JOURNAL OF Optical Communications and Networking

Symmetry-free routing and spectrum assignment: a universal algorithm based on first-fit

George N. Rouskas
North Carolina State University, Raleigh, North Carolina 27695, USA (rouskas@ncsu.edu)

Received 21 February 2024; revised 7 July 2024; accepted 7 July 2024; published 12 August 2024

First-fit (FF) is a well-known and widely deployed algorithm for spectrum assignment (SA), but until our recent study [J. Opt. Commun. Netw. 14, 165 (2022)], investigations of the algorithm had been experimental in nature and no formal properties of the algorithm with respect to SA were known. In this work, we make two contributions. First, we show that FF is a *universal* algorithm for the SA problem in the sense that, for any variant, 1) it can be used to construct solutions equivalent to, or better than, any solution obtained by any other algorithm, and 2) it can construct an optimal solution. This universality property applies to both the min-max and min-frag objectives and to variants of the SA problem with or without guard band constraints. Consequently, the spectrum symmetry-free model of our recent study [J. Opt. Commun. Netw. 14, 165 (2022)] extends to all known SA variants, which therefore reduce to permutation problems. Second, we extend the spectrum symmetry-free model to the routing and spectrum assignment (RSA) problem in general topologies. This model allows for the design of more efficient algorithms as it eliminates from consideration an exponential number of equivalent symmetric solutions. By sidestepping symmetry, the RSA solution space is naturally and optimally decomposed into a routing space and a connection permutation space. Building upon this property, we introduce a two-parameter, symmetry-free *universal* algorithm that can be used to tackle any RSA variant in a uniform manner. The algorithm is amenable to multi-threaded execution to speed up the search process, and the value of the parameters can be adjusted to strike a balance between running time and solution quality. Our evaluation provides insight into the relative benefits of path diversity (which determines the size of the routing space) and connection diversity (which determines the size of the permutation space). © 2024 Optica Publishing Group

https://doi.org/10.1364/JOCN.521978

1. INTRODUCTION

Routing and spectrum assignment (RSA) and its variants are fundamental problems in the design and control of elastic optical networks [1–6] and underlie a range of optimization problems including virtual topology design [7,8], traffic grooming [9–11], and network survivability [12,13]. The RSA problem is NP-hard [4], even in simple network topologies [14], and a wide range of integer linear programming (ILP) formulations have been developed to tackle it [5,15–17].

Conventional ILP formulations for RSA face a significant challenge due to *spectrum symmetry*, i.e., the fact that spectrum slots are *interchangeable* [18]. As explained in [19] with reference to the RWA problem, "as the wavelengths are interchangeable, given an optimal solution of the RWA problem or of one of its continuous relaxation, one can derive a large number of equivalent solutions using any permutation of the wavelengths." In other words, ILP solvers must evaluate an exponential number of distinct but equivalent optimal solutions and hence their running time can be unnecessarily long [19].

The spectrum allocation (SA) subproblem of RSA underlies much of optical network design [1] and has been studied extensively. The SA problem is intractable even when considered in isolation, i.e., separately from other aspects of network design [14]. Consequently, a number of heuristic algorithms, including first-fit, best-fit, most-used, and least loaded [20], have been developed and studied experimentally.

The first-fit (FF) algorithm is a simple heuristic for the SA problem that operates without global knowledge and has been shown to be effective across a wide range of network topologies and traffic demands [20–22]. Accordingly, it is commonly employed in practice. Even before its application to spectrum assignment, the FF algorithm had been investigated theoretically since the early days of computing in the context of bin packing [23] and memory allocation [24], among other fields. For instance, FF has been shown to be an approximation algorithm for the bin packing problem and a series of studies over four decades gradually improved the approximation ratio [23,25,26]. To the best of our knowledge, however, before our study in [18], investigations of the FF algorithm within the context of spectrum assignment, on its own or as part of optical network optimization problems, were purely experimental and no formal properties of the algorithm were known.

With this work we make two contributions. First, we carry out a theoretical investigation of FF and show that it is a *universal* algorithm for all known variants of the SA problem. In this paper, we use the term *universal* to indicate that, for any instance of any SA problem variant, the pure FF algorithm or an appropriately modified version is capable of 1) constructing a solution that is equivalent to, or better than, any solution obtained by any other algorithm and 2) constructing an optimal solution. This universality property implies that, to find an optimal solution to any SA problem variant, it is sufficient to consider only the connection permutations and apply the FF algorithm. This insight forms the basis for a *symmetry-free model* for spectrum assignment in networks of general topology. This model eliminates from consideration symmetric solutions, i.e., equivalent solutions derived from spectrum slot permutations, and opens up new algorithmic directions for the SA problem specifically and optical network design more generally.

As our second contribution, we extend the spectrum symmetry-free model to the RSA problem. We show that, by eliminating symmetric solutions, the solution space naturally and optimally decomposes into a routing space and a connection permutation space. We also introduce a universal two-parameter RSA algorithm that explores exhaustively a subset of the solution space that encompasses the complete solution space for a subset of the connections. The values of the two parameters may be used to adjust the size of the solution space to strike a balance between running time and solution quality. Finally, our study provides insight into the relative benefits of path diversity (which determines the size of the routing space) and connection diversity (which determines the size of the permutation space).

The remainder of the paper is organized as follows. In Section 2 we show that FF is a universal algorithm for known SA variants and forms the basis for symmetry-free algorithmic solutions. In Section 3, we extend the spectrum symmetry-free model to the RSA problem. Building on that insight, we develop and evaluate a universal two-parameter algorithm for RSA. We conclude the paper in Section 4.

2. FIRST-FIT: A UNIVERSAL ALGORITHM FOR SYMMETRY-FREE SPECTRUM ASSIGNMENT

In this section we prove that FF is a *universal algorithm for all known variants of the SA problem* (i.e., when routing is fixed and not part of the optimization). We first review our recent results [18] regarding the min-max SA problem, the variant that has been most extensively studied in the literature. Specifically, we discuss the optimality property of FF with respect to min-max SA that allowed us to develop the first symmetry-free model for networks of general topology. In the following three subsections, we extend these results to several SA variants, and specifically, min-max DSA (Section 2.B), min-frag SA (Section 2.C), and min-frag DSA (Section 2.D).

