Optical Switching and Networking 49 (2023) 100742

Contents lists available at ScienceDirect

Optical Switching and Networking

journal homepage: www.elsevier.com/locate/osn

A symmetry-free spectrum allocation heuristic for elastic optical networks ☆

George N. Rouskas

North Carolina State University, United States of America

ARTICLE INFO

Keywords:
Spectrum allocation
Optical network design
First-fit algorithm
Elastic optical networks
Symmetry-free allocation

ABSTRACT

We revisit spectrum allocation (SA), a fundamental problem in optical network design, and we explain that it can be modeled as a permutation problem. This new model eliminates spectrum symmetry, a property that presents a significant challenge to conventional spectrum allocation solutions. Accordingly, we develop parameterized first-fit (PFF), a new symmetry-free heuristic for the SA problem that has several desirable features: it explores a pre-defined subset of the solution space whose size is tailored to the available computational budget; it constructs this subset by sampling from diverse areas of the solution space rather than from the neighborhood of an initial solution; it finds solutions by applying the well-known first-fit (FF) algorithm and thus it can be deployed readily; its execution can be easily parallelized; and it is effective and efficient in finding good quality solutions.

1. Introduction

The design and planning of optical networks encompasses the allocation of optical spectrum resources to traffic demands as an integral part of the optimization process [1]. Spectrum allocation (SA) [2], a generalization of wavelength allocation (WA) [2,3], is tightly coupled to other aspects of network design, including the routing process [4–6], traffic grooming [7], virtual topology embedding [8,9], and network survivability [10]. Therefore, since the early days of optical networking, researchers and industry practitioners have focused on developing effective spectrum allocation strategies. These efforts, however, have been complicated by two features inherent to the SA and WA problems: spectrum continuity and spectrum symmetry.

Due to the spectrum continuity property of optical elements, a connection that optically bypasses a node must exit on the same optical frequency it entered. Hence, the spectrum resources that are in use on one link may affect the resources that may be allocated on other links, creating resource contention among the links of the network. As a result, the SA problem is computationally intractable in general topologies [11], even when it is not coupled to other objectives (e.g., routing).

Symmetry refers to the fact that spectrum slots are *interchangeable* [12]. Hence, for each possible solution, a large number of equivalent solutions may be derived simply by using a different permutation of spectrum slots [13]. Symmetry is particularly challenging for conventional ILP formulations of the SA problem, regardless of whether these were developed specifically for SA or as part of formulations that tackle the more general routing and spectrum allocation (RSA) problem [5,14–16]. Since an ILP solver will have to evaluate an exponential number of distinct but equivalent optimal solutions, its running time can be unnecessarily long [13]. ILP formulations based on maximal independent sets (MIS), such as the one we developed in [17] for the RWA problem in rings, do not suffer from symmetry. However, MIS-based formulations are impractical for networks of general topology as the number of variables increases exponentially with the network size.

Given these two challenges, the SA problem is typically solved using heuristic algorithms that attempt to minimize spectrum contention. These include the first-fit, best-fit, most-used, and least-loaded heuristics [18], each representing a different tradeoff between algorithmic complexity and amount of network state information required. In particular, first-fit (FF) is a simple algorithm that operates without any global knowledge and performs well across various network topologies and traffic demands [1,2,19]. Consequently, the FF algorithm is commonly employed for spectrum/wavelength allocation.

In recent work [20] we have proven an optimality property of the FF algorithm that provides new insight into the SA problem and allows us to model it as a permutation problem. We also leveraged this property to develop recursive first-fit (RFF), an optimal branch-and-bound algorithm for spectrum allocation. To the best of our knowledge, RFF is the first spectrum symmetry-free SA algorithm for networks of general topology, and our experiments indicate that it can find optimal or near-optimal solutions to medium-size networks quickly [20]. Nevertheless, since the SA problem is NP-hard, the RFF algorithm takes exponential

☆ This work was supported by the National Science Foundation, United States under Grant CNS-1907142. An early version of this article appeared in Rouskas et al. (2022) [21].

E-mail address: rouskas@ncsu.edu.

https://doi.org/10.1016/j.osn.2023.100742
Received 22 February 2023; Received in revised form 26 March 2023; Accepted 16 April 2023
Available online 22 April 2023

time in the worst case even though it completely avoids symmetrical solutions.

In this work, an extension of [21], we present parameterized first-fit (PFF), a new heuristic for the SA problem that builds upon the permutation model and represents a significant departure from existing heuristics. The PFF heuristic has several desirable properties: (1) it employs an intuitive parameter to calibrate the size of the solution space subset to explore according to the available computational resources; (2) it constructs this subset by including solutions distributed uniformly across the whole solution space (i.e., the explored subset is not limited to any particular region of the solution space); (3) it applies the well-known FF algorithm, hence it can be readily deployed; and (4) it is amenable to multi-threaded implementation to explore the solution space in parallel.

The rest of the paper is organized as follows. In Section 2 we discuss the SA problem, explain the concept of spectrum symmetry, discuss the optimality property of the FF algorithm, model the SA problem as a permutation problem, and describe the symmetry-free RFF algorithm. In Section 3 we present parameterized FF (PFF), a new heuristic for the SA problem that allows the network designer to explore customizable subsets of permutations that sample from diverse regions of the solution space. We evaluate the PFF algorithm in Section 4, and we conclude the paper in Section 5.

2. A symmetry-free model of spectrum allocation

2.1. The offline SA problem

Consider an optical network with topology graph $G = (V, A)$, where V is the set of nodes and A is the set of directed fiber links in the network. Let $N = |V|$ denote the number of nodes and $L = |A|$ the number of directed links. The traffic offered to the network consists of a set $\mathcal{T} = \{T_i\}$ of K connections. Each connection is represented by a tuple $T_i = (s_i, d_i, p_i, t_i)$, where: s_i is the source and d_i the destination node of the connection; p_i is the path between nodes s_i and d_i that the connection must follow; and t_i is the number of spectrum slots required to carry the traffic of the connection.

In this work we develop a new algorithm for the offline SA problem. Offline SA problem takes as input a network topology graph $G = (V, A)$ and a set $\mathcal{T} = \{T_i\}$ of K connections as defined above. The objective of our algorithm is to allocate spectrum slots to each connection along its physical path so as to minimize the index of the highest spectrum slot used on any link of the network. The allocation of spectrum slots must satisfy three constraints:

- *Contiguity:* each connection T_i is allocated a block of t_i contiguous spectrum slots;
- *Continuity:* each request is allocated the same block of spectrum slots along all links of its path p_i; and
- *Non-overlap:* requests whose paths share a link are allocated non-overlapping blocks of spectrum slots.

