Internet TV, telemedicine, e-science, social networking, 4G mobile services, cloud computing, 3D video on demand, and so forth presented to the backbone network. Flexi-grid optical network operates on underpinning Orthogonal Frequency Division Multiplexing (OFDM) technology which avails a fine-granularity wavelength grid of 12.5 GHz, 6.25 GHz or even smaller spacing. The vast pool of frequency slices, typically in excess of 300 to 400, can be devoted flexibly to heterogeneous needs of client applications, ranging from a few Gb/s up to several hundred Gb/s. Further, adoption of advanced modulation schemes into the fabrication of optical transponder modules, has enabled Elastic Optical Network (EON) to compromise spectral efficiency with the intended optical reach or Quality of Transmission (QoT). Given a required QoT and pre-computed physical distance between a pair of source and destination nodes, the highest modulation level is opted to make the most of spectrum resources.

On the other hand, network virtualization that has recently attracted massive attention as a promising solution to cope with the ossification of the existing Internet, allows multiple user network abstractions to be overlaid on a single substrate optical network to share the computing power of servers, and enormous available bandwidth of an optical fiber. In such a virtualized network environment, Service Provider (SP) leases physical resources from one or more Infrastructure Providers (InPs) to provision a Virtual Optical Network (VON) request issued by the client. Typically, a VON is comprised of two or more virtual nodes interconnected by virtual links. Each Virtual Node (VN) embodies a required computing capacity to be delivered by the substrate node counterpart (say a virtual machine in a data center), while a virtual link represents a required transmission bitrate between the pair of substrate node counterparts to be deployed as a lightpath traversing several Substrate Fiber Links (SFLs). The joint problem of mapping virtual nodes/links on substrate nodes/links, respectively, is commonly referred to as Virtual Network Embedding (VNE), and has raised intensive research interest among research communities. Even though, VNE over fixed-grid Wavelength Division Multiplexed (WDM) optical transport networks has so far been extensively studied in the literature, tight coupling of WDM optical device functionality to the underlying physical base of rigid frequency grid, restricts severely the agility and flexibility traits needed for on-demand VON provisioning. Quite recent studies have suggested that optical network virtualization over flexible-grid EONs can potentially provide efficient support to emerging cloud services and for the highly distributed and data-intensive applications such as petabits-scale grid computing [1, 2]. Consequently, VNE over EON tend to serve as the evolving platform for the future service-oriented networking paradigm.

As emerging customer-oriented network applications present different levels of grade of service requirements, the pressing need to support

[1].Assistant Professor, Department of Computer Engineering, Alzahra University, Tehran, Iran. mosavif@alzahra.ac.ir

differentiated service VON provisioning will become critical for network operators. Service differentiation opens up niche opportunity for SPs to elevate their revenue by boosting the utilization of otherwise underutilized network resources. The desired Service Class (SC), specified in Service Level Agreement (SLA) between costumers and SP, may be expressed in terms of the amount of time a user can wait before the request is provisioned, namely, *setup delay tolerance*, QoT, reliability or other quality metrics.

A. Research Themes

So far, the mainstream of research works on dynamic service provisioning context has been driven on underlying postulation that right after arrival of a request at some random instance of time, SP either serves the request by allocating to it the required amount of network resources, or else reject it immediately due to unavailability of network resource. However, a massive body of research momentum is recently attracted toward dynamic on-demand differentiated service provisioning in optical networks. Compilation of the main aspects of broadly diverse scheduling and service provisioning approaches, heretofore studied in the literature, would provide us the following insights.

First, regarding the classification of service requests, the majority of recent studies adopted setup delay tolerance for class differentiation, e.g. into three hard-real-time, real-time, and non-real-time classes, while only very few works have considered differentiated service classes in terms of signal quality (QoT). Provisioning strategy of the pending requests has lend itself on whether setup delay tolerances are solely known in advance or *connection holding-times* are also known as well. Having the former piece of information, the scheduler should allocate resources to a privileged request at the first occasion of resource availability before its delay tolerance expires since it cannot envisage future state of the network from the present state. When accompanied by the prior knowledge of connection holding-times, however, the scheduler can henceforth allocate resources at the best (not first) possible instance of time or even book-ahead those resources, formally referred to as *advance reservation*. A generic holding time aware service provisioning algorithm, however, should resort to much heavier computation power to find the optimal provisioning instance among the numerous possible alternatives. Moreover, the presumption of connection holding-time awareness, does not seem to be a valid proposition in most operational circumstances.

Management of queued requests should address issues like queue refreshment, ordering scheme, and rescheduling timing. Regarding the queue refreshment, two main approaches are pursued for populating the rescheduling queue with 1) any incident request irrespective of whether it can be immediately served or not, 2) only the request which cannot be served at the first trial. For the ordering scheme, queued requests are either reordered in ascending time of their arrivals or delay tolerances. Obviously, the latter scheme tend to place impatient requests in the head of queue, thereby hampering non-real-time requests to starve impatient request while in competition with them for grasping available resources. As a pending queued request is either served or eventually its remaining delay tolerance times out, pruning occasionally the served/expired requests as well as reordering and serving the active pending requests turned to be an event-driven process fired by whichever of arrival, departure or deadline events.

Connection setup strategy is characterized by probing scope and bundling size. In setup phase, connection requests may be handled sequentially one at the time, or bundled together for concurrent processing. In the former case, upon the onset of connection setup phase, the so-called *head-of-line probing scheme*, scans front entries of rescheduling queue to provision as many pending requests as possible. The process continues until either all pending requests are provisioned or it reaches an entry whose setup is impossible, in which case the setup phase stops its probing for further possible connection setups. Alternatively, the process may be continued until all entries of pending queue are examined.