A. Min-Max SA as a Permutation Problem

Let $G = (V, A)$ represent the topology graph of an optical network with a set V of nodes and a set A of directed fiber links, where $N = |V|$ and $L = |A|$ denote the number of nodes and links, respectively. The traffic offered to the network consists of C connections T_i, $i = 1, \ldots, C$, where each connection is a tuple $T_i = (s_i, d_i, p_i, t_i)$ such that s_i and d_i are the endpoints 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 along path p_i. (In this and the next three subsections, we focus on the SA problem and assume that the path p_i for each connection T_i has been determined. We will remove this assumption in Section 3 where we discuss the RSA problem.)

We assume that the spectrum slots on each link are indexed $1, 2, 3, \ldots$, and we define the min-max SA problem as follows:

Definition 2.1 (min-max SA):

Input: graph G and C connections $\{T_i = (s_i, d_i, p_i, t_i)\}$.

Output: an assignment of t_i spectrum slots to each connection T_i along its path p_i.

Objective: minimize the index of the highest spectrum slot used on any link in the network.

Constraints:

1) *Contiguity:* Each connection T_i is assigned a block of t_i contiguous spectrum slots starting at index f_i, i.e., block $[f_i, f_i + 1, \ldots, f_i + t_i - 1]$.
2) *Continuity:* Each connection is assigned the same block of slots on each link $\ell \in p_i$ along its path p_i.
3) *Non-overlap:* Connections whose paths share a link are assigned non-overlapping blocks of spectrum slots.

The FF algorithm [20] considers connections in a fixed order and assigns to each one a contiguous block of spectrum slots that starts at the lowest-indexed slot available along the links of the connection's path. Let P denote a permutation of the C connections, and let $FF(P)$ denote the solution constructed by applying the FF algorithm to the connections in the order implied by P. In recent work [18] we have shown that there exists a permutation $P^\star$ of the C connections such that $FF(P^\star)$ is an optimal solution to the min-max SA problem. This result has several implications.

1) **Min-max SA as a permutation problem:** To find an optimal spectrum assignment, it is sufficient to examine the connection permutations, and hence min-max SA is transformed into a permutation problem; accordingly, we have developed an optimal, recursive, branch-and-bound algorithm, recursive first-fit (RFF), to search the permutation space efficiently [18,27].
2) **Symmetry-free spectrum assignment:** In selecting among the various connection permutations there is no need to consider a spectrum assignment other than the one produced by the FF algorithm; hence, symmetric solutions are automatically eliminated from consideration and the size of the solution space is reduced by orders of magnitude [18].
3) **Inherent parallelism:** The connection permutation space is naturally represented as a tree that may be decomposed into non-overlapping subtrees [18,27]; accordingly, multi-

threaded implementations of RFF may explore the subtrees in parallel.

4) **FF is a universal algorithm:** The results of [18] imply that FF subsumes *any other* min-max SA algorithm; in other words, for any solution S constructed by any other SA algorithm (e.g., best-fit, most-used, etc. [20]), there exists a connection permutation P such that $FF(P)$ has an objective value equal to or better than that of S.

In the following, we extend these results to other SA problem variants.

B. Min-Max DSA Problem

The min-max distance SA (min-max DSA) problem is a variant introduced in [28] that includes the min-max SA problem of the previous section as a special case. Specifically, min-max DSA arises when it is required to allow for a guard band between blocks of spectrum slots assigned to different connections on the same link, e.g., to prevent crosstalk or reduce security threats at the optical layer. Min-max DSA constitutes a fairly general model in that the size of the guard band is not fixed but depends on each connection pair, e.g., on the number of spectrum slots, path, or other attribute of the two adjacent connections [28].

Formally, the min-max DSA problem can be defined as follows:

Definition 2.2 (min-max DSA):

The min-max SA problem of Definition 2.1 with:

Additional Input: guard band size $g_{ij} \geq 0$ for each connection pair (T_i, T_j), $i \neq j$.

Additional Constraint:

4) *Guard band:* If connections T_i and T_j are assigned adjacent slots on a link ℓ shared by their paths, their spectrum blocks must be separated by a block of unallocated (empty) slots of size at least g_{ij}.

Clearly, if $g_{ij} = 0 \forall i, j$, min-max DSA reduces to min-max SA.

1. DSA-FF Algorithm and Its Universality Property

Consider a permutation P of the C connections. Without loss of generality, assume that the connections are relabeled so that $P = \langle T_1, T_2, \ldots, T_C \rangle$. Let DSA-FF be a modified version of the FF algorithm for the min-max DSA problem. Similar to FF, DSA-FF takes as input some permutation P and assigns spectrum to each connection at a time in the order implied by P, i.e., $T_1, T_2, \ldots, T_C$. After considering connections $T_1, \ldots, T_{i-1}$, DSA-FF assigns to connection T_i a contiguous spectrum block of t_i slots starting at slot f_i such that f_i is the slot with the lowest index satisfying two conditions:

1) slots $[f_i, f_i + 1, \ldots, f_i + t_i - 1]$ are free along all links of the connection's path p_i, and
2) for each link $\ell \in p_i$, if some connection T_j, $j < i$, has been assigned a block of slots immediately below slot f_i on link ℓ, then there is a guard band of size at least g_{ij} between the two blocks, i.e., $f_i \geq (f_j + t_j - 1) + g_{ij}$.

We now prove the following DSA-FF universality property, which implies that, for any feasible solution to the min-max DSA problem produced by any algorithm, the DSA-FF algorithm may construct an equivalent or better solution, i.e., one with an equal or lower objective value.

Lemma 2.1 (universality property for min-max DSA): Let S be any feasible solution to the min-max DSA problem. There exists a permutation P of the C connections such that applying the DSA-FF algorithm in the order implied by P yields S or another feasible solution S' with an equal or lower objective value.

Proof. By construction.

Consider a feasible solution S to the min-max DSA problem with an objective value equal to SOL, and label the slots on each link as $1, 2, \ldots, SOL$. By definition, feasible solution S satisfies all four constraints of the min-max DSA problem. Let f_i denote the slot with the lowest index within the block of t_i slots allocated to each request T_i, $i = 1, \ldots, C$, under solution S.