This objective attempts to pack the allocated spectrum slots as tightly as possible, and hence it minimizes spectrum fragmentation and allows for growth in demand; consequently, it is one that has been adopted widely in the literature. Also, we assume that the path p_i of each connection T_i is fixed and pre-determined, i.e., any routing decision has been made before the allocation of spectrum. Therefore, any algorithm that solves this SA problem, including the one we present in the next section, may be applied as part of a multi-step, iterative approach to the RSA problem [2].

2.2. Spectrum symmetry

We have shown [11] that the offline SA problem is NP-hard even for chain (i.e., single-path) networks with four or more links. But even beyond computational intractability, a major challenge in tackling this SA problem or any of its variants that have been studied in the literature, relates to *spectrum symmetry*. Specifically, blocks of contiguous spectrum slots of a certain size are interchangeable. Therefore, for any optimal solution to the SA problem, one can derive a large number of equivalent solutions simply by permuting the spectrum blocks.

Fig. 1 illustrates how spectrum symmetry leads to multiple equivalent solutions. Fig. 1(a) shows a solution to the SA problem on a four-link chain network with $K = 9$ connections in the set $\mathcal{T} = \{A, B, \ldots, I\}$. Each connection is represented by a different color that spans all the links in its path. For instance, the bottommost connection (i.e., connection A) spans all four links of the network, indicating that connection A has been allocated the contiguous block consisting of spectrum slots 1 and 2 along each of these links. Note that the solution shown in Fig. 1(a) is optimal: the highest assigned slot on Link 3 is equal to the lower bound, i.e., the number of slots required to carry the traffic requests whose path includes Link 3.

Consider now two blocks of spectrum slots in Fig. 1(a): the three-slot block consisting of spectrum slots 3–5, and the five-slot block consisting of spectrum slots 6–10. Fig. 1(b) shows the equivalent solution that can be obtained by permuting these two blocks of slots. In the new solution, the two connections B and C that were allocated slots in the range 3–5 in Fig. 1(a) are now shifted up and are allocated the corresponding slots in the range 8–10, while the four connections D, E, F and G that were allocated slots in the range 6–10, are now shifted down accordingly. Otherwise, the two solutions in Figs. 1(a) and (b) are identical; they are also equivalent in that they yield the same objective function value. Finally, Fig. 1(c) shows a third optimal solution that is obtained from the one in Fig. 1(b) by permuting the three-slot block consisting of spectrum slots 8–10 with the three-slot block consisting of slots 11–13. Furthermore, note that (1) it is possible to obtain many more solutions equivalent to the three shown in Fig. 1 by permuting different blocks of spectrum slots, and (2) although Fig. 1 shows optimal solutions, spectrum symmetry applies to non-optimal solutions as well.

Based on the above discussion it is clear that, due to spectrum symmetry, conventional ILP formulations [5,14–16] may yield an exponentially large number of equivalent (optimal or suboptimal) solutions. Consequently, ILP solvers are forced to explore a solution space that is essentially the product of the connection permutation space and the spectrum permutation space. As we explain next, however, exploring such a vast solution space is unnecessary.

2.3. The FF optimality property

Consider an instance of the offline SA problem on graph G and connection set $\mathcal{T} = \{T_i, i = 1, \ldots, K\}$. The FF algorithm considers connection requests in a fixed order and allocates to each connection a contiguous block of spectrum slots that starts at the lowest-indexed slot available along all links of the connection's path. Let P be a permutation (i.e., an ordering) of the connection requests T_i. Let $SOL(P)$ denote the solution to the SA problem obtained by the FF algorithm when it considers each connection in the order implied by permutation P. Let OPT denote the objective value of an optimal solution to the SA problem. Clearly, for any permutation P of the connections it must be that $OPT \leq SOL(P)$.

In recent work [20] we have proven that for any optimal solution to the SA problem it is possible to construct, in polynomial time, a permutation $P^{\star}_{FF}$ of the K connections such that applying the FF algorithm to the connections in the order implied by $P^{\star}_{FF}$ yields an equivalent (i.e., optimal) solution. This result implies the following optimality property of the FF algorithm, which helps explain why many studies of the SA and WA problems have confirmed that the FF algorithm yields good solutions across a wide range of problem instances:

Definition 2.1 (*FF Optimality Property*). *For any instance of the offline SA problem there exists a permutation $P^{\star}_{FF}$ of the K connections for which the FF algorithm constructs an optimal solution, i.e., $SOL(P^{\star}_{FF}) = OPT$.*

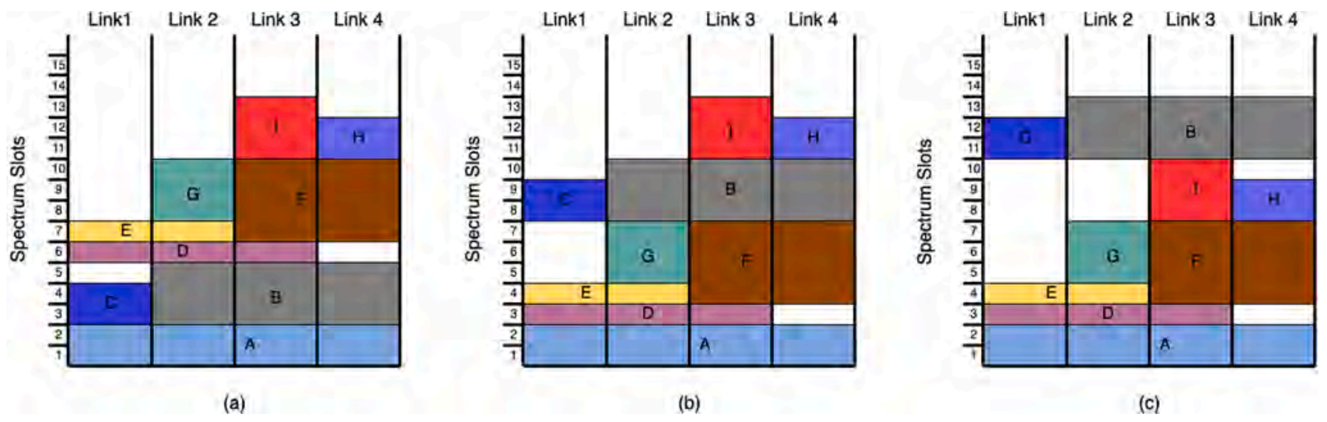


Fig. 1. (a) An optimal solution to an SA problem instance on a 4-link chain with $K = 9$ connections, $A, B, C, D, E, F, G, H, I$. (b) A second optimal solution obtained from the one in (a) by swapping slots 3–5 with slots 6–10. (c) A third optimal solution obtained from the one in (b) by swapping slots 8–10 with slots 11–13.

