JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 40, 877-898 (2024)
DOI: 10.6688/JISE.202407_40(4).0012

Survivable Routing Problem on EONs with Slicers for DPP Scheme

DER-RONG DIN
Department of Computer Science and Information Engineering
National Changhua University of Education
Changhua City, 500 Taiwan
E-mail: deron@cc.ncue.edu.tw

In elastic optical networks (EONs), the slicing-and-stitching technique has emerged as a promising solution to reduce spectrum fragmentation by breaking the subcarrier consecutiveness constraint. This physical layer technology enables all-optical slicing of requests, allowing them to be accommodated in multiple non-consecutive spectral slots within an EON. In this paper, we investigate the survivable routing problem in EONs with slicers. Specifically, we consider the dedicated path protection (DPP) scheme to address single edge failures. For each connection request, we aim to find a pair of link-disjoint lightpaths to route the request. Additionally, we propose four heuristic algorithms to determine the routing paths based on two different slicing schemes, namely source and L-shape. Through extensive simulations, we evaluate the performance of the proposed algorithms and demonstrate their effectiveness in achieving satisfactory results.

Keywords: slicing-and-stitching, survivable routing, dedicated path protection (DPP), elastic optical network (EON), routing modulation level and spectrum assignment (RMLSA)

1. INTRODUCTION

In recent years, the rapid development of Internet requests and multimedia applications has posed serious challenges for optical transmission networks [1]. Furthermore, the recent advancements in cloud computing have highlighted the significance of big data applications operating on multi-data-center (multi-DC) systems in elastic optical networks (EONs) [2, 3]. The spectrum of an optical fiber in elastic optical networks (EONs) is divided into the small unit called frequency slot (FS). When assigning spectrum for a connection request, a consecutive set of FSs is allocated [4]. This requirement holds true for Routing and Spectrum Assignment (RSA) as well [5, 6]. To successfully route requests in EONs, two important constraints must be satisfied: spectrum continuity and subcarrier consecutiveness [4]. The spectrum continuity constraint ensures that the same FS is allocated to a request on each link of the routing path from source to destination [4]. The subcarrier consecutiveness constraint dictates that the FSs assigned to each request on each link must be contiguous. These constraints cause a fragmentation problem of spectrum [7] and reduce the efficiency of the spectrum band.

1.1 Slicing and Stitching Technique

Recent advancements in physical layer optics have enabled the slicing of optical bands into multiple sub-bands with different bandwidths [8-10]. This process involves creating a copy of the original spectrum band and filtering out unwanted signals. The remain-

Received May 11, 2023; revised June 9, 2023; accepted August 28, 2023.
Communicated by Mu-Yen Chen.

ing signal is then transmitted along the optical path to the destination. At the destination, the original signal is recovered (shown in Fig. 1). This technology is known as slicing and stitching [8, 9]. To implement this technology, a slicer is allocated at the source, while a stitcher is allocated at the destination. By utilizing the slicing and stitching technology, the subcarrier consecutiveness constraint in EONs can be relaxed [10].

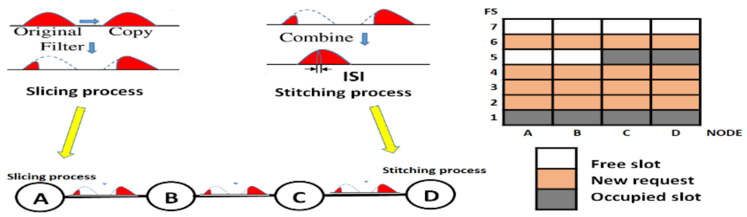


Fig. 1. Example of allocating request with slicing-and-stitching technology.

Fig. 1 illustrates an example of slicing a 4-FS request from node A to node D along the path A→B→C→D. In this case, consecutive four FSs from A to D are not available. At node A, a slicer divides the request into two partial spectra: one comprising a 1-FS component and the other comprising a 3-FS component. This request is then transmitted to node D, where it is recovered to its original signal.

Two routing and spectrum assignment (RSA) problems have been investigated for EONs with slicers: (1) The source slicing scheme [10-12], and (2) the L-shape slicing scheme [13-15]. In the source slicing scheme, the slicing process involves dividing the request bandwidth solely at the source node using slicers. In [11, 12], the authors proposed a scheme to determine the splitting position and the required number of FSs for each slice component. Additionally, in [10], a mixed-integer linear programming (MILP) model was introduced to address the RSA routing problem. The L-shape slicing scheme [11, 13-15] permits bandwidth slicing for requests at each EON node. In [11], the authors introduced two patterns of L-shape: a right-side-up L-shape and an upside-down L-shape. To address the problem, they proposed the largest L-shape fit allocation algorithm [11], which selects the largest size of the L-shape for the allocation.

These studies primarily focused on the RSA model in EONs. However, the more efficient Routing, Modulation Level, and Spectrum Assignment (RMLSA) model in EONs with slicers has not been investigated, as noted in the author's survey. In the RMLSA problem, with the use of Bandwidth-Variable Transponders (BVTs) and Bandwidth-Variable Wavelength Cross-Connects (BV-WXC) [5, 6], different modulation formats can be employed for transmissions based on the distance of the selected path, allowing for modulation changes. Choosing a higher modulation level reduces the number of required FSs for allocation and shortens the maximum transmission distance [5, 6].

An important aspect of the slicing technique, which can impact spectrum efficiency, was not considered in previous research. After each slicing, an additional guard band (GB) must be introduced to the extra block or mini-block, which may result in higher FS consumption compared to single-slice routing. Consider the example illustrated in Fig. 2, for the request : A-G, it requires 3 FSs and 1 GB. Considering the FS usage of the network shown in Fig. 2 (a), it is not possible to allocate the request using a single-slice block.

When applying source slicing, the request is divided into two slices along the path with indices 1-2 and 8-10 (as shown in Fig. 2 (b)). This allocation requires one slicer at node A and one stitcher at node G. Additionally, 6 extra GBs should be added to the slice with indices 8-10. Consequently, allocating request using source slicing necessitates 6 extra FSs compared to the single-slice allocation. For L-shape slicing, there are multiple possible cases depending on the selected slicing node and L-shape type. In the case of node C, an L-shape block (R1-1 with indices 1-4 on link A-C and indices 1-2 on segment C-G) and a mini-block (R1-2 with indices 7-9 on the path C-G) can be allocated (as shown in Fig. 2 (c)). This allocation requires 4 extra FSs to accommodate request r_1.

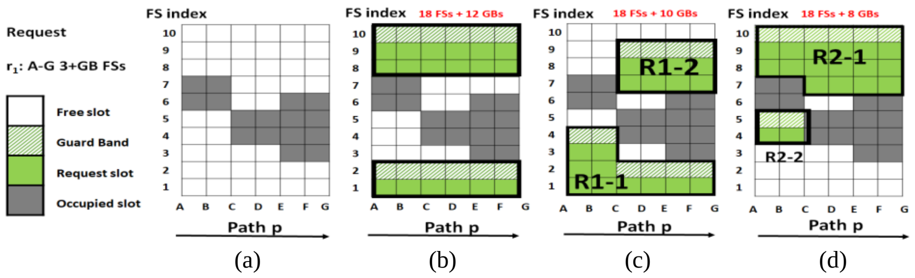


Fig. 2. FS allocation schemes; (a) Current FSs status; (b) Source slicing; (c) L-shape slicing on node C; (d) Upside down L-shape slicing on node C.

With upside-down L-shape slicing, various cases are possible based on the selected slicing node and L-shape type. For node C, an upside-down L-shape block (R2-1 with indices 8-10 on link A-C and indices 7-10 on segment C-G) and a mini-block (R2-2 with indices 4-5 on the path A-C) can be allocated (as shown in Fig. 2 (d)). This allocation requires 2 extra FSs to accommodate request r_1. From the above example, the L-shape slicing scheme utilizes fewer additional FSs compared to the source slicing scheme.

In [16], the authors proposed a scale-based policy to determine the required number of slicers for each node based on their investigation. The network's total number of slicers is predetermined, with each node having a different number of slicers based on the traffic it handles. It focuses on the RSA scheme for the given EON and utilizes the allocation methods described in [10, 14]. The authors of this paper examined the optimal number of slicers needed for each node and assigned the slicers accordingly.

In [17] a fragmentation-aware routing and spectrum allocation algorithm called SS-FA-RSA is proposed for the RSA model. In the routing selection phase of SS-FA-RSA, a path weight value formula is designed to reorder candidate paths and select the most appropriate routing path for the request. In the spectrum allocation phase, a spectrum slicing algorithm is proposed to enhance the utilization of spectrum fragmentation by reusing it in EONs. In [18], a limited number of slicers in an EON is proposed. Only selected nodes are equipped with slicers, and the allocation is based on their betweenness centrality (BC) ranking. The authors also observed that when the number of slicers in the network is fixed, the bandwidth blocking rate decreases as the number of slicers increases. However, it eventually starts to increase after reaching a minimum point. In [19], the authors analyzed three spectrum allocation (SA) policies for dynamic traffic: full slicing, partitioning, and partition with slicing. Simulation results indicate that the partition with slicing policy achieves a favorable balance between reducing spectrum fragmentation and minimizing wasted spectrum. This policy offers advantages in terms of efficient spectrum utilization while effectively managing spectrum allocation for requests of different sizes.