Let P_S be the complete permutation in which the requests T_i are listed in increasing order of f_i in solution S, with ties broken arbitrarily. Consider the block of t_j contiguous spectrum slots, starting at slot f_j, allocated to some connection T_j. Let us remove this block of t_j slots from solution S. In the remaining partial solution, it is possible that there exists a block of t_j slots starting at a lower indexed slot $f'_j < f_j$ such that 1) they are available on all links of path p_j, and 2) there are free slots above and below this block such that allocating this block to T_j will not violate the guard band constraints. If so, we can allocate the lower-indexed t_j slots starting with slot f'_j to connection T_j without 1) affecting the feasibility of the solution or 2) increasing the objective value beyond SOL.

Based on this observation, we modify solution S by considering the connections one by one, in increasing order of f_i as listed in permutation P_S. For each connection T_i, we remove its block of spectrum slots that starts at slot f_i from the solution, and we allocate to it an equal block of slots starting at the lowest possible slot index f'_i in the partial solution such that none of the problem constraints is violated, keeping in mind that f'_i may be equal to f_i. This modified solution S' does not use more than SOL slots on any link since any modifications involve the allocation of lower-indexed spectrum slots to connections. At the same time, since modifications are applied to the starting solution S only if no constraints are violated, the modified solution S' is also feasible. Hence, the modified solution S' is either 1) identical to S, if it was not possible to move any spectrum blocks, or 2) a different feasible solution with an objective function equal to or lower than SOL, if the spectrum slots allocated to some connection(s) in S were moved to a lower-indexed block. Importantly, by construction the modified solution S' is such that no connection may be allocated to a spectrum block that starts at a lower-indexed slot.

Let P be the permutation in which the connections are reordered so that they are listed in increasing order of f'_i in the modified solution S', and let us apply the DSA-FF algorithm to this permutation. The algorithm allocates to each connection T_i a block of t_i contiguous slots starting at the lowest-indexed slot for which all problem constraints are satisfied. Therefore, the DSA-FF algorithm will construct the

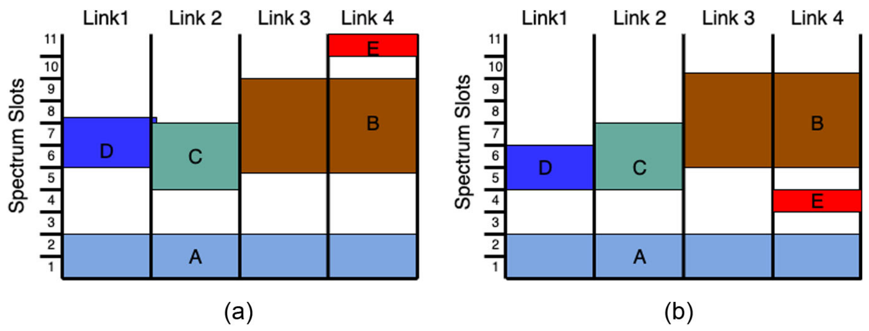

Fig. 1. (a) A feasible solution to a min-max DSA problem instance on a four-link chain with $C = 5$ connections A, B, C, D, and E. (b) The feasible (and optimal) solution constructed by the proof of Lemma 2.1.

modified solution S' above that is feasible and has an objective value no larger than SOL. ■

Figure 1 illustrates the proof of Lemma 2.1 using a simple example. Figure 1(a) shows a feasible solution to the min-max DSA problem on a four-link chain network and $C = 5$ connections labeled A, B, C, D, and E. Each connection is represented by a rectangle of a different color. The length of each rectangle spans all the links in the corresponding connection's path, whereas the width represents the number of slots allocated to the connection. For instance, the bottom-most connection A in the figure spans all four links of the network and has been allocated the contiguous block consisting of slots 1 and 2 along these links. For the problem instance shown in the figure, we have $g_{AB} = 3$, $g_{AC} = g_{AD} = 2$, and $g_{AE} = g_{BE} = 1$. Therefore, the solution in Fig. 1 is feasible as there is a guard band (i.e., block of empty slots) of appropriate size between the blocks of adjacent connections.

Figure 1(b) is the modified solution constructed by the proof of Lemma 2.1. In this solution, connection E has been moved from slot 11 to slot 4, as the three free slots between the blocks of connections A and B in Fig. 1 are sufficient to allocate one slot to connection E and form two guard bands of size 1 between A and E and between E and B. Similarly, connection D has been assigned slots 5 and 6, instead of slots 6 and 7 in the original solution. The new solution is also feasible and has an objective value of 9, lower than the objective value 11 of the original solution. The reader may also verify that 1) the solution of Fig. 1(b) is the one constructed by the DSA-FF algorithm on permutation $\langle A, B, C, D, E\rangle$, and 2) the solution of Fig. 1(a) would not have been produced by DSA-FF on any permutation of the five connections. We also note that the solution in Fig. 1(b) is optimal since the objective value is equal to the number of slots required to accommodate the amount of traffic and corresponding guard bands associated with the connections whose path includes the bottleneck link 4.

2. *Symmetry-Free Model for Min-Max DSA*

Lemma 2.1 applies to all feasible solutions, including optimal ones, yielding this corollary:

Corollary 2.1: There exists a permutation $P^\star$ of the C connections such that DSA-FF($P^\star$) is an optimal solution to the min-max DSA problem.

Consequently, the symmetry-free model we introduced in [18] for the min-max SA problem extends to this general min-max DSA variant. Specifically, min-max DSA reduces to a permutation problem in that, to find an optimal solution, one only needs to examine the solutions that the DSA-FF algorithm produces on the various connection permutations. As a result, symmetric solutions, i.e., solutions derived from a DSA-FF solution by permuting blocks of spectrum slots, are eliminated from consideration. Clearly, the min-max DSA problem remains NP-hard [28]. However, as we explain in [18], the number of symmetric solutions is exponential. Therefore, the size of the symmetry-free solution (i.e., permutation) space is orders of magnitude smaller compared to that explored by conventional ILP formulations [19,29].