Returning to Fig. 1, it is easy to verify that the solution shown in Fig. 1(a) is the product of the FF algorithm on the permutation $[A, B, C, D, E, F, G, H, I]$ of the $K = 9$ connections. On the other hand, the equivalent solution in Fig. 1(b) cannot be a product of the FF algorithm. Consistent with our result in [20], by re-allocating connection C (i.e., the one spanning only Link 1) to slots 5 and 6 from slots 8 and 9, we obtain a new solution that would be produced by the FF algorithm, say, on the permutation $[A, D, E, C, F, G, B, H, I]$ of the $K = 9$ connections. Similar observations apply to the solution in Fig. 1(c).

2.4. Spectrum allocation as a permutation problem

The FF optimality property opens up new directions for optical network design. For the first time, it is now possible to design a new class of SA algorithms that do not have to sift through the exponential number of symmetric solutions derived from spectrum permutations such as the ones in Figs. 1(b) and 1(b). Specifically, the FF optimality property has two implications:

- First, it essentially transforms the SA problem into a *permutation problem*: to find an optimal solution to the SA problem, it is sufficient to examine the connection permutations.
- Second, it shows that in selecting among the various connection permutations there is no need to consider a spectrum allocation other than the one produced by the FF algorithm.

These observations suggest a procedure for finding the unknown permutation $P^{\star}_{FF}$:

Enumerate all connection permutations and select the one for which the FF algorithm yields the smallest objective value.

Although the number $K!$ of connection permutations is exponential, reducing spectrum allocation to a permutation problem allows for the elimination of all symmetric solutions and drastically reduces the size of the solution space that needs to be explored.

Accordingly, we developed recursive first-fit (RFF), a symmetry-free optimal branch-and-bound algorithm for the SA problem [20]. RFF searches the entire space of connection permutations to find one that is optimal for the problem instance at hand, applying the FF algorithm as it incrementally builds each permutation during the search. RFF represents a significant improvement over existing approaches as it completely sidesteps the spectrum symmetry challenge.

In a network with N nodes and traffic between all node pairs, the size K of the connection set $\mathcal{T}$ is $O(N^2)$. Therefore, any algorithm, such as RFF, that considers all possible connection permutations to determine the optimal spectrum allocation must take time that is exponential in the size of the network, $O(N^2!)$. Next, we present a parameterized heuristic that is inspired by the permutation model of spectrum allocation. The heuristic applies the FF algorithm to a subset of the permutation space whose size can be customized to the available computational budget.

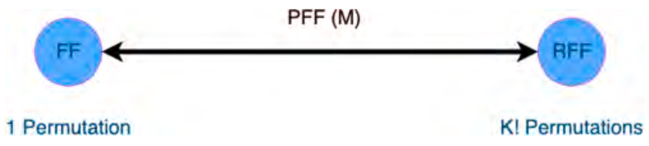


Fig. 2. PFF(M) explores the spectrum of connection permutations between the two extremes, FF and RFF.

3. Parameterized First Fit (PFF)

3.1. Motivation

Our motivation for a new algorithm for the SA problem stems from the observation, illustrated in Fig. 2, that the FF heuristic and the optimal RFF algorithm we developed in [20] represent two opposite extremes in exploring the solution space of connection permutations. The FF heuristic applies the FF algorithm to a single permutation, whereas, given sufficient time, RFF will explore all $K!$ permutations. While RFF can be executed in parallel [20], exploring the entire permutation space for SA problem instances encountered in practice would be infeasible. Typically, there exists a budget in terms of the amount of computational resources (or time) allotted to tackling network design problems such as spectrum allocation. Therefore, any algorithm, either optimal or heuristic in nature, that runs for a limited amount of time is bound to explore only the region of the solution space around its starting point, and hence fail to find optimal solutions that may exist in different parts of the space.

Furthermore, algorithms that operate in a branch-and-bound fashion, including RFF and those employed by integer linear programming (ILP) solvers, are sensitive to the values of the input parameters, in this case the spectrum demands t_i. Consequently, given two different problem instances on the same topology graph $G = (V, A)$ and number of connections K, branch-and-bound algorithms will explore a different number of permutations (i.e., a different fraction of the solution space) for each instance within a prescribed amount of running time. In addition, it cannot be known *a priori* for which instance the algorithm will explore a larger or smaller fraction of the solution space; and it may be difficult to determine the relative size of the solution space explored for each instance after the algorithm has completed.

Parameterized first fit, PFF(M), where M is a parameter such that $1 \leq M \leq K$, is a generalization of the FF and RFF algorithms that operates in the vast space between these two extremes, as shown in Fig. 2. Rather than searching the whole solution space of size $O(K!)$ from some initial and often arbitrary starting point, the key idea of PFF(M) is to completely explore a subset of the solution space of size $O(M!)$, where $M \leq K$, and, typically, $M \ll K$. Ideally, the $M!$ connection permutations to be explored should be distributed evenly across the whole solution space. One strategy for achieving this goal would be to generate the $M!$ permutations randomly. Instead, PFF(M) takes a structured approach to generating the $M!$ permutations that has several benefits for network designers:

- it provides a well-defined tradeoff between the size of the permutation space to be explored and the amount of computational resources available, via the choice of the value of parameter M;
- it employs a method that is readily reproducible, works for every value of parameter M, and explores the exact same subset of the solution space for any two instances of the same SA problem,
- it explores connection permutations that are spread over diverse regions of the solution space, and
- it provides better and more even coverage of the solution space as the value of M increases.

3.2. The PFF algorithm

In determining a solution to an instance of the SA problem, PFF(M) considers only subsets of the solution space of size equal to that of a (smaller) permutation space, i.e., one of size $O(M!), M \leq K$. It then generates $M!$ connection permutations that are spread over the entire original solution space.