1.2 Survivable EONs

Network survivability is a critical issue that has been extensively studied in the context of EONS [20, 21]. Path-based protection [20] is a technique that employs a pair of link-disjoint primary and backup paths to route a request. In the event of a link failure on the primary path, the backup path is immediately utilized to transmit the request, ensuring 100% protection. Path-based protection techniques can be classified into two categories based on the shareability of protection capacity: dedicated path protection (DPP) and shared backup path protection (SBPP) [20]. DPP involves having dedicated backup capacity to protect the primary capacity. On the other hand, SBPP allows the protection capacity to be shared among multiple protection lightpaths, as long as their corresponding primary lightpaths do not fail simultaneously.

1.3 Studied Problem

When using slicers to split a request into multiple slices, the allocation of one or more additional GBs is required on the new blocks or mini-blocks. Consequently, employing more slicing processes will result in the allocation of more extra GBs and the wastage of FS resources. Unfortunately, these issues have not been discussed in previous research studies [10-14]. Furthermore, the more efficient and crucial RMLSA model, which considers the distance and modulation format of lightpaths in EONs with slicers, has not been studied. According to the author's survey, the problem of survivable routing using DPP for the RMLSA model in EONs with slicers has not been addressed.

This article focuses on the DPP survivable routing problem for the RMLSA model on EONs with slicers. The objective is to identify a pair of link-disjoint primary and backup lightpaths and allocate appropriate FSs to meet the traffic requirements of the survivable connection request in a given EON with slicers. To address this problem, two allocation algorithms were proposed: the Source Slicing Allocation Algorithm (SSAA) and the Modified L-shape Allocation Algorithm (MLAA). These algorithms aim to determine the FA allocation alone on a given path, taking into account the additional GBs required in the EON for the RMLSA model. Additionally, four heuristic algorithms were proposed to find the survivable routing paths of the request, considering the two slicing schemes.

2. PROBLEM DEFINITION

The following section presents the notations, assumptions, and performance criteria used in the study of the problem.

2.1 Notations

$G = (V, E, D)$: the physical topology of the EON,where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of nodes ($|V| = n$), $E = \{e_1, e_2, \ldots, e_m\}$ represents the set of optical links ($|E| = m$), and $D(e_i)$ represents the distance of link $e_i \in E$. $r = (s, d, B)$: the connection request, where

$s \in V$ and $d \in V$ represent the source and destination nodes of the request r, respectively. B represents the required bandwidth (Gb/s) of the request. F: the number of FSs provided by each link of the network. $SN(v_i)$: the number of free slicers available on the node $v_i \in V$. $SN(v_i)$: the number of free stitchers available on the node $v_i \in V$. $free(e_l)$: the total number of free FSs available on link $e_l \in E$. $M = \{1, 2, 3, 4, 5, 6\}$: the set of possible modulation levels (MLs) corresponding to the set of modulation formats {BPSK, QPSK, 8-QAM, 16-QAM, 32-QAM, 64-QAM}. $ML(p)$ represents the best modulation level for path p. The respective set of maximum transmission distances (or transparent reach, TR) is {250, 500, 1000, 2000, 4000, 8000}.

To accommodate the request r, our objective is to identify a pair of link-disjoint paths, each capable of providing the required bandwidth of Gb/s. For the request, the allocated bandwidth can be assigned to different FS indices on the selected path, given the availability of slicers and stitchers. In the event of a link failure on the primary path, the backup path can ensure the provision of the required bandwidth.

2.2 Assumptions

The assumptions of the studied problem are as follows: (1) Each link is equipped with a fiber that enables bidirectional signal transmission; (2) The numbers of FSs provided by each link is the same; (3) A GB is allocated between two lightpaths; (4) The network is a 2-connected graph, and only the single-edge failure scenario is considered in this article; (5) The DPP scheme and RMLSA routing model are taken into account.

2.3 Performance Criteria

Two key metrics are utilized: the *bandwidth blocking rate* (BBR) and the *protection resource ratio* (PRR). The BBR is determined by calculating the ratio of the rejected bandwidth to the total requested bandwidth. It serves as a measure of the system's effectiveness in accommodating the requested bandwidth. The PRR is calculated as the ratio of the total number of backup resources to the total number of primary resources.

3. FS ALLOCATION AND SLICING ALGORITHM

In this section, we describe two FS allocation algorithms for source slicing and L-shape slicing.

3.1 FS Allocation of Source Slicing Scheme

For the source slicing scheme, the slicing process is applied at the source node, while the stitching process is applied at the destination node. The request may be served by several slices on the same routing path, with each slice potentially having different FS indices. The set of candidate FS slices of the selected path p is denoted as $FreeBlock(p) = \{(size_j, idx_j), j = 1, 2, \ldots, x_p\}$, where x_p represents the number of slices, and idx_j and $size_j$ represent the starting index and size of the jth slice, respectively. If the request r is routed using a single slice, the minimum number of required FSs on the lightpath p is denoted as $N(B, p)$ and can be computed as $N(B, p) = \lceil B/(12.5 \times ML_{opt}(p)) \rceil + GB$. Here, $ML_{opt}(p)$ represents the highest modulation level of the path. If the request r is supported by multiple slices (x

> 1), the minimum number of required FSs on the path is computed as $N(B, p) = \lceil B/(12.5 \times ML_{opt}(p))\rceil + x \times GB$. Consider the example shown in Fig. 2 (a), where the primary path of the request r_1: A-G (requiring 3 FSs plus 1 GB) is on the path p. In this case, we can find $FreeBlock(p) = \{(3,8), (2,1)\}$ and $\sum_{j=1}^{2} size_{j=1}^{p} = 3 + 2 \geq 5$.

To determine the allocation of FSs on the path p, the blocks in $FreeBlock(p)$ are sorted in descending order based on their size $size_j^p$. If the request r can be routed using a single slice, the slice with the minimum size that is greater than or equal to $N(B, p)$ is selected. Alternatively, if there are available slicers at the source node s and stitchers at the destination node d (*i.e.*, $NS(s) > 0 \wedge ST(d) > 0$), the request may be routed using the source slicing scheme. In this case, the block in $FreeBlock(p)$ with the maximum size are selected first to be allocated, prioritizing larger blocks to reduce the required number of slicers and stitchers. The time complexity of the Source Slicing Allocation Algorithm is $O(hop(p) \times F + x_p\log x_p)$, where $hop(p) \leq n$ represents the number of hops in the path p, and x_p is the number of slices in the network. As $x_p \leq F/2$, the worst-case time complexity becomes $O(hop(p) \times F + F\log F)$. The details of the **Source Slicing Allocation Algorithm** are described in Algorithm 1.

Algorithm 1: Source Slicing Allocation Algorithm

```
Input: $G(V,E,D)$, the request $r = (s,d,B)$, path $p$, $NS(s)$, $ST(d)$;
Output: the set $AL$ of FS allocation on path $p$ for the request;

Find $FreeBlock(p) = \{(size_j^p, idx_j^p), j = 1, 2, ..., x_p\}$ on $p$.
Sort the blocks in $FreeBlock(p)$ in descending order based on size $size_j^p$ .
Initialize $AL$ as an empty set and $C$ as $N(B,p)$.
while ($C > 0 \wedge FreeBlock(p) \neq \emptyset$ ) do
    {
    if ($size_1^p \geq C$) then
        Find the maximum index $j$ such that $size_j^p \geq C$.
        Allocate $C$ FSs on the $j^{th}$ slice with a starting index of $idx_j^p$.
        Add $(C, idx_j^p)$ to $AL$ and return $AL$.
    else if ($NS(s) > 0 \wedge ST(d) > 0$) then
        Select the first $j = 1$ block.
        Allocate $size_1^p$ FSs starting from index $idx_1^p$, and set $C = C - size_1^p + 1$ //1 GB is added.
        Add $(size_1^p, idx_b^p)$ to $AL$.
        Remove the first block $b_1$ from $FreeBlock(p)$ and readjust the order of the remaining blocks in $FreeBlock(p)$.
        $NS(s) = NS(s) - 1$, $ST(d) = ST(d) - 1$.
    end if
    }
end while
if $C > 0$ then
    return $\emptyset$;
else
    return $AL$;
end if
```

3.2 FS Allocation of L-shape Slicing Scheme

For the request $r = (s, d, B)$, if path p is selected and node i is the slicing node, the jth block with $size_j^p$ is selected, the path p is divided into two segments $seg_p(s, i)$ and $seg_p(i, d)$. When using the L-shape slicing, the total number of required FSs can be computed as $N(B, p) \times hop(p) + hop(seg_p(i, d))$, as shown in Fig. 3 (a). On the other hand, when using the source slicing, the total number of required FSs can be computed as $(N(B, p)+1) \times hop(p)$, as shown in Fig. 3 (b). Since $hop(seg_p(i, d)) \leq hop(p)$, the total number of required FSs for L-shape slicing is smaller than that of source slicing. It's important to note that both of these schemes may consume more resources compared to single-slice routing, which requires $N(B, p) \times hop(p)$ FSs, as shown in Fig. 3 (c).

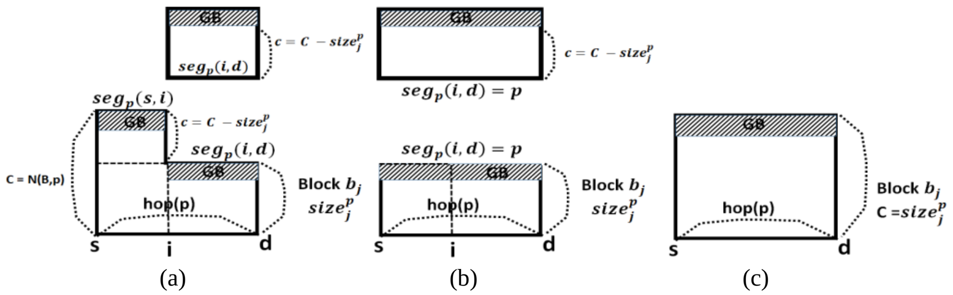


Fig. 3. FS allocation schemes; (a) L-shape slicing; (b) Source slicing; (c) Single slice.