Furthermore, with the elimination of symmetric solutions, the permutation space has a well-defined structure that is amenable to recursive and multi-threaded exploration. In particular, the recursive branch-and-bound RFF algorithm we developed in [18,27] for min-max SA can be readily extended to tackle the min-max DSA problem. This DSA-RFF algorithm operates identically to RFF; the only difference is that it applies DSA-FF, rather than the pure FF algorithm, as it recursively and incrementally builds and evaluates the connection permutations. Rather than repeating the details of RFF, we refer the reader to [18] for the operation of the algorithm and to [27] for two alternative multi-threaded implementations. We also note that min-max DSA includes more constraints than min-max SA; hence its solution space is smaller (since permutations that might yield feasible solutions for min-max SA may lead to infeasible solutions for min-max DSA). Therefore, we expect DSA-RFF to be more time-efficient than pure RFF. Nevertheless, an experimental evaluation of DSA-RFF is outside the scope of this paper and will be the subject of future research.

C. Min-Frag SA Problem

A spectrum *fragment* on a link is a block of one or more contiguous unused (free) slots located *between* two assigned spectrum blocks. The min-max objective we have discussed so far attempts to pack the assigned spectrum blocks tightly within lower-index slots so as to minimize spectrum fragmentation and allow for growth in demand. A different way for achieving the same goal would be to construct solutions that minimize the number of unused slots contained within spectrum fragments. Therefore, we define the min-frag SA problem as follows:

Definition 2.3 (min-frag SA):

The min-max SA problem of Definition 2.1 with this new objective:

Objective: minimize the sum of spectrum fragment sizes over all links of the network.

We now show that the optimal solutions to the min-max SA and min-frag SA problems can be very different.

Lemma 2.2: an optimal solution to the min-max SA problem is not necessarily optimal for the min-frag SA problem, and vice versa.

Proof. By counter-example.

Figure 2(a) shows an optimal solution to a min-max SA problem instance on a four-link chain with $C = 9$ connections; each connection is represented similarly to Fig. 1. The solution is optimal since the index of the highest slot used is 13, which is equal to the lower bound, i.e., the number of slots necessary to carry the traffic on connections using link 3. This is also a feasible solution to the min-frag SA problem with an objective value of 2 representing the sum of the two one-slot spectrum fragments on links 1 and 4.

Figure 2(b), on the other hand, shows an optimal solution to the min-frag SA problem instance where there is a single spectrum fragment of one slot. This solution is also a feasible solution for the min-max objective, but the index of the highest slot used is 14, higher than that of the optimal min-max solution in Fig. 2(a).

Hence, the min-max optimal solution of Fig. 2(a) is suboptimal for the min-frag objective, whereas the min-frag optimal solution of Fig. 2(b) is suboptimal under the min-max objective. ■

As we can see from Fig. 2, there is a tradeoff between the min-max and min-frag objectives. Specifically, to minimize the index of the highest slot used, it may be necessary to introduce additional spectrum fragments as in Fig. 2(a), and conversely, minimizing the spectrum fragments may require the use of higher-indexed slots as in Fig. 2(b).

Nevertheless, the min-max SA and min-frag SA problems have something in common, namely, that they are both solved by the FF algorithm. We showed that FF solves the min-max SA problem in [18]; to show that FF also solves the min-frag SA problem we first prove the following more general result.

Lemma 2.3 (FF universality property for min-frag SA): Let S be a feasible solution to the min-frag SA problem. There exists a permutation P of the C connections such that applying the FF algorithm in the order implied by P yields S or another feasible solution S' with an equal or lower objective value.

Proof. By construction.

The construction is very similar to that of the proof of Lemma 2.1 in that we consider the connections in solution S one at a time and in the same order, remove their assigned spectrum block from the solution, and attempt to place the block at a lower-indexed starting slot. The main difference is that there are no guard bands to consider when placing a spectrum block.

Note also that, if a spectrum block is moved to a lower-indexed starting slot, the objective function cannot increase (but may decrease). To see this, consider a spectrum block of size t that is moved to lower-indexed slots. There are two cases to consider: (a) links in which another spectrum block is assigned slots of higher index than this block, and (b) links for which this spectrum block is the one with the highest-index slots. In case (a), removing the block creates a spectrum fragment of size t, but placing the block at a lower-indexed slot removes a fragment of the same size; therefore, the net change in the objective function is 0. Case (b) has two sub-cases that correspond to the spectrum blocks of connections D and E in Fig. 1(a), ignoring the guard bands implicit in that figure. In both sub-cases, when the spectrum blocks of the two connections are moved to a lower-indexed starting slot, the objective function (i.e., the sum of fragment sizes) decreases. ■

Similar to Corollary 2.1, we have this result:

Corollary 2.2: There exists a permutation $P^\star$ of the C connections such that $FF(P^\star)$ is an optimal solution to the min-frag SA problem.

Therefore, the symmetry-free model, along with all the implications we discussed earlier, also extends to the min-frag SA problem. In particular, the RFF algorithm [18] may be modified in a straightforward manner to explore the permutation space for an optimal solution under the min-frag, rather

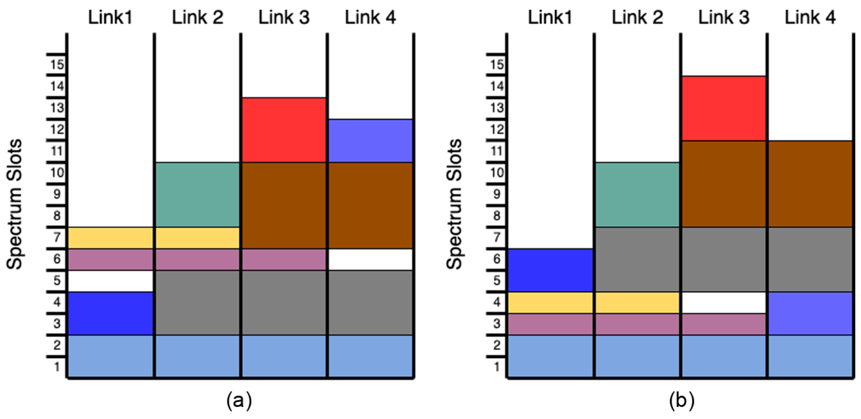

Fig. 2. (a) An optimal solution to a min-max SA problem instance on a four-link chain with $C = 9$ connections; also a feasible (but suboptimal) solution to the corresponding min-frag SA problem instance. (b) An optimal solution to the min-frag SA problem instance; also a feasible (but suboptimal) solution to the corresponding min-max SA problem instance.

the min-max, objective; multi-threaded implementations of RFF [27] are also applicable in this case.