B. Related Work

Authors of [3] proposed a QoS-aware optical connection setup management scheme that utilizes the Earliest Deadline First (EDF) queuing discipline to schedule the setup of blocked connections whereby the request having the smallest delay requirement is served first. Setup phase is activated upon the occurrence of departure and arrival events. They benchmarked their algorithm against bufferless and First-In-

First-Out (FIFO) queuing schemes by computing the overall blocking probability and distinct blocking probabilities of gold, silver, and bronze connection requests. In [4] an Integer Linear Program (ILP) model is formalized that exploited the expiration of set-up delay tolerance of the most impatient request to bundle the connection requests for concurrent processing with the aim of reducing spectrum fragmentation in the network. The performance of the ILP is assessed in terms of Bandwidth Blocking Ratio (BBR), defined as total blocked bandwidth (BW) divided by requested BW, and of network utilization entropy, which measures the level of resource fragmentation. The algorithm presented in [5] was taken into consideration the setup delay tolerance by selecting a group of connections with more number of requests and less mean of set-up delay tolerance that can be provisioned after a departure event in one round of the queue serving. Authors of [6] considered the differentiation of service classes in terms of signal quality. They presented three provisioning strategies, each took advantage in a different way the combined effect of using the setup delay tolerance and the holding-time awareness concept, to address the fairness problem. In *Dif-DTHT-B* strategy, blocked connection requests are rescheduled after the departure of provisioned one, without reserving any resources, if release of network resources can satisfy the signal quality target of the request. *Dif-DTHT-RR* strategy reserves resources while a connection request is rescheduled by using the information about holding time of provisioned requests. *Dif-DTHT-Mix* strategy incorporates the salient features of the other two, i.e., it applies the backoff strategy for SC-II and SC-III connections, while it utilizes the concept of advance reservation of network resources only for SC-I requests. The algorithm presented in [7] exploited connection holding-time knowledge to compute *k*-shortest paths between a source-destination using Dijkstra's algorithm where the cost of network links is varied inversely to the number of wavelengths entirely free within the arrival request holding-time window. Authors of [8] proposed six scheduling strategies to combine in a different way the concept of delay-tolerance and the connection holding-time awareness specifically tailored to facilitate the provisioning of the most stringent SC. Authors of [9] investigated service provisioning of requests with set-up delay tolerance taking into account holding time awareness of the requests in the context of EON. With the combination of a routing path selection policy and a request scheduling strategy, their algorithm constructed a weight matrix for each pending AR request and provisioned the request with it.

C. Our Contribution

The objective of this paper is to explore highly efficient scheduling algorithm that can drastically improve the success ratio of differentiated service embedding of VON requests onto a substrate EON in a fair way. The contribution of the paper is three fold. First, our rescheduling process takes advantage of all arrival, departure, and deadline event types to enhance both QoS and provisioning performances. Second, instead of reordering the pending requests in terms of their ascending time of arrivals or their initial delay tolerances, where a low priority but urgent request can be preceded by a high priority request of ample remained delay tolerance, the proposed ordering scheme invokes the actual remaining delay tolerances of pending VON requests. Third, rather than probing the whole rescheduling queue or just head-of-line entries, probing is applied to a subset of urgent requests whose remaining delay tolerances is about to expire. Having urgency of serving a request decided by its left over delay tolerance, not by its mere associated service class, turns to distribute network resources to service classes in a very equitable manner. We will verify that this novel double-queue scheduling design significantly outperforms the single-queue counterpart.

D. Paper Organization

The remaining sections are organized as follows. Section II presents mathematical formulation for optimal node and link mapping. Our contribution to scheduling algorithm is discussed in Section III, followed by a detailed numerical analysis and quantitative evaluations of the model presented in Section IV. Finally, Section V concludes the paper.

Mathematical Model

An elastic optical network (EON) topology is represented by a graph $G^s(V^s, E^s)$, where V^s is the set of substrate nodes (SNs) and E^s is the set of undirected substrate fiber links (SFLs). Each $v^s \in V^s$ is associated with $c^s_{v^s}$ available computing capacity and each SFL $e^s \in E^s$

accommodates up to B^s frequency slots (FS's). We consider an online VON provisioning scenario where an incident VON request is likewise modeled by an undirected graph $G^r(V^r, E^r)$, where V^r depicts the set of virtual nodes (VNs), and E^r represents the set of virtual optical links (VOLs). Each VN $v^r \in V^r$ is associated with $c_{v^r}^r$ computing resource requirement, while the Bandwidth (BW) requirement of each VOL $e^r \in E^r$ is denoted by $bw_{e^r}^r$ (in multiple units of subcarrier bit-rate).

In the course of VON embedding, both node mapping and link mapping processes are carried on simultaneously through the following mathematical formulation. For the node mapping operation, each VN of the VON request is mapped onto a unique SN that has sufficient computing capacity. We assume further that every SN can host at most one VN, if any. The link mapping procedure is nothing than a kind of RSA operation where a lightpath is set up in the physical infrastructure for each VOL to satisfy its BW requirement, a VOL BW should be carried by a subset of contiguous frequency slots among the ordered set of FS available slots under the spectrum non-overlapping and contiguity constraints [1].

In-line with modulation-level adaptability provided by Optical Orthogonal Frequency Division Multiplexing (OOFDM) scheme, we assume that a lightpath adopts the highest possible modulation index ($m \in MI = \{1, 2, 3, 4\}$) among the set of four schemes, BPSK, QPSK, 8QAM, and 16QAM, which can cover the physical distance of the lightpath. Based on the analytical estimation articulated in [10], transmission reach (TR) of 16QAM signal is preset to $R_{\max} = 375$km, whereby TRs of other modulation schemes can be simply derived as $R_{\max} \cdot 2^{4-m}$. Note that, hereafter, a path $p \in P$ refers to a transparent light path segment. Other employed notations in ILP formulation are listed below.

The next subsection introduces a path-based ILP formulation for the VONE over EONs following the recent work presented in [1].

A. Definitions

1) $Y \triangleq |E^r|$ Total number of VOLs in a VON request,
2) $M \triangleq |E^s|$ Total number of links in substrate EON.