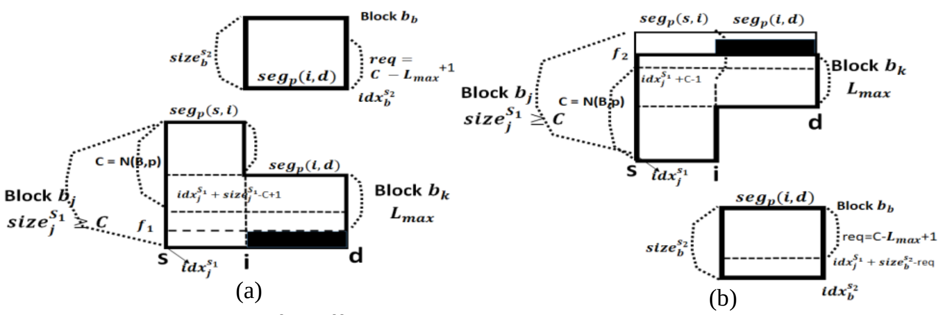


Fig. 4. FS allocation for different L-shapes; (a) L-shape (b); upside down L-shape.

To determine a feasible L-shape slice and mini-block for routing the request on the selected path, the selection of the slicing node, the available blocks on the path, and the type of L-shape and mini-block are crucial factors that can impact the final selection. Finding the best L-shape option can be a time-consuming process. In the proposed algorithm, a first-fit approach is utilized to find and return the first feasible L-shape slice and mini-block combination.

The details of the **Modified L-shape Allocation Algorithm** are described in Algorithm 2. The algorithm focuses on finding the L-shape slice and mini-block. The notation and concept of the algorithm are illustrated in Fig. 4 (a), which shows the process for finding the L-shape slice. The algorithm for finding the upside-down L-shape slice can be easily extended, and its notation and concept are illustrated in Fig. 4 (b). For the selected intermediate node i, the path p is divided into two segments $s_1 = seg_p(s, i)$ and $s_2 = seg_p(i, d)$. Then, the sets $FreeBlock(s_1)$ and $FreeBlock(s_2)$ are then obtained as the candidate sets of blocks.

To find the L-shape block, each block in $FreeBlock(s_1)$ is examined to check if it has enough free FSs (C). If the starting index of the L-shape block is f_1, the maximum size L_{max} of $s_2 = seg_p(i, d)$ is determined. The remaining required FSs, denoted as $req = C - L_{max} + 1$, are checked in the set $FreeBlock(s_2)$. Additionally, the starting index of the mini-block should be greater than $f_1 + L_{max}$. The time complexity of the Modified L-shape Slicing Allocation Algorithm is $O(hop(p)\times F + x_p\log x_p + hop(p)\times(x_p^2))$. Thus, the worst-case time complexity is $O(hop(p)\times F + F\log F + hop(p)\times F^2) = O(hop(p)\times F^2)$, considering $x_p \leq F$.

4. SURVIVABLE ROUTING ALGORITHMS FOR DPP SCHEME

In this section, four heuristic algorithms are proposed: the *Max-Flow Disjoint Path Routing Algorithm* (MFDPRA), the *Alternate Disjoint Path Routing Algorithm* (ADPRA), the *Semi-Dynamic Path Routing Algorithm* (SDPRA), and the *Dynamic Path Routing Algorithm* (DPRA). The First-Fit (FF) FS allocation is applied for each algorithm. In the worst case, ADPRA, SDPRA, and DPRA algorithms consider $K \times K$ possible pairs of paths. For the MFDPRA algorithm, there are $MF \times (MF - 1)/2$ possible pairs, where MF is the maximum number of edge-disjoint paths between the source and destination. In the proposed methods, the first selected pair of paths with an available FS allocation is considered the final decision for the request.

Algorithm 2: Modified L-shape Allocation Algorithm

```
Input: $G(V,E,D)$, the request $r = (s,d,B)$, path $p$;
Output: the set $AL$ of segments, FS-indices allocation, and sizes of the request;

$C = N(B,p)$, $AL = \emptyset$.
Find $FreeBlock(p) = \{b_j = (size_j^p, idx_j^p), j = 1,2,...,x_p\}$ on path $p$.
Sort blocks in $FreeBlock(p)$ according to $size_j^p$ in descending order.
// Single slice first
if ($C$ FSs can be allocated on path $p$ without using slicing and stitching process) then
    // $\exists j \in \{1,2,...,x_p\}$ $size_j^p \geq C = N(B,p)$
    Select block $b_j$ with maximum $j$ and with $size_j^p \geq C$.
    return $AL = \{(p, C, idx_j^p)\}$.
else
    if ($hop(p) < 2$) then
        Perform Algorithm 1 Source Slicing and return result.
    else
        //Using L-shape slicing
        for (all possible intermediate node $i \in p$) do
            Divide path $p$ into two segments $s_1 = seg_p(s,i)$ and $s_2 = seg_p(i,d)$.
            Find $FreeBlock(s_1)$ and $FreeBlock(s_2)$.
            while (there exist slice $b_j \in FreeBlock(s_1)$ with $size_j^{s1} \geq C$) do
                Select block $b_j = (size_j^{s1}, idx_j^{s1}) \in FreeBlock(s_1)$.
                for (index $f_1 \in [idx_j^{s1}, ind_j^{s1} + size_j^{s1} - C + 1]$) do
                    Find block $b_k \in FreeBlock(s_2))$, such that $f_1 \in [idx_k^{s2}, idx_k^{s2} + size_k^{s2}]$, and with maximum $L_{max} = idx_k^{s2} +$
                    $size_k^{s2} - f_1$.
                    if ($L_{max} > 0$) then
                        Compute $req = C - L_{max} + 1$.
                        Find slice $b_b \in FreeBlock(s_2))$, $b_b \neq b_k$, and $idx_b^{s2} > f_1 + L_{max}$.
                        if ($size_b^{s2} \geq req$) then
                            Add $(s_1, C, f_1)$, $(s_2, L_{max}, f_1)$, and $(s_2, req, idx_b^{s2})$ to $AL$), return $AL$.
                        end if
                    end if
                end for
            end while
        end for
        Perform a similar algorithm to find the upside-down L-shape and mini-block on path $p$ to route the request $r$ and and
        obtain $AL$.
        if ($AL \neq \emptyset$) then
            return $AL$.
        else
            Perform Algorithm 1 Source Slicing Allocation Algorithm and return result.
        end if
    end if
end if
```

4.1 Max-Flow Disjoint Path Routing Algorithm (MFDPRA)

In this subsection, we propose the Max-Flow Disjoint Path Routing Algorithm (MFD

PRA) and the provide details on the Max-Flow computation. First, we compute a pre-computed set P_{sd} of link-disjoint paths using the Ford-Fulkerson algorithm (FFA) [23]. These paths serve as the candidate paths for routing requests. The paths in P_{sd} are then sorted in increasing order based on the value $N(B, p) \times hop(p)$, where p belongs to P_{sd}. Next, we select two paths from the set P_{sd} to be evaluated as the primary and backup paths for the request. The time complexity of the MFDPRA is $O(nm^2)$, where n is the number of nodes and m is the number of links.

4.2 Alternate Disjoint Path Routing Algorithm (ADPRA)

In this subsection, we propose the Alternate Disjoint Path Routing Algorithm (ADPRA) and provide details on its implementation. First, we compute a pre-computed set $P_{sd} = \{(p_p^i, p_b^i), i = 1, 2, \ldots, K^2\}$ consisting of link-disjoint path pairs. This set represents the candidate pairs of paths for routing. To construct P_{sd}, we perform the k-shortest path algorithm [24] on the network graph $G(V, E, D)$ to find a set of K paths $\{p_p^i, i = 1, 2, \ldots, K\}$. For each path p_p^i, we construct the graph $G^i(V, E, D)$ by removing all links in p_p^i from $G(V, E, D)$ and apply the k-shortest path algorithm on $G^i(V, E, D)$ to find the backup path p_b^{ij} for p_p^i, where $j = 1, 2, \ldots, K$. If the backup path p_b^{ij} is found, we add the pair of paths (p_p^i, p_b^{ij}) to the set P_{sd}. For each selected path p_p^i and backup path p_b^i we determine the best modulation level of the path, denoted as $ML_{opt}(p_p^i)$ and $ML_{opt}(p_b^{ij})$, respectively.

In ADPRA, all path pairs $(p_p^i, p_b^{ij}) \in P_{sd}$ are sorted in increasing order based on the metrics $N(B, p_p^i) \times hop(p_p^i) + N(B, p_b^{ij}) \times hop(p_b^{ij})$, where $hop(p_p^i)$ represents the number of hops in path p_p^i. Path pairs in P_{sd} are examined and selected in the order they appear in the sorted list as primary and backup paths. The time complexity to find the set of link-disjoint paths is $O(Kn(m + n\log n))$.