However, there is a crucial difference between the min-max and min-frag objectives. Specifically, as the FF algorithm operates on a certain permutation to build a solution under the min-max objective, the objective value (i.e., the index of the highest assigned slot) is monotonically non-decreasing as a function of the number of connections considered. This property allows RFF to determine if a permutation that has been partially considered will not lead to a solution that is better than a baseline one; if so, RFF eliminates the current subtree of the permutation space and backtracks to explore a different subtree. The min-frag objective, on the other hand, does not have the same property: a connection further down in a permutation may fit within an existing spectrum fragment, thus *reducing* the objective value. Consequently, RFF will have to operate on each and every connection of a permutation without the possibility of backtracking; hence, it will take more time to explore the same fraction of the permutation space than for the min-max objective. As we mentioned in the previous section, an experimental investigation of this modified version of RFF is outside the scope of this paper, but we are considering other options for speeding up the exploration of the min-frag solution space.

D. Min-Frag DSA Problem

Recall that the min-frag objective attempts to minimize the number of unused slots stranded in fragments between assigned spectrum blocks. In the presence of guard band constraints, however, certain slots within a spectrum fragment may represent a required guard band and cannot be considered as "unused." Consider, for instance, the three-slot fragment in link 1 of Fig. 1(a). Since $g_{AD} = 2$, two slots of that fragment represent the required guard band between connections A and D; hence only one slot is unused. Therefore, once connection D is moved to slots 5 and 6 in the modified solution of Fig. 1(b), the two-fragment slot on link 1 represents the guard band and does not include any unused slots.

With these observations, the min-frag DSA problem is derived from min-max DSA with the new objective of minimizing the size of the spectrum fragments that do not represent guard bands. More formally, the following:

Definition 2.4 (min-frag DSA):

The min-max DSA problem of Definition 2.2 with this new objective:

Objective: minimize, over all links of the network, the sum of spectrum fragment sizes *minus* the sum of all guard bands.

Using a construction proof similar to the ones we presented in the previous two sections, we can show that DSA-FF is also a universal algorithm for the min-frag DSA problem; specifically, we have the following lemma:

Lemma 2.4 (universality property for min-frag DSA): Let S be any feasible solution to the min-frag DSA problem. There exists a permutation P of the C connections such that applying the DSA-FF algorithm in the order implied by P yields S or another feasible solution S' with an equal or lower objective value.

Similar to our earlier observations, the implication of Lemma 2.4 is that the recursive DSA-RFF algorithm we discussed in Section 2.B.2 may be used to search the symmetry-free solution (permutation) space for a permutation on which the DSA-FF algorithm produces an optimal solution to min-frag DSA. As we discussed in the previous section, the min-frag objective is such that the objective value is not monotonically non-decreasing as a function of the number of connections in the permutation to which the DSA-FF algorithm has assigned spectrum. Therefore, DSA-RFF cannot operate in a branch-and-bound mode and has to completely evaluate each permutation, i.e., perform an exhaustive search of the permutation space. A quantitative evaluation of DSA-RFF for this problem is the subject of ongoing research in our group.

3. SPECTRUM SYMMETRY-FREE RSA

We now turn our attention to the RSA problem, a generalization of the SA problem in which finding a path for each connection is part of the optimization process. Each of the C connections provided as input to the RSA problem consists of a tuple $T_i = (s_i, d_i, \mathcal{P}_i, \mathcal{D}_i)$, $i = 1, \ldots, C$, where s_i and d_i are defined as in Section 2.A, $\mathcal{P}_i$ is a set of k paths $\{p_i^{(1)}, \ldots, p_i^{(k)}\}$ between nodes s_i and d_i, and $\mathcal{D}_i$ is a set of k demands $\{t_i^{(1)}, \ldots, t_i^{(k)}\}$ such that $t_i^{(l)}$ is the number of spectrum slots required to carry the traffic of the connection along path $p_i^{(l)}$. Unlike earlier research that assumes a small number C of connections, in this paper we consider connections between all node pairs in the network and let $C = N(N-1)/2$.

We assume that k is a small integer and that the k paths of a connection are pre-determined. The paths of a connection may be calculated as the k shortest paths between the particular source-destination pair or using any other desirable criteria. Further, we assume that splitting the spectrum demand of a connection over multiple paths is not allowed.

We consider the following general definition of the RSA problem; consequently, for each of the SA problems we discussed in the previous section, there is a corresponding RSA problem variant. [In this definition we only consider spectrum-specific constraints. We assume that any routing-specific constraints (e.g., in terms of path length), are applied in determining the paths passed as input to this RSA problem.]

Definition 3.1 (RSA): For each connection T_i, select one of the physical paths $p_i^{(1)}, \ldots, p_i^{(k)}$, say, path $p_i^{(l)}$, and assign $t_i^{(l)}$ spectrum slots along this path so as to 1) optimize spectrum assignment (e.g., under the min-max or min-frag criteria of Section 2) and 2) satisfy the contiguity, continuity, non-overlap, and other variant-specific constraints (e.g., guard-band constraints as in Sections 2.C and 2.D).

It is well known that the RSA problem is intractable [5,20]. Moreover, conventional algorithms for the RSA problem must unnecessarily tackle an exponential number of symmetric and equivalent solutions. We now show how our results on symmetry-free solutions to the SA problem may be extended to the more general RSA problem. This discussion provides the motivation for the new, universal algorithm we present next in Section 3.A.

An RSA solution (optimal or not) determines one of the k paths for each connection. Let us define a *routing configuration* R_j, as an assignment of one path to each connection, whereby the path assigned to connection T_i is selected among the set $\mathcal{P}_i$ of k paths input to the RSA problem. Then, an RSA solution encompasses selecting one of the k^C routing configurations.

Consider a specific routing configuration R_j. When the path of each connection is fixed to the one specified in R_j (i.e., routing is not subject to optimization), the RSA problem reduces to the (SA problem. While simpler, the SA problem is also intractable in general topologies [14].