B. Notations

1) $G^s(V^s, E^s)/G^r(V^r, E^r)$ Graph of physical network infrastructure/ VON request,
2) $c_{v^s}^s/c_{v^r}^r$ Computing capacity/resource requirement of each SN $v^s \in V^s$/VN $v^r \in V^r$,
3) B^s Total number of FSs on each SFL $e^s \in E^s$,
4) $w_{e^s,k}^s/z_{e^s,k}^s$ Starting/Ending FS index of the kth Maximal Contiguous Slot-Block (MCSB), which is a Contiguous Slot-Block (CSB) that embodies all consequent FS(s) at a spectral location on SFL e^s,
5) $s_{e^r}^r/d_{e^r}^r$ End-nodes of the VOL $e^r \in E^r$,
6) P^s Set of all pre-calculated directed paths in the substrate network $G^s(V^s, E^s)$,
7) $s_{p^s}^s/d_{p^s}^s$ Source/ Destination node of the path $p^s \in P^s$,
8) $P_{e^s}^s$ Set of routing paths that crosses the SFL e^s, obviously $P_{e^s}^s \subset P^s$,
9) $m_{p^s}^s$ Highest modulation-level for the path p^s,
10) $bw_{e^r}^r$ Normalized BW requirement of the VOL e^r in the VON request $G^r(V^r, E^r)$,
11) $n_{e^s}^s$ Number of available MCSBs on SFL e^s before FS assignment.

C. Variables

1) ξ_{v^r,v^s} Boolean variable that equals 1, if the VN v^r is mapped onto the SN v^s, otherwise it is 0,
2) ζ_{e^r,p^s} Boolean variable that equals 1, if a VOL e^r is mapped onto the substrate lightpath p^s, otherwise it is 0,
3) π_{e^r,e^s} Boolean variable that equals 1, if a VOL e^r uses the SFL e^s, otherwise it is 0,
4) $\sigma_{e_1^r,e_2^r}$ Boolean variable that equals 1, if two VOLs e_1^r and e_2^r use common SFL(s), otherwise it is 0,
5) $\rho_{e_1^r,e_2^r}$ Boolean variable that equals 1, if the starting FS index of the assigned CSB of the VOL e_1^r is smaller than that of the VOL e_2^r, otherwise it is 0,
6) w_{e^r} Integer variable that indicates the starting FS index of the assigned CSB for VOL e^r,
7) z_{e^r} Integer variable that indicates the ending FS index of the assigned CSB for VOL e^r,
8) $\delta_{e^r,e^s}^{(k)}$ Boolean variable that equals 1, if the kth MCSB on an SFL e^s is selected to serve the VOL e^r, otherwise it is 0,

D. Objective Function

Minimize

$$\sum_{e^r \in E^r} \sum_{p^s \in P^s} \zeta_{e^r,p^s} \cdot |p^s| \cdot \left\lceil \frac{bw_{e^r}^r}{m_{p^s}^s} \right\rceil \quad (1)$$

where $|p^s|$ denotes the hop count of p^s and the term $\left\lceil \frac{bw_{e^r}^r}{m_{p^s}^s} \right\rceil$ provides the number of FS' that is required to allocate to VOL e^r, when it is mapped onto the substrate lightpath p^s. The objective in equation (1) aims to minimizing the total number of consumed FS' for the incident *VON* request.

E. Constraints

Equations (2)–(17) formulate general requirements for node and link mapping, as articulated in [1] after subtle corrections.

$$\sum_{v^s \in V^s} \xi_{v^r,v^s} = 1, \ \forall v^r \in V^r \quad (2)$$

$$\sum_{v^r \in V^r} \xi_{v^r,v^s} \le 1, \ \forall v^s \in V^s \quad (3)$$

Equations (2) ensures that each VN in the *VON* request is mapped onto a unique SN, whereas equation (3) dictates each SN can host at most one VN, if any.

$$\sum_{v^s \in V^s} \xi_{v^r,v^s} \cdot c_{v^s}^s \ge c_{v^r}^r, \ \forall v^r \in V^r \quad (4)$$

Equation (4) entails that the hosting SN has enough available computing capacity to accommodate the VN.

$$\zeta_{e^r,p^s} \le \xi_{s_{e^r}^r,s_{p^s}^s} + \xi_{d_{e^r}^r,s_{p^s}^s}, \ \forall e^r \in E^r, \forall p^s \in P^s \quad (5)$$

$$\zeta_{e^r,p^s} \le \xi_{s_{e^r}^r,d_{p^s}^s} + \xi_{d_{e^r}^r,d_{p^s}^s}, \ \forall e^r \in E^r, \forall p^s \in P^s \quad (6)$$

$$\sum_{p^s \in P^s} \zeta_{e^r,p^s} = 1, \ \forall e^r \in E^r \quad (7)$$

$$\sum_{e^r \in E^r} \zeta_{e^r,p^s} \le 1, \ \forall p^s \in P^s \quad (8)$$

Equations (5)–(8) ensure that each *VOL* is mapped onto a single substrate lightpath, and the start/end nodes of the lightpath are the SNs that either the corresponding start/end or end/start VNs of that VOL are mapped onto,

$$\sigma_{e_1^r,e_2^r} \ge \zeta_{e_1^r,p_1^s} + \zeta_{e_2^r,p_2^s} - 1, \ \forall e_1^r, e_2^r \in E^r, \forall p_1^s, p_2^s \in P_{e^s}^s \quad (9)$$

$$\rho_{e_1^r,e_2^r} + \rho_{e_2^r,e_1^r} = 1, \ \forall e_1^r, e_2^r \in E^r \quad (10)$$

$$z_{e_2^r} - w_{e_1^r} + 1 \le B^s \cdot (1 + \rho_{e_1^r,e_2^r} - \sigma_{e_1^r,e_2^r}), \ \forall e_1^r, e_2^r \in E^r \quad (11)$$

$$z_{e_1^r} - w_{e_2^r} + 1 \le B^s \cdot (2 - \rho_{e_1^r,e_2^r} - \sigma_{e_1^r,e_2^r}), \ \forall e_1^r, e_2^r \in E^r \quad (12)$$