4.3 Dynamic Path Routing Algorithm (DPRA)

In this subsection, the dynamic path routing algorithm (DPRA) that considers link spectrum usage on the fly is described. For a given request $r = (s, d, B)$, we define $REQ(r)$ as the minimum number of FSs that should be allocated on the network to accommodate the request. We assume that a single slice is used to route the request r. For the shortest path p from source node s to destination node d on the original network graph $G(V, E, D)$, we have $REQ(r) = N(B, p)$. This idea is inspired by a study conducted by Zhu *et al.* [25], where they proposed a dynamic service provisioning algorithm that incorporates a hybrid single-/multi-path routing scheme on EONs with the RMLSA model. To search for the set of candidate paths for the request, we construct a new graph $G'(V, E, d')$ based on link spectrum usage, where d' represents the weight of link $e \in E$ and is defined as follows,

$$d'(e) = \begin{cases} +\infty & \text{if } free(e) < REQ(r) \\ w(e) \times \frac{F - free(e) + 1}{F} & \text{otherwise} \end{cases} \tag{1}$$

where $free(e)$ returns the total number of free FSs of link e, and ML_{max} is the highest modulation level supported in the network, the weight $w(e)$ of link e is calculated from $d(e)$ as $w(e) = ML_{max} - ML_{opt}(e) + 1$ [25]. In the new network graph $G'(V, E, d')$, a link $e \in E$ is omitted from the path finding if $free(e) < REQ(r)$. Otherwise, the link weight $d'(e)$ is proportional to the product of $w(e)$ and the number of used FSs $F - free(e) + 1$.

The set P_{sd} contains K candidate paths, which are found by performing the k-shortest path algorithm on the new graph (V, E, d'). Paths in P_{sd} are sorted in increasing order according to the new distance d' and are examined in order. If the primary path can be found and the required number of FSs can be allocated by performing the Modified L-shape Algorithm, then the resources for the primary lightpath are temporarily allocated, and a new graph is constructed by removing all links on the primary path. Then, the Dynamic Backup Path Finding Algorithm (DBPFA) is performed to find the backup path for the primary path. If both the primary and backup lightpaths can be found, then these paths are recorded. Otherwise, another possible primary path is selected, and the DBPFA is performed again to find the backup lightpath. The details of the DPRA and the DBPFA are described in Algorithms 3 and 4, respectively.

4.4 Semi-Dynamic Path Routing Algorithm (SDSPRA)

In this subsection, the Semi-Dynamic Path Routing Algorithm (SDPRA) is proposed. First, a set P_{sd} of candidate paths is pre-constructed by performing the K-shortest path algorithm [24] on $G(V, E, D)$, and it will be used to find the primary path. The paths in P_{sd} are sorted in increasing order according to the cost $N(B, p) \times hop(p)$ of path $p \in P_{sd}$, and then the paths are examined in order. If the examined path can be allocated on the network, the primary lightpath is temporarily allocated, and then the link-disjoint backup path is dynamically found. The details of the SDPRA are described in Algorithm 5.

Algorithm 3: Dynamic Path Routing Algorithm (DPRA)

```
1: **Input:** $G(V,E,D)$, the request $r = (s,d,B)$;
2: **Output:** the primary path and backup paths of the request;
3: Collect and update link status of $G(V,E,D)$.
4: Construct new graph $G'(V,E,d')$ and update link weights $d'(e), \forall e \in E$ using (1).
5: Construct the set $P_{sd}$ by performing the K-shortest path algorithm on network $G'(V,E,d')$ from $s$ to $d$.
6: Sort the paths in $P_{sd}$ in increasing order according to the weighted distance $\sum_{e \in p} d'(e)$ of path $p \in P_{sd}$.
7: $Cost_{opt} = \infty$, $PP^{opt} = \emptyset$;
8: for (all possible primary path $p_1$ in $P_{sd}$) do
9:     Perform the Modified L-shape Algorithm to check whether request $r$ can be allocated on the primary path $p_1$ with a
       minimum cost $cost(p_1)$.
10:    if (success) then
11:        Temporally allocates resources of the primary path $p_1$.
12:        Perform Dynamic Backup Path Finding Algorithm (DBAFA) to find backup path $p_b$ with a minimum cost
           $cost(p_b)$.
13:        if (($p_b$ can be found) $\wedge$ ($cost(p_1) + cost(p_b) < Cost_{opt}$)) then
14:            $PP^{opt} = \{p_1, p_b\}$, $Cost_{opt} = cost(p_1) + cost(p_b)$
15:        end if
16:    end if
17:    Release allocated resources for the path $p_1$.
18: end for
19: if ($Cost_{opt} \neq \infty$) then
20:     Allocate resources for paths in $PP^{opt}$ and return $PP^{opt}$, $Cost_{opt}$.
21: else
22:     return block.
23: end if
```

4.5 Fragmentation-Aware Routing

In this subsection, we present a fragmentation-aware routing method that addresses the issue of fragmentation in the survival routing problem. To effectively handle fragmentation, a path-specific fragmentation metric is needed, and the metric should be computed with minimal computation time. Thus, two metrics, namely *cuts* and *misalignment*, proposed in [22], were used in the (primary and backup) lightpath provisioning process.

Algorithm 4: Dynamic Backup Path Finding Algorithm (DBAFA)

```
1: Input: $G(V,E,D)$, the request $r=(s,d,B)$, primary path $p_1$;
2: Output: backup path $p_b$ with cost $cost(p_b)$;
3: Construct $G^1(V,E,D)$ by removing links of path $p_1$ from $G(V,E,D)$.
4: Collect and update link status of $G^1(V,E,D)$.
5: Construct a new graph $G'(V,E,d')$ with weight $d'(e), \forall e \in E$ using (1) from $G^1(V,E,D)$.
6: Find the set $P_{sd}$ of paths from $s$ to $d$ on network $G'(V,E,d')$ by performing the k-shortest path algorithm.
7: Sort the paths in $P_{sd}$ according to the weighted distance $\sum_e d'(e)$ of path $p \in P_{sd}$.
8: $Cost_{opt}=\infty$, $BP^{opt}=\emptyset$
9: for (all possible paths $p_b$ in $P_{sd}$) do
10:     Perform the Modified L-shape Algorithm to check whether request $r$ can be allocated on the backup path $p_b$ and with
        minimum cost $cost(p_b)$.
11:     if (($p_b$ can be found) $\wedge$ ($cost(p_b) < Cost_{opt}$)) then
12:         $BP^{opt}=\{p_b\}, Cost_{opt}=cost(p_b)$
13:     end if
14: end for
15: if ($Cost_{opt} \neq \infty$) then
16:     return $BP^{opt}, Cost_{opt}$.
17: else
18:     return $\emptyset, \infty$.
19: end if
```

Clearly, for a given connection request, the *cut* value (a nonnegative integer) is a straightforward metric for evaluating the newly introduced spectrum fragmentation along a candidate path [22]. The depicted spectrum assignment not only fills up the fragmented slot on the candidate path but also resolves (or causes) the misalignment problem between the candidate link and its neighboring links. In other words, it may result in fragmentation of the existing continuous big spectrum block on the links. the solutions that minimize the cut value may conflict with the solutions that minimize the increase in misalignment [22]. Therefore, fragmentation-aware routing algorithms should consider both metrics together.

Algorithm 5: Semi-Dynamic Survivable Path Routing Algorithm (SDSPRA)

```
1: Input : $G(V,E,D)$, request $r=(s,d,B)$;
2: Output : primary and backup paths of the request;
3: Construct the set $P_{sd}$ of paths of request $r$ by performing the K-shortest path algorithm on network $G(V,E,D)$.
4: Paths in the $P_{sd}$ are sorted increasingly according to the $N(B,p) \times hop(p)$ of path $p$.
5: while ($P_{sd} \neq \emptyset$) do
6:     {
7:     Select and remove a path $p_1$ from $P_{sd}$.
8:     Perform the Modified L-shape Algorithm to check whether request $r$ can be allocated on the primary path $p_1$.
9:     if ($p_1$ can be found) then
10:         Temporally allocates resources of the primary path $p_1$.
11:         Perform Dynamic Backup Path Finding Algorithm (DBAFA) to find backup path $p_b$.
12:         if ($p_b$ can be found) ) then
13:             return $PP^{opt}=\{p_1,p_b\}$.
14:         end if
15:     end if
16:     Release allocated resources for the path $p_1$.
17:     }
18: end while
19: return $\emptyset$.
```

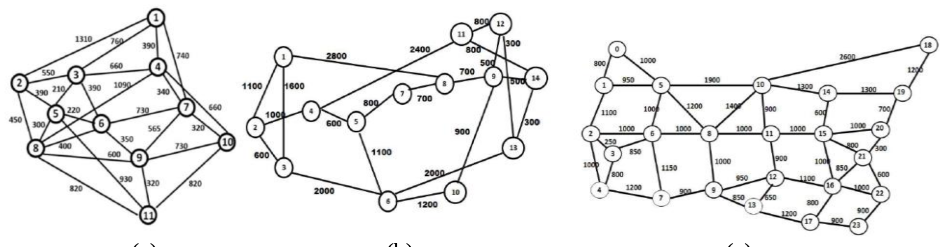
Let's assume that the number of required FSs of the connection request on the candidate path P_i is denoted as FS_M. The cut of the candidate path P_i within the selected FS range $[j, j+FS_M+1]$ is represented as $FC_{ij} = \sum_{e_l \in P_i} cut_{ij}^l$. Here, $cut_{ij}^l \in \{0,1\}$ indicates the cut status of the link e_l when allocating the FSs from j to $j+FS_M+1$ on the candidate path P_i. If the FSs from j to $j+FS_M+1$ on the link e_l of path P_i are allocated and result in a cut, then cut_{ij}^l is set to 1; otherwise, cut_{ij}^l is set to 0. The value of FC_{ij} falls within the interval [0, $hop(P_i)$], where $hop(P_i)$ represents the number of links in the path P_i. The misalignment of the candidate path P_i within the selected FS range $[j, j+FS_M+1]$ is represented as FM_{ij} and defined by Eq. (2),

$$FM_{ij}=\sum\nolimits_{\forall e_l\in P_i}\sum\nolimits_{\forall e'_l\in N_{e_l}}\sum\nolimits_{j'=j}^{j+FS_M+1}A(e_l,e'_l,j'). \tag{2}$$

In the equation above, N_{e_l} refers to the set of links that are adjacent to the link $e_l \in P_i$ but not part of P_i. The misalignment for the link e_l in P_i and the neighboring link e'_l in N_{e_l} for allocation on the jth is denoted as FS $A(e_l, e'_l, j')$, where $A(e_l, e'_l, j') \in \{0, 1\}$. If there is a misalignment, then $A(e_l, e'_l, j') = 1$; otherwise, $A(e_l, e'_l, j') = 0$. The value of FM_{ij} falls within the interval $[0, \sum_{\forall e_l \in P_i}|N_{e_l}| \times hop(P_i) \times FS_M]$, where $|N_{e_l}|$ represents the number of links in N_{e_l}.