The symmetry-free model for the SA problem we introduced in Section 2 implies that to solve optimally *any* SA problem variant, it is sufficient to examine only the solutions obtained by applying the FF algorithm to each of the $C!$ permutations of the C input connections. (As we explained in Section 2, the FF algorithm must be appropriately modified to account for any variant-specific constraints beyond the contiguity, continuity, and non-overlap constraints, but such modifications are straightforward.) In recent work [18,27] we also developed RFF, an optimal, parallelizable branch-and-bound algorithm that recursively searches the permutation space and applies the (pure or modified) FF algorithm incrementally as it builds the various permutations. While the size of the solution space (i.e., the number of permutations) is exponential, to the best of our knowledge, our approach is the first that completely eliminates from consideration the exponential number of *additional* symmetric solutions (i.e., solutions that cannot be the result of the FF algorithm).

We now make the observation that the various routing configurations of the RSA problem are pairwise independent. Therefore, our work in [18,27] can be naturally extended to a spectrum symmetry-free algorithm for the RSA problem:

Run the RFF algorithm on all routing configurations and return the routing configuration and connection permutation that results in the best solution.

In essence, the solution space of the RSA problem is optimally decomposed into a routing space of k^C routing configurations and a permutation space of $C!$ permutations that must be explored separately for each routing configuration. While the above algorithm completely sidesteps all symmetric solutions, it amounts to an exhaustive search of a combined solution space of size $k^C \times C!$. Assuming that there is a connection between each node pair in the network, then $C = O(N^2)$ and both the number of routing configurations and the number of connection permutations are exponential in the size of the network. Therefore, it is prohibitive to search exhaustively the combined solution space for network topologies encountered in practice.

In the following, we present a solution approach that performs an exhaustive search only on a part of the combined solution space whose size can be calibrated appropriately. Importantly, the algorithm may be applied directly to all RSA variants with only minor modifications to the underlying FF algorithm, depending on the variant-specific spectrum constraints.

A. Universal Symmetry-Free RSA Algorithm

Our goal is to bridge the gap between an exhaustive search of the entire solution space, which is prohibitively expensive computationally for deployed networks, and greedy heuristic approaches by introducing a parameterized approach that achieves a desirable tradeoff between running time and quality of solution. Our universal algorithm for RSA problem variants is characterized by two parameters:

1) the number $c < C$ of high-priority connections, and
2) the number k of paths for each high-priority connection.

Network designers may apply any appropriate criteria to decide which connections are included in the high-priority set. For instance, connections may be characterized as high priority based on 1) the size of their demands, 2) the distance between their endpoints, 3) a measure of the importance of the traffic they carry, 4) the revenue they produce, or 5) any combination thereof. Each high-priority connection is provided with $k > 1$ alternate paths; each low-priority connection, on the other hand, may use only a single (fixed) path. Any appropriate routing algorithm and link weights may be used to generate these paths.

Consider an RSA problem with C connections and define the *high-priority subproblem* as one that includes only the $c < C$ high-priority connections (i.e., the subproblem created by eliminating the $C - c$ low-priority connections from the original problem). The solution space for this high-priority subproblem is the combination of the routing configurations (of size k^c) and permutations (of size $c!$) for the c high-priority connections. Since c and k are small integers selected by the network designer, the size of this solution space can be considered as fixed and can be carefully adjusted to match the available computational resources.

Our approach to tackling the original RSA problem with C connections is to explore exhaustively a subset of its solution space that encompasses the entire solution space of the high-priority subproblem. This strategy collectively optimizes the allocation of resources to the connections that the network designer regards as important. For instance, let us assume that priority is proportional to the demand size and/or distance of a connection. Intuitively, connections with large demands or that travel long distances require correspondingly large resources. Hence, their path and spectrum must be optimized, not only individually but in combination with other such connections, to ensure that network resources are allocated efficiently. Furthermore, a small fraction of all connections may account for a considerable fraction of total demand (refer to Section 3.B), so that exploring the entire solution space of such connections may be computationally feasible.

Algorithm 1 provides a pseudo-code description of the symmetry-free RSA algorithm with parameters (k, c). The preprocessing step of the algorithm constructs the solution space to explore. Specifically, Steps 1–5 generate all $c!$ permutations of high-priority connections and extend each (at Step 4) by appending the $C - c$ low-priority connections in a fixed order to create $c!$ permutations of all C connections. Similarly, Steps 6–10 generate all k^c routing configurations of high-priority connections and extend each (at Step 9) with

Algorithm 1. Universal Spectrum Symmetry-Free RSA

```
Input:
    $G = (V, A)$: network topology
    $C$: number of connections
    $c$: number of high-priority connections
    $k$: number of paths for each high-priority connection
    $\mathcal{T} = \{T_i = (s_i, d_i, \mathcal{P}_i, \mathcal{D}_i)\}$: set of connections
Output:
    BestSOL: RSA solution
SymFree-RSA($k$, $c$)
  Preprocessing
1: $q$: List of $C - c$ low-priority connections in decreasing priority;
2: Generate all $c!$ permutations $q_l$ of the $c$ high-priority connections;
3: for $l = 1$; $l \le c!$; $l++$ do
4:     $Q_l \leftarrow$ Append $q$ to $q_l$ (permutation of all $C$ connections);
5: end for
6: $r$: routing configuration with single path for the $C - c$
   low-priority connections;
7: Generate all $k^c$ routing configurations $r_j$ for the $c$ high-priority
   connections;
8: for $j = 1$; $j \le k^c$; $j++$ do
9:     $R_j \leftarrow$ Append $r$ to $r_j$ (routing configuration for all $C$
       connections);
10: end for
    Main Algorithm
11: for $j = 1$; $j \le k^c$; $j++$ do
12:     for $l = 1$; $l \le c!$; $l++$ do
13:         SOL $\leftarrow$ solution obtained by FF on routing configuration
            $R_j$ and permutation $Q_l$
14:         if SOL < BestSOL then
15:             BestSOL = SOL;
16:         end if
17:     end for
18: end for
19: return BestSOL;
```

the single path of each low-priority connection to create a routing configuration for all C connections. Finally, the main algorithm in Steps 11–19 exhaustively searches this solution space by applying the FF algorithm to each combination of routing configuration/permutation. Since each such combination is independent of any other, the execution of Steps 11–15 may be easily parallelized by 1) partitioning the combinations into pair-wise disjoint and collectively exhaustive subsets and 2) deploying multiple threads running concurrently, each thread working on a different subset.