Equations (9)–(12) ensure that the spectrum assigned to any two *VOL*s, whose associated substrate lightpaths have common SFL(s) do not overlap.

$$z_{e^r} - w_{e^r} + 1 = \sum_{p^s \in P^s} \zeta_{e^r,p^s} \cdot \left\lceil \frac{bw_{e^r}^r}{m_{p^s}^s} \right\rceil, \ \forall e^r \in E^r \quad (13)$$

Equation (13) ensures that the number of FS' assigned to each VOL can just satisfy its BW requirement.

$$\pi_{e^r,e^s} = \sum_{p^s \in P_{e^s}^s} \zeta_{e^r,p^s}, \ \forall e^r \in E^r, \forall e^s \in E^s \quad (14)$$

$$\sum_k \delta_{e^r,e^s}^{(k)} = \pi_{e^r,e^s}, \ \forall e^r \in E^r, \forall e^s \in E^s \quad (15)$$

$$w_{e^r} \ge w_{e^s,k}^s \cdot \left(\delta_{e^r,e^s}^{(k)} + \pi_{e^r,e^s} - 1\right), \ \forall e^r \in E^r, \forall e^s \in E^s, \forall k \quad (16)$$

$$z_{e^r} - z_{e^s,k}^s \le B^s \cdot \left(2 - \delta_{e^r,e^s}^{(k)} - \pi_{e^r,e^s}\right), \ \forall e^r \in E^r, \forall e^s \in E^s, \forall k \quad (17)$$

Equations (14)–(17) ensure that the assigned CSB for each *VOL* locates in a single MCSB on SFL e^s, and the MCSB's size is not smaller than that of the assigned CSB.

Scheduling Algorithm

In the dynamic VON provisioning scenario postulated in this paper, VON requests issued by the higher level user applications arrive instantly into the scheduler in a sequence of random arrival events. Upon the arrival of an incoming VON request with a specified setup delay tolerance, the scheduler may come to the decision of either serving the VON request or otherwise enqueue it in a waiting queue of pending requests for subsequent treatment. On the other hand, termination of a served *VON* request followed by freeing up all corresponding occupied resources, may trigger the scheduler to serve a subset of pending requests. Specifically, the dynamism of incoming and outgoing events that govern the operation of our proposed scheduling algorithm are illustrated in Figure 1.

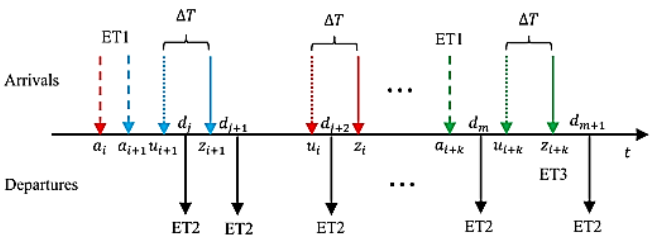


Fig. 1. Sequence and types of incoming and outgoing events occurring in scheduling algorithm

Notice that inward dashed-arrows represent the time instances $a_i, a_{i+1}, \ldots a_{i+k}$ when incoming requests initially arrive at the scheduler. This type of events are referred to as Event Type 1 (ET1) in

the following discussions. Outward solid-arrows however, indicate the time instances $d_j, d_{j+1}, d_{j+2} \dots d_m, d_{m+1}$where previously deployed VON requests are tore down and their associated allocated resources released. These outgoing events are labeled as Event Type 2 (ET2). Besides, inward dotted-arrows specify the time instances $u_{i+1}, u_i, \dots u_{i+k}$ where unserved requests $a_i, a_{i+1}, \dots a_{i+k}$ have to be urgently served within the left over set-up delay tolerance of ΔT seconds, otherwise they will be expired at time instances $z_{i+1}, z_i, \dots z_{i+k}$. Our proposed scheduler employs a succeeding Urgent Queue (UQ) that keeps up with the urgent subset of pending unserved requests stored in the preceding Waiting Queue (WQ). Normally, upon the occasion of a departure event and the follow-on evacuation of resources allotted to the terminated VON request, the scheduler is triggered to serve as many unserved VON requests in UQ as possible. This is possibly the case for the example VON requests $i + 1$ and i in Figure 1 at departure event instances d_j and d_{j+1}, respectively, since these requests were embraced in UQ beforehand. This mode of operation clearly applies to all pending VON requests where some departure event (ET2) happens within their urgent life-time period ΔT. In contrast, let us consider the case of VON request a_{i+k} in Figure 1. It is not appeared in UQ at neither d_m nor d_{m+1} instances, so it is simply discarded by the normal mode of operation triggered by ET2. The proposed scheduling algorithm could address these exceptions by defining another event, Event Type 3 (ET3), which triggers the scheduler just at the expiration times of those VON requests not experiencing any ET2 in their urgent life-time. This mode of operation is justified by observing that the most recent ET2 may have left adequate resources to serve the request at ET3.

The following subsection provides a closer look at different operational modes of the scheduler invoked by whichever of the above event types.

A. Event Type 1

As shown in Figure 2, upon the incidence of a new incoming VON request, depending on the status of WQ, the scheduler will choose either of trying to serve the request immediately regardless of setup delay tolerance or queuing it for subsequent actions. This behavior is attributed to the fact that when the substrate network is lightly loaded, intricate scheduling procedure for optimal resource allocation turns to be unjustified since all requests can be served instantly. Besides, all incoming requests are served without posing artificial setup delays which means higher QoS levels. Here, the status of WQ is assumed a faithful indication of substrate network load.