Then, the cut and misalignment are utilized to select the path and the FS range j~$(j+FS_M-1)$ for the path. The primary consideration is as follows: if the number of required FSs for candidate paths is the same, the path and spectrum assignment(s) with the minimum cut is selected. If the number of required FSs and the cut of candidate paths are also the same, the path and spectrum assignment(s) with the minimum misalignment are selected. To achieve this objective, the cut and misalignment are normalized to the range [0, 1]. The cost function F_{cmt} for the selected path P_i and FS range $[j, j+FS_M-1]$ is defined as Eq. (3),

$$F_{cmt}=cost(P_i)+\frac{FC_{ij}}{hop(P_i)}+\frac{FM_{ij}}{FS_M\times(n-1)^2\times hop(P_i)}. \tag{3}$$


(a) (b) (c)
Fig. 5. Simulation networks; (a) COST239; (b) NSF14; (c) NSF24.

5. SIMULATION RESULTS

The proposed algorithms were implemented using the C++ programming language. Simulations were conducted on an ad hoc simulator running on a personal computer with the following specifications: Intel Core i7-11700 processor with 8 cores, clocked at 2.50 GHz, 16.0 Gigabytes RAM, and running the Windows 11 Pro 64-bit operating system. Three topologies, namely COST239, NSF14, and NSF24 (shown in Fig. 5), were used for the simulations. All links in these networks are bidirectional, and each fiber is assumed to have 300 available FSs. In these simulations, static traffic was used, where the connection request bandwidth was randomly generated. The average bandwidth of each connection was calculated as $150 \times load$, where *load* represents the load factor. Different values of the load factor were used in the simulations to investigate various traffic scenarios. The traffic was uniformly generated randomly between 10 Gb/s and 290 Gb/s for each pair of nodes in the network. Six types of modulation formats were considered. The parameter K, which determines the number of candidate paths considered for routing the request (both for the primary and backup lightpaths), was set to the default value of 3. This means that three candidate paths were examined, and if there were no available free slots to allocate on the selected path, the request was considered blocked. Each node in the network was equipped with 60 slicers and 60 stitchers, which are used for the slicing and stitching of optical signals, respectively.

5.1 Simulated Algorithms

For each proposed algorithm, two slicing schemes were applied: source-slicing (_S) and L-shape-slicing (_L). In addition, the no-slicing (_N) routing algorithm was also implemented for comparison. The BBR, PRR, and CPU time of each slicing scheme and each proposed algorithm were compared. Two heuristic unicast routing algorithms proposed in [26] were also implemented and used for comparison. They are **improved kSP (IKSP)** and **modified shortest path 2 (MSP2)** algorithms.

The IKSP algorithm is a modified version of the k-shortest paths algorithm and can be classified as fixed alternate routing algorithm. It determines a set of the shortest paths whose lengths are determined by the number of links (or hops), and the paths of equal hops are sorted in ascending order based on the actual length. For each candidate path, the aggregated spectrum is determined, and the modulation level and required number of slots are determined based on the actual length. The aggregated spectrum is searched using the first-fit procedure. The computational complexity of the IKSP algorithm is $O(Knm + Kn^2 \log n + nF)$.

Table 1. Time complexity of all proposed algorithms.

Alg.	Slic.	Time complexity	Alg.	Slic.	Time complexity
FA	N	$O(hop(p)\times F) = O(nF)$			
	S	$O(hop(p)\times F+F\log F)$			
	L	$O(hop(p)\times F^2) = O(nF^2)$			
MFDPRA		$O(nm^2+n^2FA)$	SDPRA		$O(KmF+K^2nm+K^2n^2\log n+K^2FA)$
	N	$O(nm^2+n^3F)$		N	$O(KmF+K^2nm+K^2n^2\log n+K^2nF)$
	S	$O(nm^2+n^3F+F\log F)$		S	$O(KmF+K^2nm+K^2n^2\log n+K^2nF+K^2F\log F)$
	L	$O(nm^2+n^3F^2)$		L	$O(KmF+K^2nm+K^2n^2\log n+K^2nF^2)$
ADPRA		$O(Knm+Kn^2\log n+K^2FA)$	IKSP		$O(K^2nm+K^2n^2\log n+K^2FA)$
	N	$O(Knm+Kn^2\log n+K^2nF)$		N	$O(K^2nm+K^2n^2\log n+K^2nF)$
	S	$O(Knm+Kn^2\log n+K^2nF+K^2F\log F)$		S	$O(K^2nm+K^2n^2\log n+K^2nF+K^2nF\log F)$
	L	$O(Knm+Kn^2\log n+K^2nF^2)$		L	$O(K^2nm+K^2n^2\log n+K^2nF^2)$
DPRA		$O(KmF+K^2nm+K^2n^2\log n+K^2nF^2)$	MSP2		$O(ML_{max}mF+ML_{max}n^2FA)$
	N	$O(KmF+K^2nm+K^2n^2\log n+K^2nF)$		N	$O(ML_{max}\times n^3F)$
	S	$O(KmF+K^2nm+K^2n^2\log n+K^2nF\log F)$		S	$O(ML_{max}\times n^3F+ML_{max}\times n^2F\log F)$
	L	$O(KmF+K^2nm+K^2n^2\log n+K^2nF^2)$		L	$O(ML_{max}\times n^3F^2)$

The MSP2 algorithm is a modified version of Dijkstra's algorithm and can be classified as adaptive routing. The MSP2 algorithm uses the utilization of network links, where loaded links are assigned a slightly higher weight and omitted on the calculated paths. The link weights in the network are multiplied by a coefficient that takes into account the utilization of the network links. This algorithm is iterative, with each iteration finding the minimum weight path for a given modulation level $m \in [1, ML_{max}]$. If the required FSs can be allocated along the selected path, it is returned. Otherwise, the modulation level decreases and the process is repeated. The computational complexity of the MSP2 algorithm is $O(ML_{max} \times Fn^2)$.

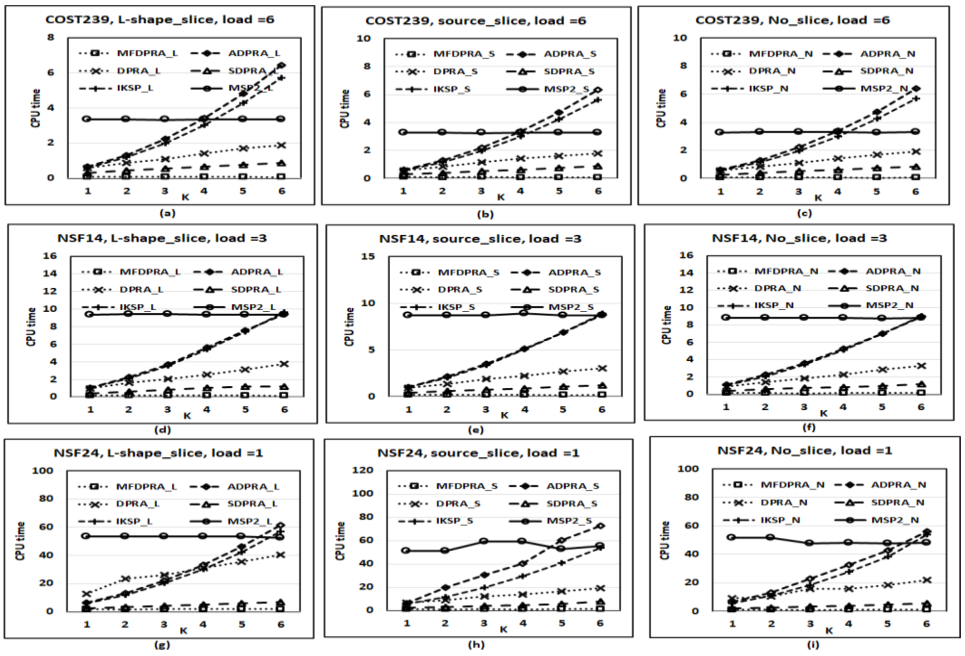


Fig. 6. CPU time results for different values of K and different algorithms and different slicing schemes on networks; (a)-(c) COST239; (d)-(f) NSF14; (g)-(i) NSF24.

These algorithms were applied to find a pair of link-disjoint paths for each request. The primary path p is found by performing the algorithm, and then all links passed by the primary path are removed from the graph. Finally, the backup path is found by performing the same algorithm. In the MSP2 algorithm, a pair of link-disjoint paths is found, while in the IKSP algorithm, K^2 pairs of link-disjoint paths are found.