The preprocessing step takes time $O(c! + k^c)$, but since c and k have small integer values determined by the network designer, this step can be considered as taking a fixed amount of time. Note that a network designer may have to solve multiple instances for a given RSA problem defined by the network topology $G = (V, A)$ and number of connections C; 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 permutations and routing configurations, 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 11–15 simply runs the FF algorithm on each of the $M = k^c \times c!$ combinations of permutations and routing configurations generated in the preprocessing step. Each application of the FF algorithm takes time $O(CL)$, as each permutation consists of C connections and each connection may involve any of the L links in the network. Therefore, the total running time of this part of the algorithm is $O(CLM)$, where M is again considered as having a fixed value.

Note that parameter k (respectively, c) represents the degree of path (respectively, connection) diversity. Each parameter independently controls the size of the solution space of the routing subproblem and spectrum allocation subproblem, respectively; hence the family of algorithms represents a wide spectrum of RSA solution strategies. Specifically, 1) when $c = 0$, the algorithm reduces to FF as it considers a single path for all connections and one permutation; 2) when $c = C$ and $k = 1$, it reduces to the RFF algorithm [18], and, given sufficient time to run, it explores all connection permutations on a single routing configuration; and 3) when $c = C$ and $k > 1$, it is an extension to RFF that, given sufficient time to run, is optimal for the given set of routing paths. By carefully selecting values for the two parameters k and c, a network designer may strike a desirable balance between the running time and the quality of the final solution.

Finally, we emphasize again that Algorithm 1 is applicable to any variant of the RSA problem, not just the basic variant of Definition 3.1. For instance, the k paths may be calculated so as to take into account reach, various available modulation formats [30], intra- or inter-core crosstalk [13], etc. Additional constraints may eliminate some of the routing configurations, reducing the effective size of the routing space well below k^c and, thus, allowing for larger values for parameters k and/or c. Due to page limitations, however, the study in the next section only focuses on the basic RSA problem with just the contiguity, continuity, and non-overlap constraints.

B. Simulation Study of Min-Max RSA

Recall from Section 3.A that, when $c = C$ and $k > 1$, Algorithm 1 is optimal for the given set of paths as it examines all combinations of routing configurations and connection permutations. In practice, however, it would not be possible to search the entire solution space for anything but toy networks. Importantly, we expect that, as the values of parameters c and k increase, the incremental improvement in solution quality will drop off due to the diminishing returns of considering low-priority (e.g., small) demands and long, circuitous paths. Therefore, our objective is to investigate the relative benefits of increasing path diversity (i.e., value of k) and connection diversity (i.e., value of c) on solution quality.

In this simulation study we consider the min-max RSA variant derived from the min-max SA problem of Section 2.A. (A simulation study of the other RSA variants is outside the scope of this paper and will be the subject of future research.) We create RSA problem instances characterized by two parameters: the network topology and the distribution used to generate random traffic demands. We use two network topologies, the 14-node, 21-link NSFNET and the larger 32-node, 54-link

GEANT2 network, and for each topology we generate connections between all node pairs in the network as follows. We consider data rates of 10, 40, 100, 400, and 1000 Gbps. For a given problem instance, we generate a random value for the demand between a pair of nodes based on one of three distributions:

- *uniform*: each rate is selected with equal probability;
- *skewed low*: the rates above are selected with probability 0.30, 0.25, 0.20, 0.15, and 0.10, respectively; or
- *skewed high*: the five rates are selected with probability 0.10, 0.15, 0.20, 0.25, and 0.30, respectively.

Then, we determine the number of spectrum slots that each demand requires based on its data rate and path length by assuming a slot width of 12.5 GHz and adopting the parameters of [3]. For each topology and traffic distribution we generate 100 random problem instances, for a total of 600 instances for this study.

We run the experiments on the Henry2 Linux HPC cluster at NC State University [31], which consists of more than 1000 compute nodes and over 10,000 cores. We deployed $R = 32$ parallel threads, the maximum number available to us on the Henry2 cluster, to run the main Steps 11–15 of Algorithm 1. In the experiments, we vary the number of paths, $k = 2, \ldots, 7$, and the number of high-priority connections, $c = 1, 2, \ldots, 7$. We also impose a running time limit of 1000 s for each problem instance; hence, as shown in the figures below, any combination of (k, c) values for which the running time would exceed this time limit is not considered. For Algorithm 1, we select the c high-priority connections as those with the largest demands, and we use the depth first search (DFS) algorithm to calculate the k shortest paths for the high-priority connections.

The performance measure we consider is the maximum number of spectrum slots on any network link. For a given routing configuration R_j, a lower bound for this metric for an RSA problem instance can be calculated by ignoring the problem constraints and simply adding up the demands along each link and taking the maximum value over all links. Let *SPLB* denote this lower bound under shortest path routing, i.e., for the routing configuration consisting of the shortest path for all C connections. To make the results comparable across problem instances and (k, c) values, we normalize each solution *SOL* returned by the algorithm by taking the ratio

$$h = \frac{SOL - SPLB}{SPLB}. \tag{1}$$

The figures in this section plot this ratio, which represents the normalized difference between *SOL* and *SPLB*. Each data point in the figures is the average of h over the 100 problem instances for the stated topology and traffic distribution. However, we emphasize that *SPLB does not represent a lower bound for RSA algorithms that use two or more paths for some connections.* As we will see in a moment, with increasing path diversity the algorithm finds solutions that are *better* (i.e., lower) than the shortest path lower bound SPLB; in such cases, the value of h is negative. Nevertheless, we use the SPLB value for normalization because 1) it is a well-understood baseline quantity, and 2) it provides insight into the improvement that is possible with increasing path diversity.