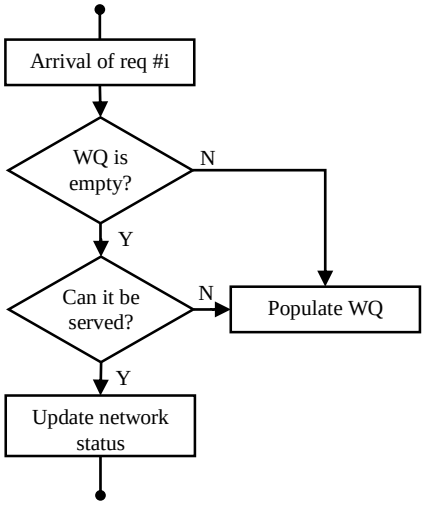


Fig. 2. Scheduler opertion invoked by Event Type 1

B. Event Type 2

The main scheduler operation is invoked by departure events as depicted in Figure 3. Each time an ongoing VON deployment departs from network and its allocated resources are released afterward, UQ is reconstructed by flushing urgent entries of WQ into it as discussed above. From now on, entries of UQ are examined one-by-one cyclically in ascending order of their left over setup delay tolerances until UQ eventually becomes empty.

C. Event Type 3

Those VON requests which do not experience any ET2 during their urgent period, would trigger the scheduler mode of operation shown in Figure 4 at their expiry instances. Upon the incidence of ET3, the scheduler tries to serve the VON request in the first place, otherwise the request is blocked since no remaining setup delay tolerances is left behind.

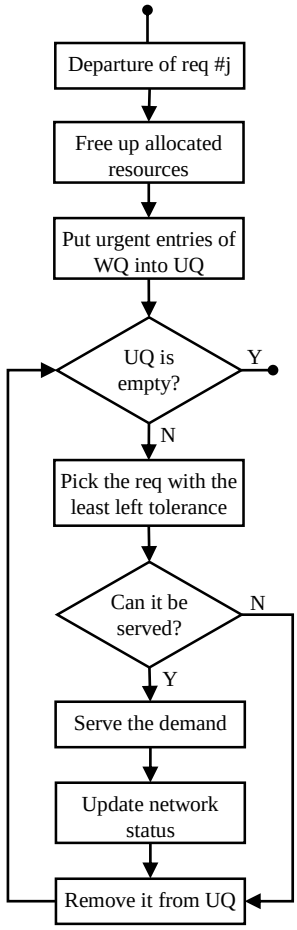


Fig. 3. Scheduler opertion invoked by Event Type 2

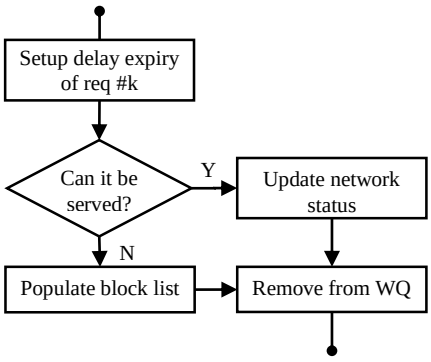


Fig. 4. Scheduler opertion invoked by Event Type 3

Performance Evaluation

In order to evaluate and verify the performance of the proposed double-queuing scheduler algorithm against the bufferless as well as the single-queue scheduler counterparts, we implemented a custom-made discrete event-based simulator using IBM ILOG CPLEX 12.6 development environment. The core of simulator is an even-driven engine that emulates the operation of the scheduler. It is also responsible for populating WQ and keep maintaining and updating it through the whole session of execution. On the occasion of a departure event, UQ is temporarily constructed and kept updated. Whenever a VON request is going to be served, the ILP solver is invoked and if fulfilled successfully, network status as well as relevant input variables of ILP model are accordingly updated.

We assumed that arrival pattern of VON requests can be represented by Poisson probability distribution while their holding time follow negative exponential distribution. Without loss of generality, the average holding time was fixed to 600 seconds, thereby the mean arrival rate of VON requests could be derived for the given offered load. For each VON request, the VON graph was randomly arranged by picking the number of VNs from a preset range, uniformly. Then, direct connection between two VNs was created with a probability of 0.5 which means that there would be $n(n-1)/4$ VOLs on average for a VON request with n VNs. The resulting graph was checked to be a connected graph, otherwise the graph generation process was reiterated. Likewise, required node computing capacities and required link bandwidths were picked randomly from a preset range of normalized values. Regarding the assignment of setup delay tolerance, a total of four predefined service classes were considered to address broad sensitivity variations in user applications. Instead of defining an absolute time delay, each service class designates the ratio of setup delay tolerance to the holding time of the associated VON request. The first service class represents “stringent real-time” applications with a very tight setup delay tolerance, such as online gaming, while the second and third ones represent “less time critical” applications, such as video and audio streaming, and the last class symbolizes “non-real-time” applications, such as critical data backup and grid computing. The service class of each VON request was then assigned randomly based on a predefined probability distribution profile.

Numerical simulations were carried on for a simple six-node and the 14-node NSFNET topologies, shown in Figure 5, as substrate test networks. All physical links of the six-node network have the same length of 100km, while the lengths (in km unit) of physical links in NSFNET topology are appeared beside the edges. For a given offered load, the blocking performance of the network under study was precisely figured out by aggregating the results over ten rounds of executions where each run was supplied by a distinct set of 10,000 random arrival and departure instances. Delay ratios of

the service classes I, II, III, and IV were set to 0.005, 0.02, 0.1, and 0.5, respectively, across the whole experiments. Table I lists other parameter settings.