5.2 Computation Time

The worst-case time complexity of the six algorithms and three FA slicing schemes is summarized and shown in Table 1. Additionally, the CPU time in seconds for the three networks, different slicing schemes, and different values of K (in 1, 2, 3, 4, 5, 6) is shown in Fig. 6. Fig. 6 illustrates that the CPU time in seconds for the ADPRA, DPRA, SDPRA, and IKSP algorithms increases as the value of K increases. The CPU time for the MFDPRA and MSP2 algorithms remains unchanged because a fixed number of candidate pairs of link-disjoint paths are examined. The results in Figs. 6 (a)-(c) indicate that the ADPRA is the most time-consuming routing algorithm when the value of K is greater than 4 for the

COST239 network. On the other hand, the MFDPRA is the quickest routing algorithm. The DPRA algorithm is faster than the IKSP and ADPRA algorithms. From the analysis presented in Table 1 and Fig. 6, it can be observed that for the same algorithm but with different slicing schemes, L-shape slicing is the most time-consuming slicing scheme, while no-slicing is the quickest slicing scheme.

5.3 BRR Performance

First, the simulation results of BBR for different algorithms and different slicing schemes are shown in Fig. 7 for the COST239 network. Figs. 7 (a)-(c) depict the BBR for different routing algorithms and different slicing schemes for different values of *K*. It can be observed that the value of *K* does not affect the BBR of the MFDPRA and MSP2 algorithms since the number of candidate disjoint paths is fixed in the given network. However, for the other algorithms with different slicing schemes, as the value of *K* increases, the BBR value decreases. This suggests that the value of *K* may impact the dynamic search for primary and backup paths, and increasing the value of *K* can lead to a reduction in the BBR.

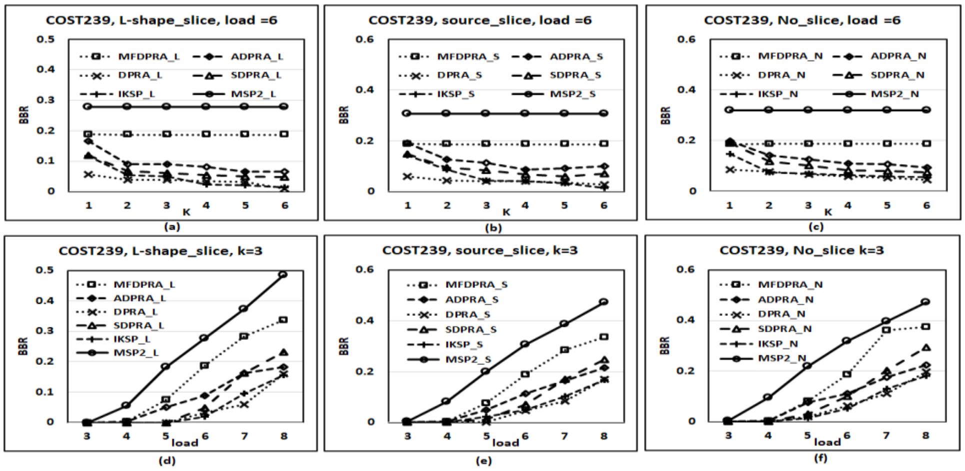


Fig. 7. BBR simulation on COST239 network: (a)&(d) L-shape slicing; (b)&(e) source-slicing; (c)&(f) no-slicing DRPA.

Figs. 7 (d)-(f) show the simulation results of BBR for different routing algorithms and different slicing schemes for different *load* values. It can be observed that the DPRA achieves the lowest BBR for most cases, while the MSP2 algorithm yields the highest BBR for different slicing schemes on the COST239 network. The IKSP algorithm performs as the second-best algorithm in terms of BBR on the COST239 network. Additionally, for most slicing schemes, the MFDPRA exhibits the second-worst BBR performance. Furthermore, the simulation results shown in Fig. 8 demonstrate that, in most cases, the BBR of the algorithm with the L-shape-slicing scheme is superior to that of the other slicing schemes. The fragmentation-aware FS allocation scheme can achieve a better BBR compared to the no-slicing scheme, but it may be worse than that of the source and L-shape slicing schemes.

First, for the NSF14 network, the simulation results of BBR for different algorithms and different slicing schemes are shown in Fig. 9. First, the simulation results of BBR for different algorithms and different slicing schemes are shown in Fig. 9 for the NSF14 network. Figs. 9 (a)-(c) illustrate the BBR for different routing algorithms and different slicing schemes for different values of K. Similar to the COST239 network, the value of K does not affect the BBR of the MFDPRA and MSP2 algorithms. However, for the other algorithms with different slicing schemes, as the value of K increases, the BBR value decreases.

This observation suggests that the value of K can impact the dynamic search for primary and backup paths, and increasing its value can lead to a reduction in the BBR. In these simulations, the BBR performance of the DPRA algorithm is approximately 5% lower than that of the IKSP algorithm.

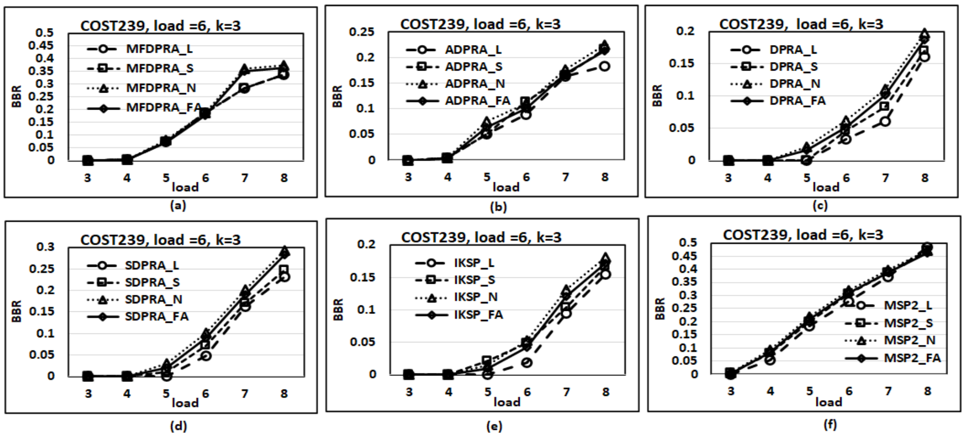


Fig. 8. BBR simulation on COST239 network for different load and different algorithms: (a) MFD RPA; (b) ADPRA; (c) DRPA; (d) SDPRA; (e) IKSP; (f) MSP2.

The simulation results of BBR for different routing algorithms and different slicing schemes for different *load* values (in 1, 2, 3, 4, 5, 6) are shown in Figs. 9 (d)-(f) for the NSF14 network. The DPRA algorithm consistently achieves the lowest BBR for all load scenarios, indicating its effectiveness in minimizing blocking. On the other hand, the MFDPRA algorithm exhibits the highest BBR among the different slicing schemes on the NSF14 network. The IKSP algorithm performs well, achieving the second-best BBR performance, followed by the ADPRA algorithm. The simulation results shown in Fig. 10 demonstrate that in most cases, the BBR of the algorithm with the L-shape-slicing scheme is better than that of the other slicing schemes. This suggests that the L-shape slicing scheme can effectively improve the BBR performance in the NSF14 network.

For the NSF24 network, the simulation results of BBR for different algorithms and different slicing schemes are shown in Fig. 11. The simulation results of BBR for different routing algorithms and different slicing schemes for different values of *K* are shown in Figs. 11 (a)-(c). Similar to the previous networks, the value of *K* does not significantly affect the BBR of the MFDPRA and MSP2 algorithms, as they have a fixed number of candidate disjoint paths. For the other algorithms on different slicing schemes, as the value of *K* increases, the BBR value decreases, indicating that a larger value of *K* allows for better routing and lower blocking rates.

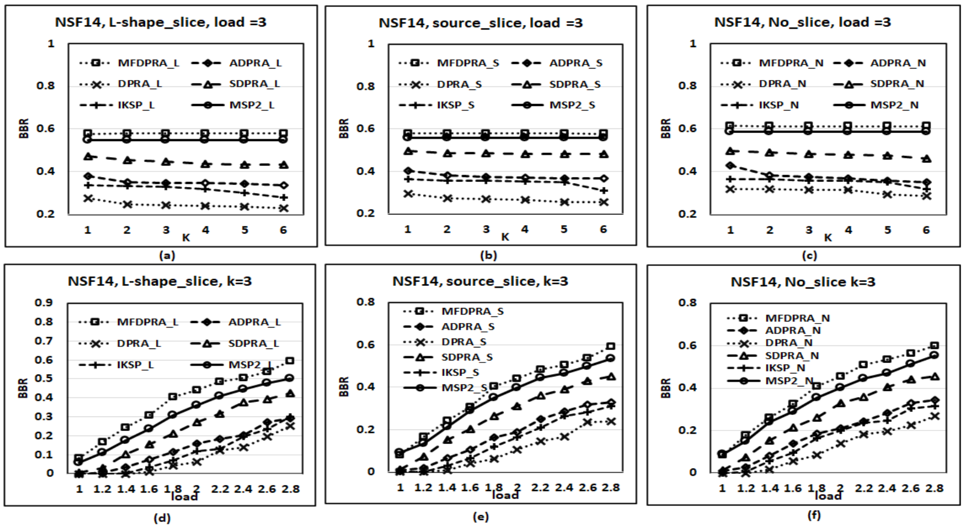


Fig. 9. BBR simulation on NSF14 network: (a)&(d) L-shape slicing; (b)&(e) source-slicing; (c)&(f) no-slicing DRPA.