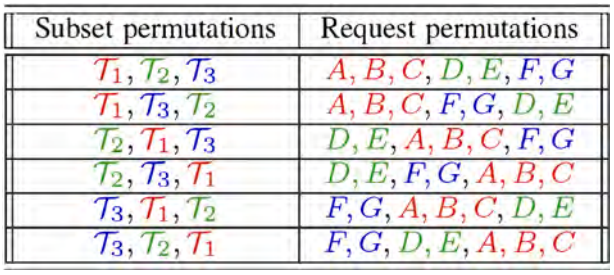
To achieve this objective, PFF(M) first partitions the set $\mathcal{T}$ of K connections into $M \leq K$ subsets, $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_M$, and generates all $M!$ permutations of the M subsets. We assume that the connections within each subset are listed in a fixed order, and we consider this list as a single *meta-connection*. In essence, then, this operation produces all $M!$ meta-connection permutations.

PFF(M) then generates the $M!$ *connection* permutations to evaluate by parsing each meta-connection permutation and replacing each meta-connection with the list of its constituent connections. This approach produces connection permutations that are spread over the entire solution space. To see this, note that two meta-connection permutations that differ in the first meta-connection will produce two connection permutations that are far apart in the connection permutation space; similar observations apply to permutations that differ in the second, third, etc., meta-connection. In other words, replacing each meta-connection with its individual connections involves large jumps in the connection permutation space, achieving our objective of generating connection permutations that cover the entire solution space.

Finally, PFF applies the FF algorithm to the $M!$ connection permutations created in this manner, and selects the permutation that results in the best solution to the SA problem.

Fig. 3 illustrates this concept for the set $\mathcal{T} = \{A, B, C, D, E, F, G\}$ of $K = 7$ connections partitioned into three subsets, i.e., meta-connections, $\mathcal{T}_1, \mathcal{T}_2$, and $\mathcal{T}_3$. There are several options for partitioning the set $\mathcal{T}$ into subsets. In this work, we only consider partitions in which the sizes of the various subsets vary by at most one. Therefore, each subset $\mathcal{T}_i$ is such that $|\mathcal{T}_i| = \lfloor K/M \rfloor$ or $\lfloor K/M \rfloor + 1$. Without loss of generality, we determine the subsets such that $|\mathcal{T}_1| \geq |\mathcal{T}_2| \geq \cdots \geq |\mathcal{T}_M|$. Therefore, the $M = 3$ subsets of $\mathcal{T}$ are: $\mathcal{T}_1 = \{A, B, C\}$, $\mathcal{T}_2 = \{D, E\}$, and $\mathcal{T}_3 = \{F, G\}$. Each subset is shown in Fig. 3 in a different color.

The left column of Fig. 3 shows the $M! = 6$ subset permutations (or meta-connection permutations, as we mentioned above). The right column of the figure shows the corresponding 6 connection permutations obtained by replacing each subset (meta-connection) with its constituent connections. Although this is a small number compared to the $7! = 5040$ possible connection permutations to provide adequate coverage of the solution space, it is evident that these six permutations



Subset permutations	Request permutations
$\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$	A, B, C, D, E, F, G
$\mathcal{T}_1, \mathcal{T}_3, \mathcal{T}_2$	A, B, C, F, G, D, E
$\mathcal{T}_2, \mathcal{T}_1, \mathcal{T}_3$	D, E, A, B, C, F, G
$\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_1$	D, E, F, G, A, B, C
$\mathcal{T}_3, \mathcal{T}_1, \mathcal{T}_2$	F, G, A, B, C, D, E
$\mathcal{T}_3, \mathcal{T}_2, \mathcal{T}_1$	F, G, D, E, A, B, C

Fig. 3. PFF permutation space for the set $\mathcal{T} = \{A, B, C, D, E, F, G\}$ of $K = 7$ connection requests, partitioned into $M = 3$ subsets, $\mathcal{T}_1 = \{A, B, C\}, \mathcal{T}_2 = \{D, E\}, \mathcal{T}_3 = \{F, G\}$.

are spread across different regions of the space. In this example, PFF will evaluate only the 6 permutations shown in the right column of the figure by running the FF algorithm on each.

Algorithm 1 shows the operation of PFF(M). The preprocessing step in Lines 1-6 generates the M subsets of the connection set $\mathcal{T}$, and from them the $M!$ connection permutations that the algorithm considers. This step takes time $O(M!)$, but since M is determined by the network designer and is not part of the input to the problem, the running time can be considered as having a fixed value. Note that a network designer may have to solve multiple instances for a given SA problem defined by the network topology $G = (V, A)$ and number of spectrum requests K; for instance, this may be due to carrying out a "what-if" analysis to explore the sensitivity of design decisions to forecast traffic demands. In this case, the designer only needs to perform the preprocessing step once, store the $M!$ permutations, and use them to solve all instances that are part of the analysis. Therefore, the computational cost of this step can be amortized over multiple problem instances.

The main part of the algorithm in Lines 7-13 simply runs the FF algorithm on each of the $M!$ permutations generated in the preprocessing step, and selects the one that offers the best solution to the problem at hand. Each application of the FF algorithm takes time $O(KL)$, as each permutation consists of K requests and each request may involve any of the L links in the network. Therefore, the total running time of this part of the algorithm is $O(KLM!)$, where M is again considered as having a fixed value.

We also note that Steps 9-11 of PFF are applied separately to each connection permutation. Therefore, these steps may be executed in parallel by locking access to the variables $BestSOL$ and $BestP$ and spawning multiple threads, each thread executing Steps 9-11 on a different connection permutation. Assuming that up to R threads may execute in parallel, applying the FF algorithm to the $M!$ permutations will require $\lceil M!/R \rceil$ batches of parallel threads, reducing the time complexity of the PFF algorithm to $O(KL(M!/R))$.

The PFF(M) algorithm encompasses the FF heuristic (for $M = 1$) and the optimal RFF algorithm (for $M = K$), as special cases. Parameter M affords the network designer a wide range of options between these two extremes, and its value can be selected so as to strike an appropriate balance between the quality of solution and the running time $O(KL(M!/R))$ of the FF application step of the algorithm. In particular, the degree of parallelism, R, which generally depends on the operating system and/or computing environment, has a direct impact on the maximum value of M that can be used. Let us assume that Steps 9-11 of PFF (i.e., the FF algorithm) take s time units to execute on a single permutation, and there is a time budget of $S > s$ units for obtaining a solution. Then, the value of M must be selected such that $M! \leq RS/s$. In other words, the number of permutations that may be explored within a given time budget increases linearly with R.