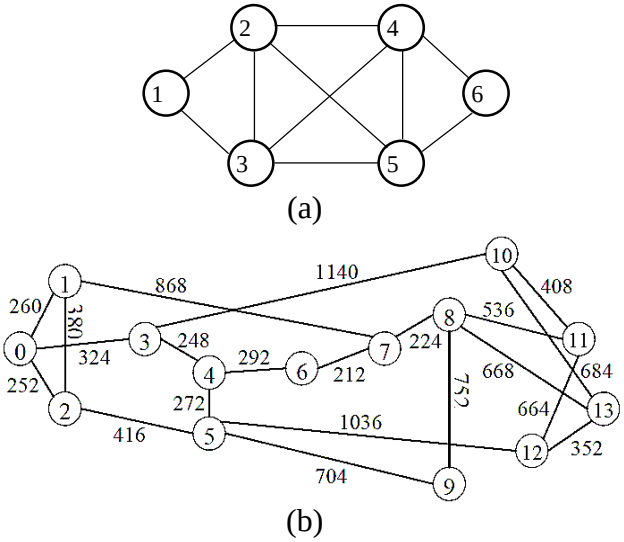


Fig. 5. (a) Six-node topology, (b) NSFNET topology

TABLE I. Simulation parameters

	Six-node	NSFNET
Number of SNs	6	14
Number of SFLs	10	21
Substrate nodes computing capacity	50 units	200 units
Fiber links BW capacity	50 FS'	400 FS'
Number of VNs in a VON	[2,3]	[2,3]
Number of VON requests	10×10,000	10×10,000
Virtual nodes computing requirement (units)	[1,4]	[1,4]
Virtual links BW requirement (units)	[1,12]	[1,12]

We proceeded, first, to scrutinize the impact of ΔT setting on blocking performance of the scheduler and to explore the optimum ΔT setting, if any, as the key design parameter. For the six-node network under study, the offered traffic load was set to 45 erlangs whereby ten different sets of random arrival and departure instances each comprising 10,000 entries as well as 10,000 random VON requests were prepared and stored a priori. In addition, three different sets of service class distributions, namely traffic mixes, {0.1, 0.2, 0.3, 0.4}, {0.25, 0.25, 0.25, 0.25} and {0.4, 0.3, 0.2, 0.1}, were considered here to reflect both uniform and non-uniform traffic mixes. Next, in accordance with pre-specified distributions, service classes were randomly assigned to VON requests and stored as separate data sets to be used in further simulation trials. The above sets of reference data files were then supplied, whenever applicable, to the whole simulation cycles to improve authentic comparison. As shown in Figure 6, $\Delta T \cdot \lambda$, where λ is the mean arrival rate of VON requests, started from as low as 0.15 and incremented exponentially to fairly large values. When ΔT approaches 0, urgent life time vanishes and UQ remains always empty at ET2 instances. In this case, the scheduler essentially behaves like a bufferless broker except that it has introduced certain amount of artificial delay to the incoming VON request before setting it up. In contrast, when ΔT grows large, UQ is no more than a duplicated copy of WQ, and the scheduler behaves the same as the single-queue scheduler. We observe that for all service class distribution scenarios, the optimum $\Delta T \cdot \lambda$ setting resides somewhere inside the limits, where double-queuing scheduler outperforms both bufferless with delay and single-queue schedulers. In line with the above analysis, the optimal $\Delta T \cdot \lambda$ values for other offered traffic loads were obtained and plotted in Figure 7. These plots also elucidate the fact that the optimal ΔT value keeps being proportional to mean inter-arrival time $(1/\lambda)$ over a wide range of offered loads.

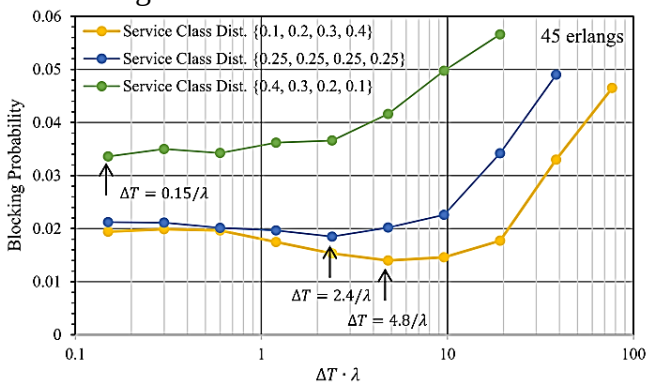


Fig. 6. Blocking performance of different service classes against $\Delta T \cdot \lambda$

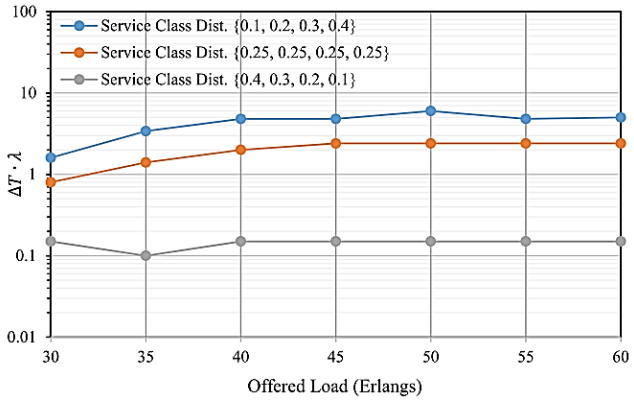


Fig. 7. Optimum $\Delta T \cdot \lambda$ of different service classes for different offered loads

Having found the optimal ΔT settings for the set of offered loads, blocking probabilities (BPs) of the proposed double-queuing scheduler versus offered load variation are plotted in Figure 8 for whichever of traffic mixes. The corresponding plots of bufferless broker accompanied with the single-queue scheduler are included for reference purpose. It is worth noting that while the single-queue scheme could somewhat improve the blocking performance of bufferless design, it operates still far behind that of the double-queuing scheme. Particularly, in the modest load region (35~45 erlangs), double-queue design could achieve some 3~6-folded performance improvement compared to the conventional single-queue one.

At this standpoint, one may argue that such improvement have possibly attained at the cost of exacerbating blocking rate fairness among service classes. To resolve this skepticism, BP of VON requests belonging to a given traffic load are separately found for different service classes of single-queue and double-queue schemes and normalized with respect to the SC-I BP of the single-queue scenario, as depicted in Figure 9. We observe that for all traffic mixes, BPs of SC-I classes of single-queue design are severely higher than low priority SC-III and SC-IV classes under most traffic load cases. In contrast, the proposed double-queue design, has not only improved individual BPs of most service classes, but at the same time, has achieved higher performance gains in SC-I and SC-II classes, which means an enhanced fairness property. This qualitative assertion was then verified by the so-called *Jain's fairness index*, as defined below, to measure the equality of resource allocation.

$$f_{BP} = \frac{\left(\sum_{i=1}^{N} bp_i\right)^2}{N \cdot \sum_{i=1}^{N} bp_i^2} \qquad (18$$

where N is the number of service classes and bp_i is the blocking probability of the ith class. f_{BP} is a number between 0 and 1, the closer to 1, the more equally the blocking probability is shared between N classes. Jain's fairness index of single-queue and double-queue schemes with respect to offered load are noted in Table II, for three traffic mixes.