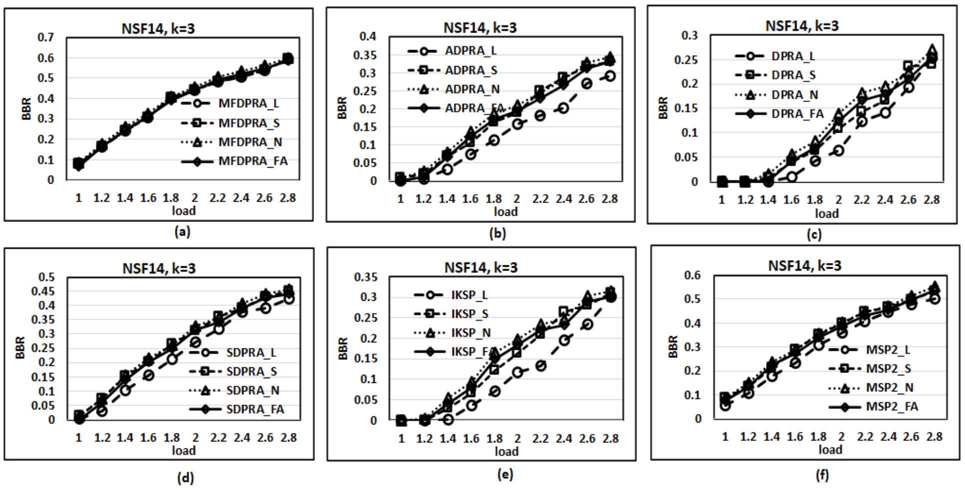


Fig. 10. BBR simulation on NSF14 network for different load and different algorithms: (a) MFDRPA; (b) ADPRA; (c) DRPA; (d) SDPRA; (e) IKSP; (f) MSP2.

In these simulations, the BBR performance of the DPRA algorithm is lower than that of the IKSP algorithm by approximately 4%. Furthermore, for the no-slicing scheme, the SDPRA algorithm achieves a lower BBR compared to the IKSP algorithm on the NSF24 network. These results demonstrate the effectiveness of the SDPRA algorithm in minimizing blocking when no slicing is applied.

The simulation results of BBR for different routing algorithms and different slicing schemes for different *load* values (ranging from 0.3 to 3.0) are shown in Figs. 11 (d)-(f). Among the algorithms, the DPRA consistently achieves the lowest BBR for all load levels, indicating its effectiveness in minimizing blocking in the NSF24 network. On the other hand, the MSP2 algorithm tends to have the highest BBR among the algorithms for different slicing schemes, indicating its relatively higher blocking rate. The IKSP algorithm performs well, achieving the second-best BBR performance for different load levels when using both L-shape and source slicing schemes. However, for the no-slicing scheme, the SDPRA algorithm outperforms the IKSP algorithm, achieving a lower BBR.

The simulation results in Fig. 12 demonstrate that in most cases, the L-shape slicing scheme leads to a lower BBR compared to the other slicing schemes. In particular, for the ADPRA, DPRA, and IKSP algorithms, using the L-shape slicing scheme can reduce the BBR by approximately 10% compared to the other slicing schemes.

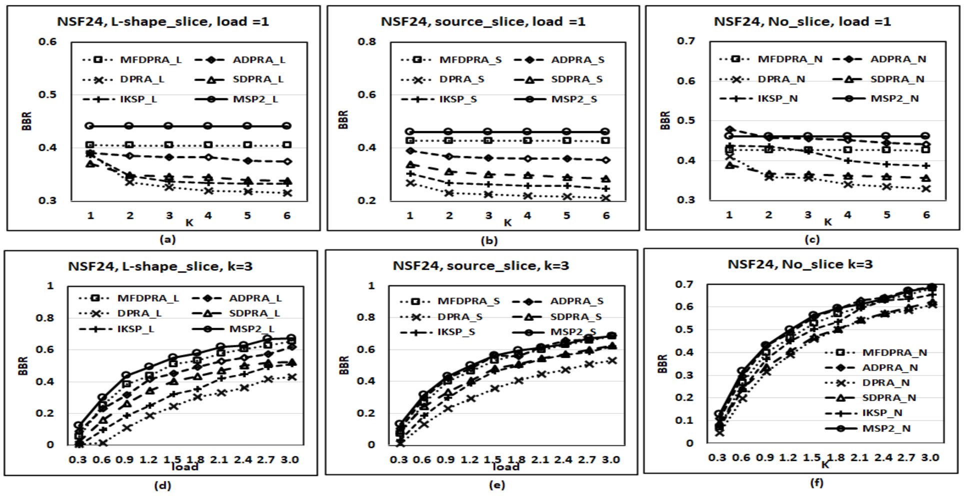


Fig. 11. BBR simulation on NSF24 network: (a)&(d) L-shape slicing; (b)&(e) source-slicing; (c)&(f) no-slicing DRPA.

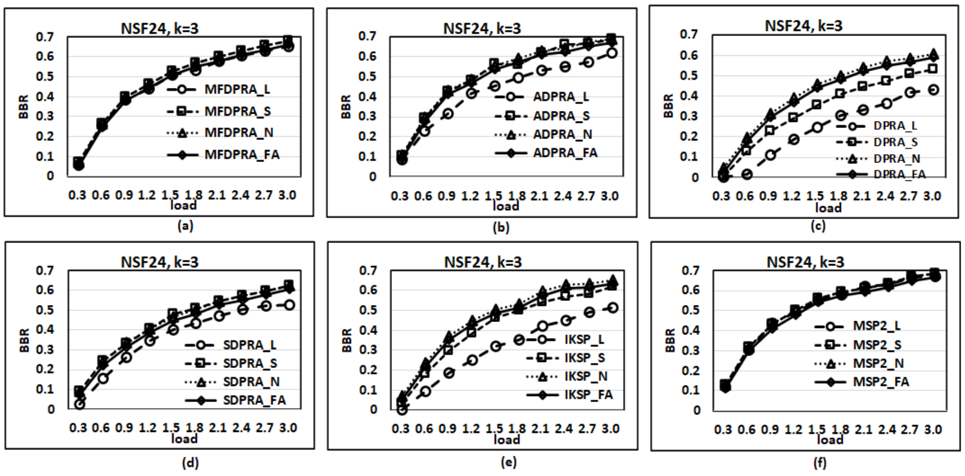


Fig. 12. BBR simulation on NSF24 network for different load and different algorithms: (a) MFDRPA; (b) ADPRA; (c) DRPA; (d) SDPRA; (e) IKSP; (f) MSP2.

5.4 PRR Performance

Fig. 13 presents the PRR performance of all algorithms and different slicing schemes for various load levels on all networks. However, it can be observed that for the DPP scheme, the PRR values of all algorithms (except for the MSP2 algorithm in the light load case) consistently exceed 1.0, and in some cases, they even reach 1.9. This implies that the backup resources are scarce and difficult to find for short-distance paths, leading to longer paths being used as backup paths and consequently increasing the PRR values. For the MSP2 algorithm, since it always finds the minimum hop paths to route the connection request, if the FS allocation is available, the PRR value can remain low. For different networks and slicing schemes, as the load increases, the PRR of the MFDRPA and MSP2 algorithms also increases. This can be attributed to the fact that the backup resources become more challenging to allocate for short-distance paths, resulting in longer paths being selected as backup paths. Since the MFDRPA and MSP2 algorithms have a lower number of candidate paths compared to the other algorithms, they are more likely to choose longer paths, leading to higher PRR values.

Fig. 13 indicates that the DPRA algorithm consistently achieves the best PRR for most cases, regardless of the slicing scheme used. Furthermore, there is little variation in the PRR values across different slicing schemes. This suggests that the backup resource allocation process is challenging for both short-distance and long-distance paths, resulting in similar PRR values across slicing schemes.

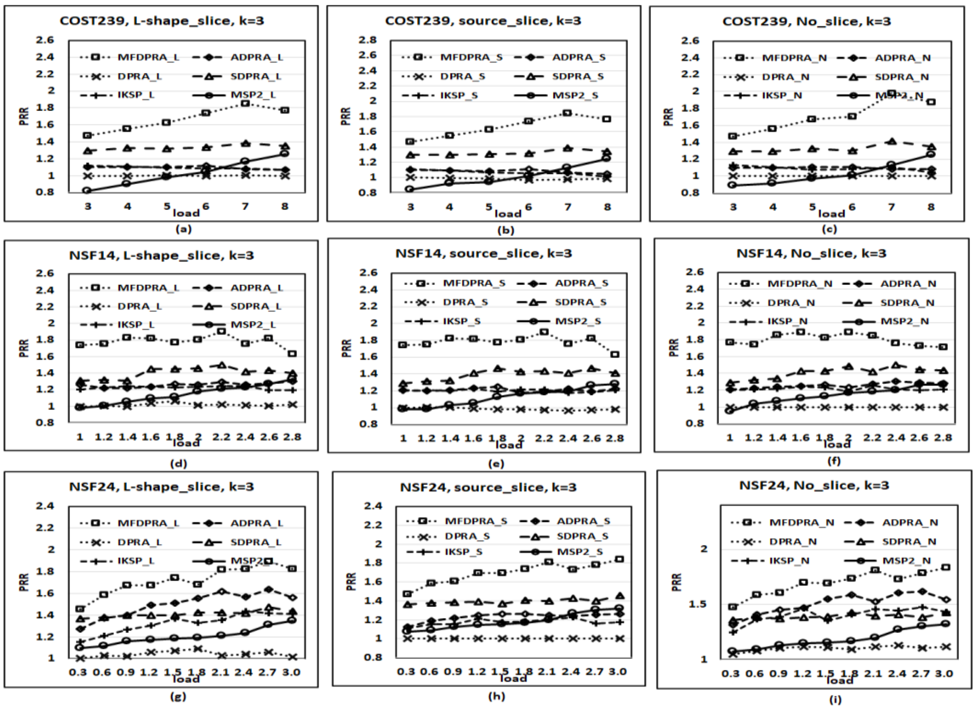


Fig. 13. PRR simulations on networks for different slicing schemes and different loads: (a)-(c) COST239; (d)-(f) NSF14; (g)-(i) NSF24.