As a final note, let SOL^M_{PFF} denote the value of the best solution returned by PFF(M). Incrementing the value of parameter M from, say, m to $m + 1$, results in an increase in the size of the solution space explored by a factor of $m + 1$. Nevertheless, despite the fact

Algorithm 1 Parameterized First Fit

```
Input:
    $G = (V, A)$: network topology
    $\mathcal{T} = \{T_i = (s_i, d_i, p_i, t_i\}$: set of connections
    $K = |\mathcal{T}|$: number of connections
Output:
    $BestP$: best connection permutation
    $BestSOL$: SA solution corresponding to $BestP$

PFF($M, 1 \le M \le K$)
1: {Preprocessing: Generate $M!$ connection permutations}
2: Partition $\mathcal{T}$ into $M$ subsets, $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_M$, such
   that $|\mathcal{T}_i| = \lfloor K/M \rfloor$ or $\lfloor K/M \rfloor + 1$;
3: Generate all $M!$ permutations of the $M$ subsets;
4: for $i = 1; i \le M!; i{+}{+}$ do
5:     $P_i \leftarrow$ connection permutation created by replacing
       each subset in subset permutation $i$ with its connections;
6: end for
7: {Main: Apply FF to the $M!$ connection permutations}
8: for $i = 1; i \le M!; i{+}{+}$ do
9:     $S \leftarrow SOL(P_i)$; {solution obtained by FF on $P_i$}
10:    if $S < BestSOL$ then
11:        $BestSOL = S$;  $BestP = P_i$;
12:    end if
13: end for
14: return;
```

that PFF($m+1$) evaluates a larger number of connection permutations than PFF(m), it does *not* necessarily follow that $SOL^{m+1}_{PFF} \le SOL^{m}_{PFF}$. Intuitively, since PFF($m+1$) and PFF(m) each evaluate a different set of connection permutations, it is conceivable that a permutation in the set of PFF(m) may yield a solution that is better than those produced by permutations in the set of PFF($m+1$). Our experiments confirm this observation as we have occasionally found that $SOL^{m}_{PFF} < SOL^{m+1}_{PFF}$ on the same problem instance. Therefore, in this work we apply the PFF(M) algorithm as follows:

(1) Run PFF(m) for all $m = 1, \dots, M$.
(2) Return $SOL_{PFF} = \min_{m=1,\dots,M}\{SOL^{m}_{PFF}\}$.

Consequently, the solutions in Step 2 above are monotonically non-increasing as a function of m.

4. Simulation study

4.1. Simulation setup

We now present the results of simulation experiments we have carried out to evaluate the performance of the PFF(M) heuristic we presented in the previous section. We run the experiments on the Henry2 Linux HPC cluster at NC State University [22–24] which consists of more than 1000 compute nodes and over 10,000 cores.

The experimental setup is similar to the one we used in [20]. Specifically, we consider two network topologies, NSFNET and GEANT2 shown in Fig. 4, and route each connection along the minimum-hop path between its source and destination nodes. The number of nodes, links, connections, and meta-connections for the two topologies that we used in our simulations are listed in
Table 1; note that in our study we used the maximum possible number of connections for each network, i.e., one connection between every node pair for a total of $K = N(N-1)/2$ connections, where N is the number of nodes.

We create SA problem instances by generating connections between all pairs of nodes in the network. We consider data rates of 10, 40, 100, 400, and 1000 Gbps. For a given problem instance, we generate a

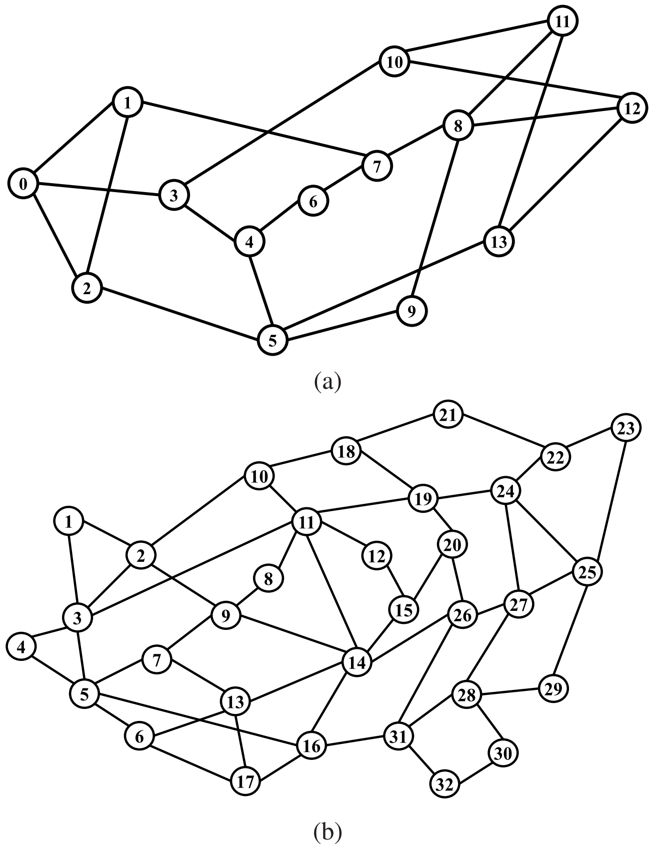


Fig. 4. Network topologies used in our study: (a) NSFNET, (b) GEANT2.

random value for the demand between a pair of nodes based on one of three distributions: (1) *Uniform*: each of the five rates is selected with equal probability; (2) *Skewed low*: the rates above are selected with probability 0.30, 0.25, 0.20, 0.15, and 0.10, respectively; or (3) *Skewed high*: the five rates are selected with probability 0.10, 0.15, 0.20, 0.25, and 0.30, respectively. Once the traffic rates between each node pair have been generated, we calculate the corresponding spectrum slots by assuming that the slot width is 12.5 GHz, and adopting the parameters of [25] to determine the number of spectrum slots that each connection requires based on its data rate and path length. Specifically, we consider two modulation formats: for paths with up to (respectively, more than) ten links we assume 16-QAM (respectively, QPSK), such that demands of size 10, 40, 100, 400, and 1000 Gbps are assigned 1, 1, 2, 8, and 20 (respectively, 1, 2, 4, 16, and 40) slots, consistent with the values used in [25, Table 1]. For each traffic distribution, we generate 100 random problem instances.