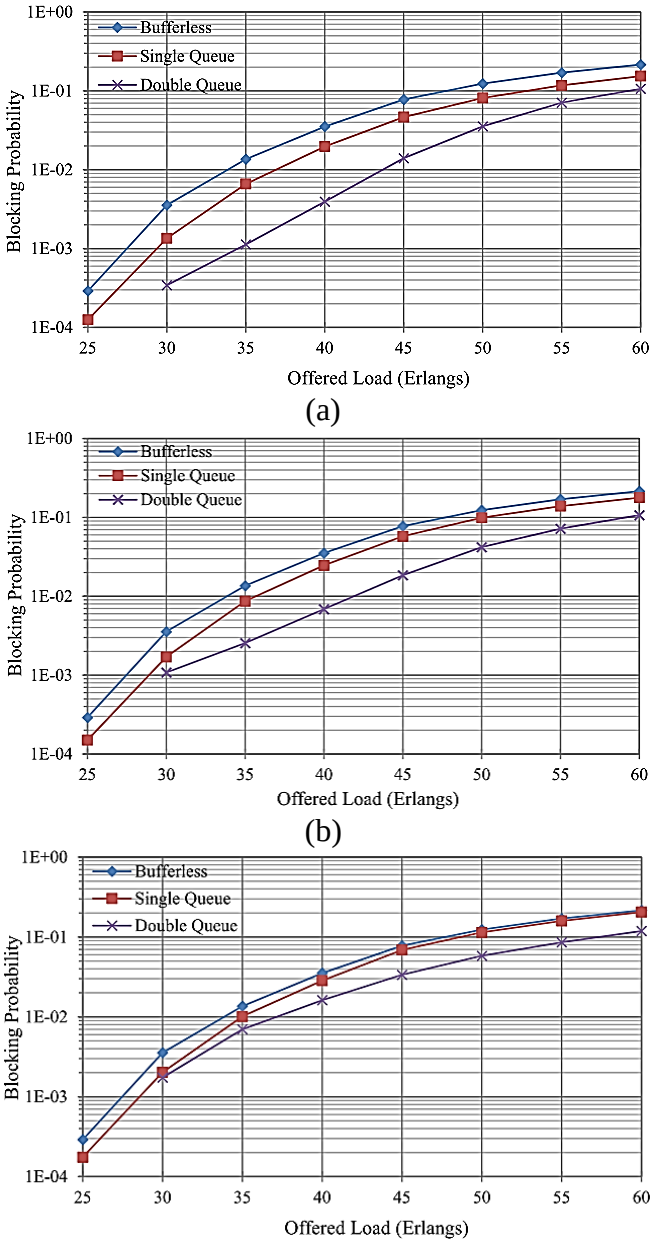
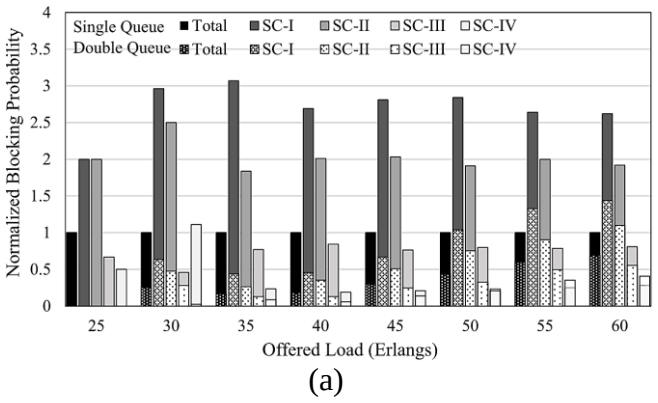
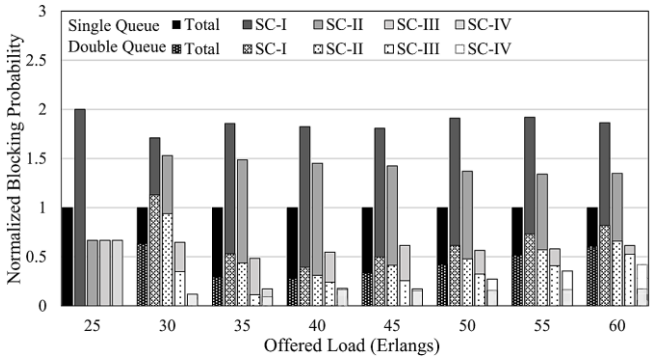
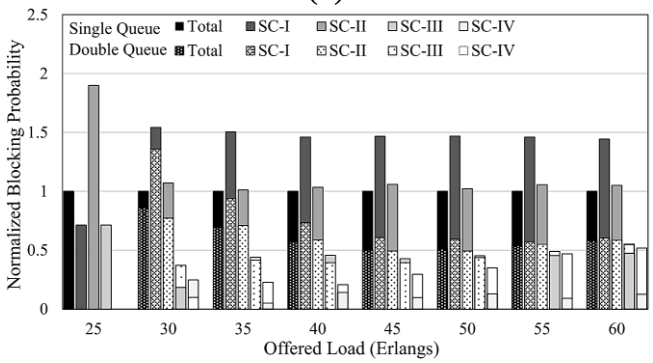


Fig. 8. Blocking performance of two scheduling designs for traffic mixes of (a) {0.1, 0.2, 0.3, 0.4}, (b) {0.25, 0.25, 0.25, 0.25}, and (c) {0.4, 0.3, 0.2, 0.1}