5.5 Effect of Number of Slicers

Fig. 14 presents the BBR results of the source-slicing and L-shape slicing schemes for all proposed algorithms on the three networks (COST239, NSF14, and NSF24) with $K = 3$. The load factor *load* is set to 6, 2, and 1 for the COST239, NSF14, and NSF24 networks, respectively. Additionally, the number of slicers per node is varied in the range of {10, 30, 50, 70, 90, 110, 130, 150}.

The simulation results depicted in Fig. 14 reveal that as the number of slicers per node increases, the BBR decreases. This suggests that having a higher number of slicers per node enables more efficient allocation of resources, leading to lower blocking rates. Furthermore, it can be observed that the L-shape slicing scheme consistently outperforms the source-slicing scheme in terms of BBR. This implies that the L-shape slicing scheme offers better resource utilization and allocation efficiency, resulting in lower blocking rates compared to the source-slicing scheme.

One possible explanation for these findings is that under heavy loads, slicers may become less efficient and struggle to allocate resources effectively. As a result, the source-slicing scheme may experience higher blocking rates compared to the L-shape slicing scheme, which demonstrates the advantages of the latter in such scenarios.

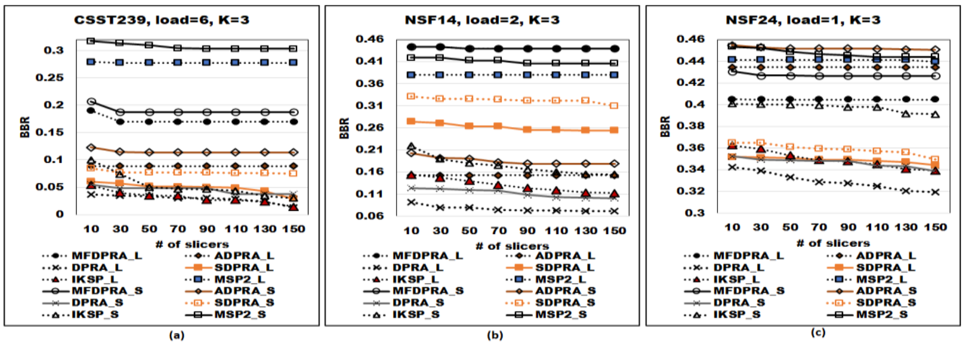


Fig. 14. BBR for different number of slicers (a) COST239; (b) NSF14; (c) NSF24.

5. CONCLUSIONS

In this paper, we have investigated the problem of survivable routing in the context of the RMLSA model on EONs with slicers. The objective is to develop routing methods that can effectively handle connection requests while satisfying their specific requirements. To address this problem, we have considered the DPP protecting scheme and proposed four heuristic algorithms. These algorithms aim to find suitable primary and backup paths for each request, considering factors such as available resources, modulation levels, and link spectrum usage.

Our findings indicate that the L-shape slicing scheme outperforms the source slicing and no-slicing schemes in terms of blocking rate (BBR). By using the L-shape slicing scheme or the source-slicing scheme, we can achieve lower BBR compared to the no-slicing scheme. Additionally, we have observed that increasing the number of candidate paths considered during the routing process can help reduce the BBR. This suggests that exploring a wider range of path options improves the chances of finding suitable primary and backup paths. In conclusion, our study has provided insights into the survivable routing problem in EONs with slicers. The proposed algorithms, particularly those utilizing L-shape slicing or source-slicing schemes, offer effective solutions for minimizing bandwidth blocking rates and improving network performance.

REFERENCES

1. N. Mahala, Ujjwal, and J. Thangaraj, "Weight distributed spectrum allocation in flexible-grid optical networks," *Optik*, Vol. 228, 2021, pp. 1-9.
2. P. Lu, L. Zhang, X. Liu, J. Yao, and Z. Zhu, "Highly-efficient data migration and backup for big data applications in elastic optical inter-data-center networks," *IEEE Network*, Vol. 29, 2015, pp. 36-42.
3. A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Computer Communication Review*, Vol. 39, 2009, pp. 68-73.
4. G. Zhang, M.D. Leenheer, A. Mcrea, and B. Mukherjee, "A survey on OFDM-based elastic core optical networking," *IEEE Communications Surveys & Tutorials*, Vol. 15, 2012, pp. 65-87.
5. F. S. Abkenar and A. G. Rahbar, "Study and analysis of routing and spectrum allocation (RSA) and routing, modulation and spectrum allocation (RMSA) algorithms in elastic optical networks (EONs)," *Optical Switching and Networking*, Vol. 23, 2017, pp. 5-39.
6. L. Costa, G. Ramos, and A. Drummond, "Leveraging adaptive modulation with multihop routing in elastic optical networks," *Computer Networks*, Vol. 105, 2016, pp. 124-137.
7. B. C. Chatterjee, S. Ba, and E. Oki, "Fragmentation problems and management approaches in elastic optical networks: a survey," *IEEE Communications Surveys & Tutorials*, Vol. 20, 2018, pp. 183-210.
8. Y. Cao, *et al.*, "Experimental demonstration of tunable optical channel slicing and stitching to enable dynamic bandwidth allocation," in *Proceedings of Optical Fiber Communication Conference*, paper Th1F.1.
9. Y. Cao, *et al.*, "Reconfigurable channel slicing and stitching for an optical signal to enable fragmented bandwidth allocation using nonlinear wave mixing and an optical frequency comb," *Journal of Lightwave Technology*, Vol. 36, 2018, pp. 440-446.
10. N. Kitsuwan, P. Pavarangkoon, and A. Nag, "Elastic optical network with spectrum slicing for fragmented bandwidth allocation," *Optical Switching and Networking*, Vol. 38, 2020, p. 100583.
11. K. Akaki and N. Kitsuwan, "First-large fit spectrum allocation for elastic optical network with spectrum slicing," in *Proceedings of the 16th International Conference on IP + Optical Network*, 2020, P-1.
12. N. Kitsuwan and R. Matsuura, "Performance of elastic optical network with spectrum slicing for fragmented bandwidth allocation," in *Proceedings of International Conference on Computing, Networking and Communications*, 2019, pp. 607-611.
13. K. Akaki, P. Pavarangkoon, and N. Kitsuwan, "Large L-shape fit spectrum allocation for elastic optical network with spectrum slicing," in *Proceedings of International*

Conference on Information Networking, 2021, pp. 537-540.

14. N. Kitsuwan, K. Akaki, P. Pavarangkoon, and A. Nag, "Spectrum allocation scheme considering spectrum slicing in elastic optical networks," *IEEE/OSA Journal of Optical Communications and Networking*, Vol. 13, 2021, pp. 169-181.
15. K. Akaki, P. Pavarangkoon, and N. Kitsuwan, "Elastic optical network for fragmented bandwidth allocation with limited slicers," in *Proceedings of the 13th International Conference on Information Technology and Electrical Engineering*, 2021, pp. 1-4.
16. N. Kitsuwan and P. Pavarangkoon, "Reducing bandwidth blocking rate in elastic optical networks through scale-based slicer placement strategy," *IEEE Access*, Vol. 11, 2023, pp. 45313-45322.
17. H. Liu, J. Ren, Y. Chen, J. Hu, C. Tang, and M. Tang, "Spectrum slicing-based fragmentation aware routing and spectrum allocation in elastic optical networks," *Optical Switching and Networking*, Vol. 45, 2022, p. 100673.
18. N. Kitsuwan and K. Akaki, "Performance of elastic optical network with limited slicers," *ICT Express*, Vol. 9, 2022, pp. 362-365.
19. P. Zou, S. Petale, J. Zhao, and S. Subramaniam, "Performance modeling of partitioning and slicing spectrum assignment schemes in EONs," in *Proceedings of International Conference on Communications*, 2022, pp. 3028-3033.
20. G. Shen, H. Guo, and S. K. Bose, "Survivable elastic optical networks: survey and perspective," *Photonic Network Communications*, Vol. 31, 2016, pp. 71-87.
21. M. Klinkowski and K. Walkowiak, "Offline RSA algorithms for elastic optical networks with dedicated path protection consideration," in *Proceedings of the 4th International Congress on Ultra-Modern Telecommunications and Control Systems and Workshops*, 2012, pp. 670-676.
22. L. Liu, *et al.*, "Software-defined fragmentation-aware elastic optical networks enabled by OpenFlow," in *Proceedings of the 39th European Conference and Exhibition on Optical Communication*, 2013, pp. 1-3.
23. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Section 26.2: The Ford-Fulkerson method," *Introduction to Algorithms*, 2nd ed., MIT Press and McGraw-Hill, 2001, pp. 651-664.
24. J. Y. Yen, "Finding the *k* shortest loopless paths in a network," *Management Science*, Vol. 17, 1971, pp. 712-716.
25. Z. Zhu, W. Lu, L. Zhang, and N. Ansari, "Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing," *Journal of Lightwave Technology*, Vol. 31, 2013, pp. 15-22.
26. I. Olszewski, "Improved dynamic routing algorithms in elastic optical networks," *Photonic Network Communications*, Vol. 34, 2017, pp. 323-333.

Der-Rong Din (丁德榮) received Ph.D. degree in Computer and Information Science from National Chiao-Tung University in 2001, respectively. Now, he is a Professor in the School of Department of Computer Science and Information Engineering at National Changhua University of Education. His research interests include routing, multicast routing, survivability, network design, network planning in optical networks.