We consider the highest index of allocated spectrum slots on any network link as the performance measure to compare the various algorithms. For a meaningful comparison between different problem instances, we normalize the solutions returned by the various algorithms by dividing with the lower bound for the corresponding instance. A lower bound LB on the optimal objective value may be obtained by ignoring the spectrum contiguity and continuity constraints and simply counting the spectrum slots required by all connections routed along the most congested link:

$$LB = \max_{l \in A} \left\{ \sum_{T_i \in \mathcal{T} : l \in p_i} t_i \right\}. \quad (1)$$

In the experiments, we varied the value of parameter M of the PFF(M) heuristic as $M = 1, 2, 3, \dots, 8$. We start with a permutation P in

Table 1
Values of network topology parameters. N: number of nodes, L: number of (bidirectional) links, $K = N(N-1)/2$: number of connections, M: number of meta-connections (defined in Section 3.2).

	N	L	K	M
NSFNET	14	21	91	$1, 2, \dots, 8$
GEANT2	32	54	496	$1, 2, \dots, 8$

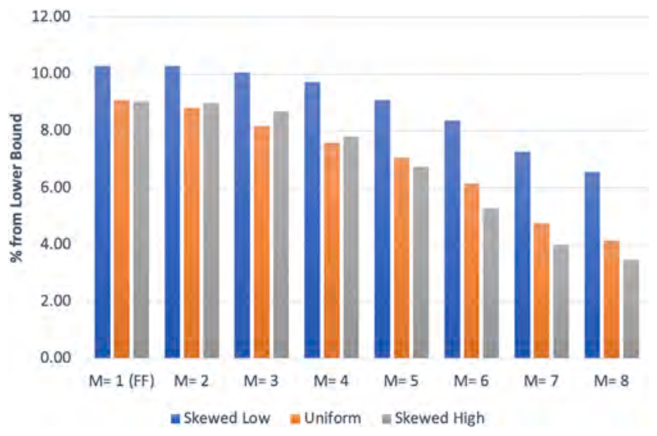


Fig. 5. PFF(M) solution quality as % from LB, NSFNET.

which the K traffic requests are listed in decreasing order of spectrum demand t_i, and requests with the same demand are listed in decreasing order of path length. For each value of $M \geq 2$, we partition permutation P into M subsets as we described in the previous section. Recall that when $M = 1$, PFF(1) considers only one permutation, P, and hence it is equivalent to the FF heuristic. In all experiments we deployed $R = 32$ parallel threads, the maximum number available to us on the Henry2 cluster.

4.2. Solution quality

Figs. 5 and 6 summarize the results we have obtained regarding the relative performance of the PFF(M) heuristic, $M = 1, 2, \dots, 8$, with respect to the lower bound. Specifically, for each value of M, the figures show how far the solutions obtained by PFF(M) are from the lower bound for problem instances generated with the skewed low, uniform, and skewed high traffic distributions. The values shown are averages over the 100 problem instances from the specified distribution.

As we can see for the NSFNET topology in Fig. 5, the FF algorithm (i.e., PFF(1)) performs well and, on average, produces solutions that are within 9%–10% of the lower bound across the 300 problem instances we used in our experiments. These results are consistent with earlier research indicating that the FF algorithm finds solutions of good quality. Turning our attention to the PFF(M) results for $M = 2, \dots, 8$, we observe that initially, as M increases to 2 and 3, the quality of the solutions improves only slightly relative to FF. This is not surprising, as the number of permutations that PFF considers only increases to 2 (for PFF(2)) and 6 (for PFF(3)). As M increases further, the solution quality shows more pronounced improvement; for $M = 8$, the solutions found by PFF are within 3.5–6.5% of the lower bound on average, depending on the traffic distribution, a substantial reduction relative to the FF solutions.

Similar observations apply to the GEANT2 results shown in Fig. 6. We note that FF produces solutions of high quality, within 2%–5% of the lower bound, on average, across the GEANT2 problem instances. Nevertheless, PFF(M) was able to improve on the FF results across all values of $M = 2, \dots, 8$. For $M = 8$, specifically, PFF constructs solutions that are within 2% of the lower bound for the skewed low

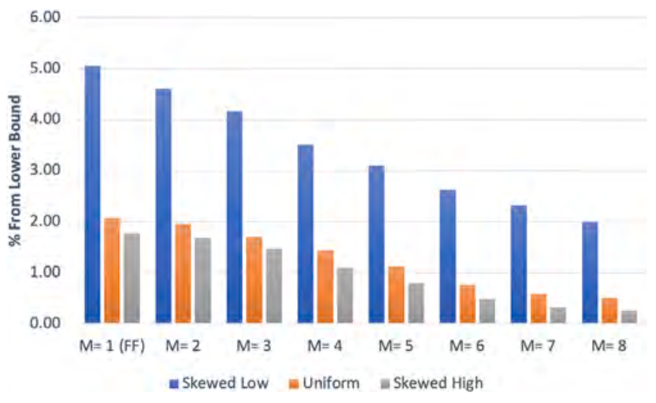


Fig. 6. PFF(M) solution quality as % from LB, GEANT2.

distribution and within 0.5% of the lower bound for the other two traffic distributions.

Figs. 5 and 6 indicate that both the FF and PFF(M), $M \geq 2$, algorithms produce solutions that are closer to the lower bound under the uniform and skewed high traffic distributions compared to the solutions under the skewed low distribution, for both the NSFNET and GEANT2 topologies. This can be explained by noting that, on average, the data rate of a connection is 193 Gbps under the skewed low distribution, 312 Gbps under the uniform distribution, and 461 Gbps under the skewed high distribution. Since the random problem instances differ only in terms of the traffic demands (i.e., the number of connections and their paths are exactly the same), it follows that the spectrum slot requirements of instances generated under the skewed low distribution are substantially smaller than those of instances generated with the other two distributions. As a result, both the lower bound and the solution values are lower under the skewed low distribution. Therefore, the ratio of solution values to the lower bound, as plotted in Figs. 5 and 6, tends to be higher for the skewed low distribution: even if the absolute difference between the solution and the lower bound is the same for instances generated by different distributions, the ratio will be higher when the denominator (i.e., the lower bound) is smaller. As we can see from the two figures, this relative gap in performance holds true for the FF algorithm; but although PFF(M) starts at a larger gap for the skewed low distribution, the relative *improvement* is similar to the other two distributions as M increases. Moreover, this relative gap is higher for the GEANT2 topology when there are $K = 496$ connections contributing to the lower bound (rather than $K = 91$ connections for NSFNET), further increasing the difference, in absolute terms, between the lower bound of the skewed low and that of the other two distributions. Overall, the higher ratios observed in the two figures are mostly due to the absolute values of demands, not to the inability of the PFF (and FF) algorithm to find solutions close to the lower bound.