(b)



(c)

Fig. 9. Total and individual BPs of service classes of two scheduling designs for traffic mixes of (a) {0.1, 0.2, 0.3, 0.4}, (b) {0.25, 0.25, 0.25, 0.25}, and (c) {0.4, 0.3, 0.2, 0.1}

Table II. Jain's Fairness Index (%)

Erlangs	Traffic mix 1		Traffic mix 2		Traffic mix 3	
	Single	Double	Single	Double	Single	Double
25	76.75	-	75	-	59.8	-
30	75	71	71	70	64.02	65.3
35	65	74	67.6	70	64.94	81.54
40	68	70	69.4	91.7	69.69	85.34
45	66.8	78	70	87.3	67.18	93.73
50	67.4	75.27	68	91	68.98	96.60
55	69	80.5	68.7	92.5	67.72	99.34
60	70	81.7	70	94.2	69.77	99.65

Clearly, in almost all simulation trials, the double-queue scheme could provide superior fairness index than the single-queue one, in particular, fairness is remarkably enhanced in the last two scenarios. Similar experiments were carried out for NSFNET topology, with parameter settings depicted in rightmost column of Table I, set close to the actual operating conditions. Due to excessive execution time, we conducted numerical simulations just for the middle service class distribution. However, similar behavior will likely be observed if one runs other distributions. Blocking performance and Jain Fairness Index are plotted in Figures 10 and 11, respectively.

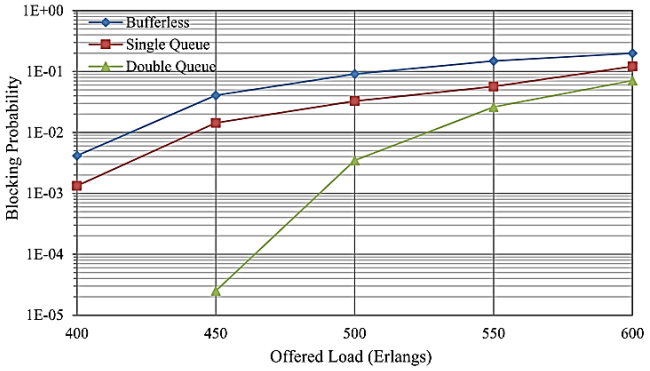


Fig. 10. Blocking performance of various scheduling designs for traffic mix of {0.25, 0.25, 0.25, 0.25}

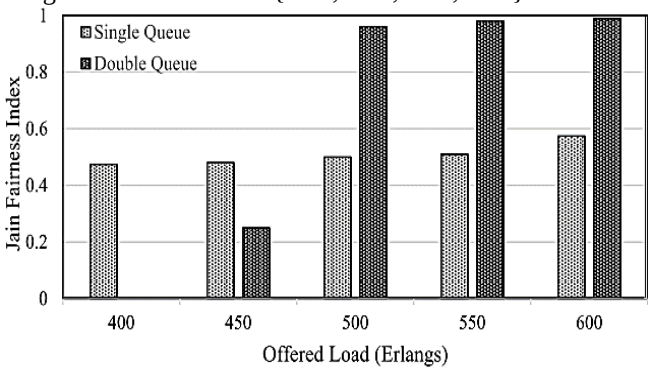


Fig. 11. Firness index of various scheduling designs for traffic mix of {0.25, 0.25, 0.25, 0.25}

Concluding Remarks

The scheduling algorithm proposed in section III works on triple scheduling events, where each event triggers refreshment and rescheduling of corresponding queues. The second queue functioned as a prioritizing queue that contributed in alleviating the overall blocking rate while leveling the success rate of service classes. Fine adjustment of urgent period could simply optimize the performance of scheduler for different service class distributions.

References

[1] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *Lightwave Technology, Journal of*, vol. 32, pp. 450-460, 2014.

[2] W.-H. Hsu and Y.-P. Shieh, "Virtual network mapping algorithm in the cloud infrastructure," *Journal of Network and Computer Applications*, vol. 36, pp. 1724-1734, 2013.

[3] W. Fawaz, "Improved EDF-based management of the setup of connections in opaque and transparent optical networks," *Photonic Network Communications*, vol. 27, pp. 8-15, 2014.

[4] A. Muhammad, M. Furdek, P. Monti, L. Wosinska, and R. Forchheimer, "An

Optimization Model for Dynamic Bulk Provisioning in Elastic Optical Networks," in *Asia Communications and Photonics Conference*, 2014, p. AF3E. 6.

[5] A. Mirhosseini, M. N. Soorki, and F. S. Tabataba, "Delay tolerance aware queueing and scheduling for differentiated services in dynamic survivable WDM networks," *Optical Switching and Networking*, vol. 17, pp. 1-13, 2015.

[6] A. Muhammad, C. Cavdar, and P. Monti, "Fair scheduling of dynamically provisioned WDM connections with differentiated signal quality," in *Optical Network Design and Modeling (ONDM), 2012 16th International Conference on*, 2012, pp. 1-6.

[7] A. Muhammad and R. Forchheimer, "Efficient scheduling strategies for dynamic WDM networks with set-up delay tolerance," in *Transparent Optical Networks (ICTON), 2013 15th International Conference on*, 2013, pp. 1-4.

[8] A. Muhammad, C. Cavdar, L. Wosinska, and R. Forchheimer, "Service differentiated provisioning in dynamic WDM networks based on set-up delay tolerance," *Journal of Optical Communications and Networking*, vol. 5, pp. 1250-1261, 2013.

[9] W. Lu and Z. Zhu, "Dynamic service provisioning of advance reservation requests in elastic optical networks," *Journal of Lightwave Technology*, vol. 31, pp. 1621-1627, 2013.

[10] M. Jinno, B. Kozicki, H. Takara, A. Watanabe, Y. Sone, T. Tanaka, *et al.*, "Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network [Topics in Optical Communications]," *Communications Magazine, IEEE*, vol. 48, pp. 138-145, 2010.