Figs. 7 and 8 present a different perspective regarding the quality of the solutions obtained for the NSFNET and GEANT2 topologies, respectively. Specifically, the two figures plot the number of problem instances for which the PFF(M) solutions are either (1) better than the corresponding FF solution (denoted as "<FF" in the figures), or (2) equal to the lower bound LB of the corresponding instance (denoted as "=LB"). To interpret the results in the two figures we note that:

- a solution that is equal to LB is an optimal one;
- a solution with a value higher than LB but lower than the corresponding FF solution *may* be optimal, but we cannot say for certainty as the optimal value is unknown in this case; and
- as we explained in the previous section, the PFF(M) heuristic returns solutions that are monotonically non-increasing as a function of M, and hence, for any instances that it cannot find a better solution it returns the FF (i.e., PFF(1)) solution.

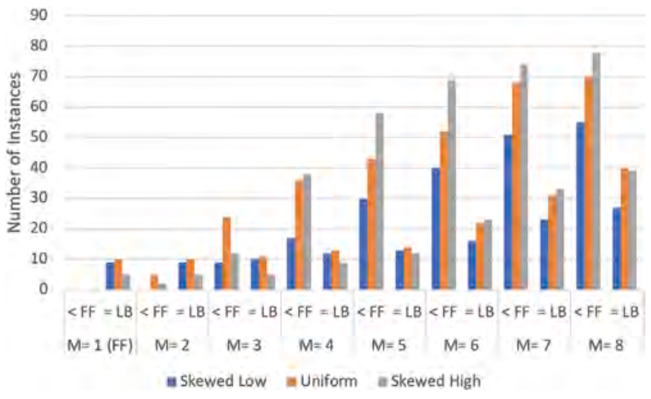


Fig. 7. PFF(M) solutions relative to FF and LB, NSFNET.

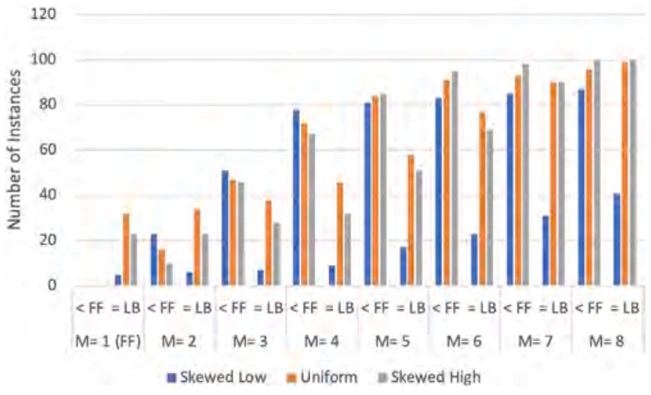


Fig. 8. PFF(M) solutions relative to FF and LB, GEANT2.

From the two figures we observe that PFF(M) can find solutions that are better than those of FF, for at least some problem instances, even when $M = 2$. As M increases, the number of instances for which the PFF solution is either better than the FF solution or equal to the lower bound, increases further. When $M = 8$, PFF improves upon the FF solution in 55%–78% of the problem instances in the case of the NSFNET, and in 87%–100% of the instances in the case of GEANT2, depending on the traffic distribution. Furthermore, it finds an optimal solution in 27%–40% (respectively, 41%–98%) of the instances for NSFNET (respectively, GEANT2), again depending on the traffic distribution.

4.3. Running time

Fig. 9 plots the running time of the PFF(M) algorithm as a function of M for the GEANT2 topology; we omit results for the NSFNET topology as the running time shows very similar trends but the absolute values are smaller. The algorithm was run in parallel using $R = 32$ threads. The running time of PFF does not depend on the traffic distribution, hence the values shown are representative of all three distributions we have used in our experiments. We first observe that the running time does not change as M increases from 1 to 4. This can be explained by the fact that $4! = 24$, and hence all connection permutations up to $M = 4$ can be handled by a single batch of 32 threads. Starting with $M = 5$, multiple batches of 32 threads must be deployed (sequentially) to apply the FF algorithm to all $M!$ permutations, and hence the running time starts increasing with M. As expected, as the value of M is incremented, say, from m to $m+1$, the running time increases by approximately a factor of $m+1$.

The most important observation from Fig. 9 is that within just 800 s (i.e., in less than 15 min), the PFF(8) algorithm can produce solutions

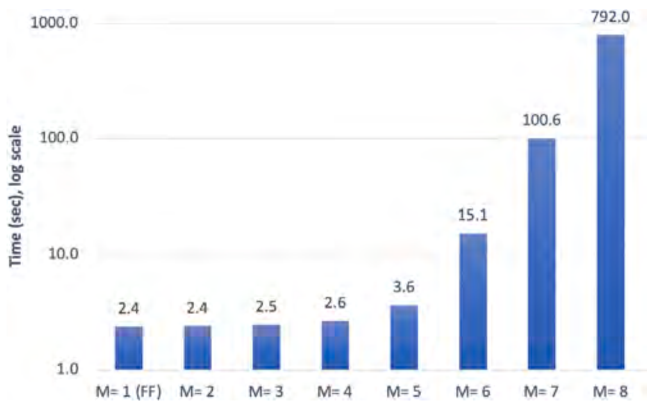


Fig. 9. Running time of PFF(M) with $R = 32$ threads as a function of M, GEANT2 topology.

that are significantly better than those of FF and very close to the lower bound, as shown in Figs. 6 and 8. The PFF(8) algorithm also constructs solutions of quality comparable to those of the RFF algorithm [20,26]. Overall, these results indicate that, by examining permutations that are spread across the entire solution space, PFF makes effective use of computational resources in identifying solutions of high quality.

5. Concluding remarks

We have developed parameterized first-fit (PFF), a new heuristic for the SA problem that completely sidesteps the spectrum symmetry challenge and can identify near-optimal solutions to moderate-size networks in minutes. We plan to extend this work in two directions: we will explore the potential of PFF in larger-size networks and a wider range of traffic distributions, and we will develop spectrum symmetry-free algorithms for optical network design problems that include SA as a subproblem. To this end, a natural research direction would be to combine PFF with the routing algorithms we developed in [27,28] to tackle large RSA problems efficiently.

Declaration of competing interest

One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have potential conflict of interest with this work. For full disclosure statements refer to https://doi.org/10.1016/j.osn.2023.100742.George Rouskas reports financial support was provided by National Science Foundation.

Data availability

Data will be made available on request.

References

[1] J.M. Simmons, Optical Network Design and Planning, Springer, 2008.
[2] J. Simmons, G.N. Rouskas, Routing and wavelength (spectrum) allocation, in: B. Mukherjee, I. Tomkos, M. Tornatore, P. Winzer, Y. Zhao (Eds.), Springer Handbook of Optical Networks, Springer, 2020.
[3] G.N. Rouskas, Routing and wavelength assignment in optical WDM networks, in: J. Proakis (Ed.), Wiley Encyclopedia of Telecommunications, John Wiley & Sons, 2001.
[4] B. Jaumard, C. Meyer, B. Thiongane, Comparison of ILP formulations for the RWA problem, Opt. Switch. Netw. 3–4 (2007) 157–172.
[5] M. Klinkowski, P. Lechowicz, K. Walkowiak, Survey of resource allocation schemes and algorithms in spectrally-spatially flexible optical networking, Opt. Switch. Netw. 27 (C) (2018) 58–78.

[6] S. Talebi, F. Alam, I. Katib, M. Khamis, R. Khalifah, G.N. Rouskas, Spectrum management techniques for elastic optical networks: A survey, Opt. Switch. Netw. 13 (2014) 34–48.
[7] B. Chen, G.N. Rouskas, R. Dutta, Clustering methods for hierarchical traffic grooming in large-scale mesh WDM networks, J. Opt. Commun. Netw. 2 (2010) 502–514.
[8] R. Dutta, G.N. Rouskas, A survey of virtual topology design algorithms for wavelength routed optical networks, Opt. Netw. 1 (2000) 73–89.
[9] H. Wang, G.N. Rouskas, Hierarchical traffic grooming: A tutorial, Comput. Netw. 69 (2014) 147–156.
[10] D. Zhou, S. Subramaniam, Survivability in optical networks, IEEE Netw. 14 (2000) 16–23.
[11] S. Talebi, E. Bampis, G. Lucarelli, I. Katib, G.N. Rouskas, Spectrum assignment in optical networks: A multiprocessor scheduling perspective, J. Opt. Commun. Netw. 6 (2014) 754–763.
[12] R. Ramaswami, K. Sivarajan, Routing and wavelength assignment in all-optical networks, IEEE/ACM Trans. Netw. 3 (1995) 489–500.
[13] B. Jaumard, C. Meyer, B. Thiongane, ILP formulations for the routing and wavelength assignment problem: Symmetric systems, in: Handbook of Optimization in Telecommunications, Springer US, 2006, pp. 637–677.
[14] F. Bertero, M. Bianchetti, J. Marenco, Integer programming models for the routing and spectrum allocation problem, TOP 26 (2018) 465–488.
[15] L. Velasco, M. Ruiz, M. Klinkowski, The Routing and Spectrum Allocation Problem, John Wiley & Sons, Ltd, 2017, pp. 43–60, ch. 3.
[16] J. Wang, M. Shigeno, Q. Wu, ILP models and improved methods for the problem of routing and spectrum allocation, Opt. Switch. Netw. 45 (2022) 100675.
[17] E. Yetginer, Z. Liu, G.N. Rouskas, Fast exact ILP decompositions for ring RWA, J. Opt. Commun. Netw. 3 (2011) 577–586.
[18] H. Zang, J.P. Jue, B. Mukherjee, A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks, Opt. Netw. 1 (2000) 47–60.
[19] Y. Zhu, G.N. Rouskas, H.G. Perros, A comparison of allocation policies in wavelength routing networks, Photonic Netw. Commun. 2 (2000) 265–293.
[20] G.N. Rouskas, C. Bandikatla, Recursive first fit: A highly parallel optimal solution to spectrum allocation, IEEE/Opt. J. Opt. Commun. Netw. 14 (2022) 165–176.
[21] G.N. Rouskas, P. Sharma, S. Gupta, Parameterized first fit (PFF): Eliminating symmetry in spectrum allocation, in: Proceedings of IEEE GLOBECOM 2022, 2022.
[22] NC State Henry2 Linux Cluster. https://projects.ncsu.edu/hpc/About/ComputeResources.php.
[23] C. Castillo, G.N. Rouskas, K. Harfoush, Efficient resource management using advance reservations for heterogeneous grids, in: Proceeding of the Twenty Second IEEE International Parallel and Distributed Processing Symposium, IPDPS 2008, 2008.
[24] V. Sivaraman, G.N. Rouskas, HiPeR-ℓ: A high performance reservation protocol with ℓook-ahead for broadcast WDM networks, in: Proceedings of INFOCOM '97, IEEE, 1997, pp. 1272–1279.
[25] M. Jinno, B. Kozicki, H. Takara, A. Watanabe, Y. Sone, T. Tanaka, A. Hirano, Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network, IEEE Commun. Mag. 48 (8) (2010) 138–145.
[26] G.N. Rouskas, S. Gupta, P. Sharma, Experimental evaluation of a symmetry-free parallel algorithm for spectrum allocation, in: Proceedings of IEEE GLOBECOM 2022, 2022.
[27] M. Fayez, I. Katib, G.N. Rouskas, T.F. Gharib, H. Faheem, A scalable solution to network design problems: Decomposition with exhaustive routing search, in: Proceedings of IEEE GLOBECOM 2020, 2020.
[28] G.N. Rouskas, C. Bandikatla, Parameterized exhaustive routing with first fit for RSA problem variants, in: Proceedings of IEEE GLOBECOM 2021, 2021.