1. Results and Discussion

Figure 3 plots the values of h (as a percentage) for the NSFNET topology and the skewed low distribution as a function of the number c of high-priority connections. [Due to page constraints, we only include results for one (different) traffic distribution for each of the two networks; however, the trends for the other two distributions are very similar to the ones in the figures we present here.] There are six curves in the figure, each corresponding to a number k, $k = 2, \ldots, 7$, of alternate paths for the c high-priority connections (the $C - c$ low-priority connections are routed along their minimum-hop path). We observe that the solution quality of all curves improves significantly with c; for instance, with $k = 2$ paths, the solutions obtained by the algorithm improve, on average, by 16% relative to SPLB, i.e., from a value about 6.5% *above* SPLB to a value about 9.5% *below* SPLB. As expected, the curves start to level off after a while, depending on k, but it appears that there is room for further improvement as c increases beyond 7 (recall that we did not run experiments with $c = 8$ or higher as the running time would exceed our time budget of 1000 s). We also observe that there is an increase in quality across all values of c as we move from $k = 2$ to $k = 3$ paths, but further increases in the number of alternate paths have little benefit.

Figure 4 presents the same results as Fig. 3 but plots them instead as a function of k. This figure more clearly shows that 1) most benefits of path diversity are realized as soon as the number k of paths is 3 or 4, and further increases have little impact, and 2) the marginal gain in solution quality from incrementing connection diversity (i.e., c) is considerably higher than that from incrementing path diversity (i.e., k); even so, the effect of diminishing returns as c increases is also clear.

Figures 5 and 6 are similar to Figs. 3 and 4, respectively, but present results for the GEANT2 network and the skewed high distribution. We observe similar trends as for the NSFNET instances in terms of the relative benefits of path and connection diversity and their diminishing returns. One difference between the two sets of results is that the percent improvement in solution quality is smaller for the GEANT topology, which we explain shortly. Nevertheless, even a small improvement for

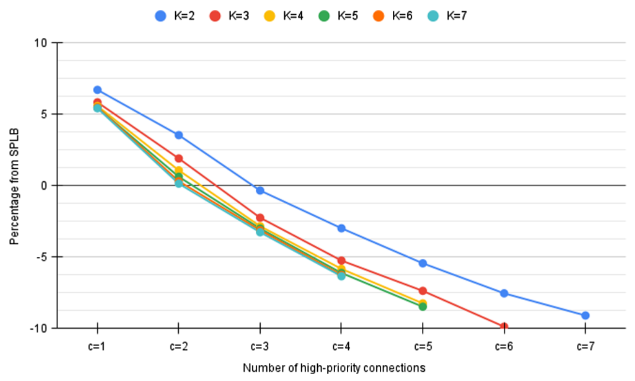

Fig. 3. Solution quality as a function of c, NSFNET, skewed low distribution.

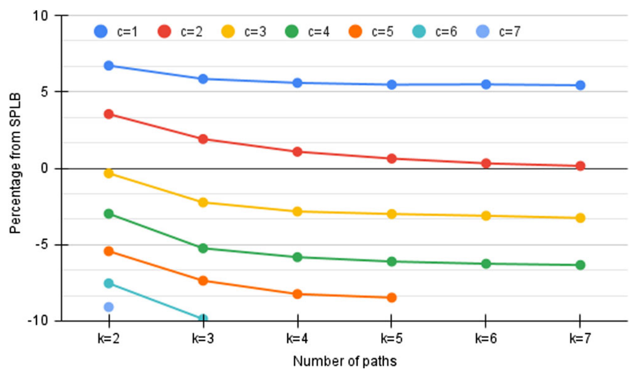

Fig. 4. Solution quality as a function of k, NSFNET, skewed low distribution.

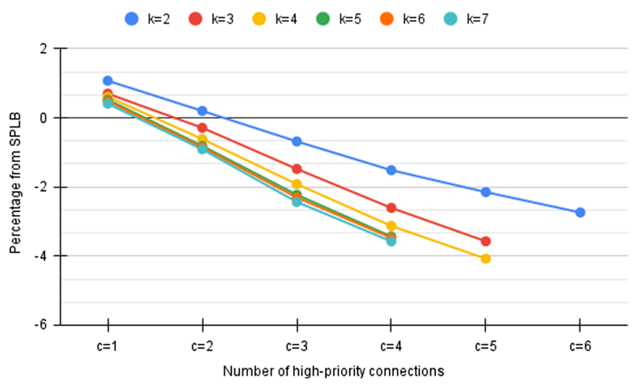

Fig. 5. Solution quality as a function of c, GEANT2, skewed high distribution.

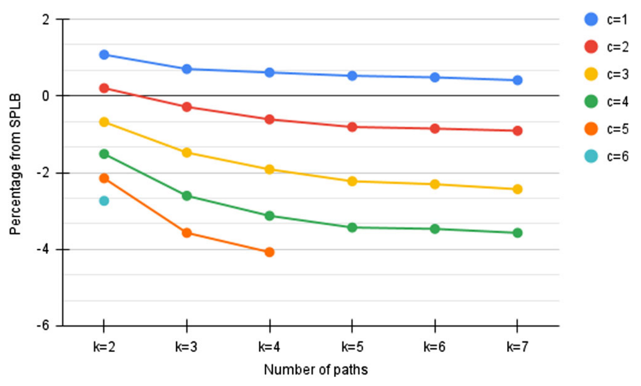

Fig. 6. Solution quality as a function of k, GEANT2, skewed high distribution.

the larger GEANT2 network (which represents a much larger total demand) translates into significant savings of network resources, especially since these spectrum savings apply to a larger number of links.

To put the results of Figs. 3–6 into perspective, Figs. 7 and 8 plot the percentage of total demand that the c high-priority connections represent, for the two networks and three traffic distributions we considered in this study. Recall that the average demand size is smallest (respectively, largest) for the skewed low (respectively, high) distribution, with the demand size of the uniform distribution falling in between these

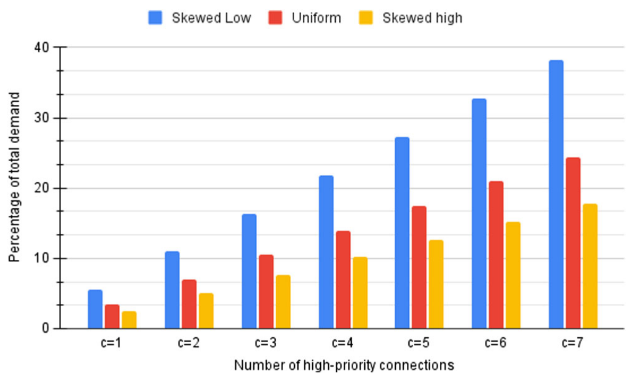

Fig. 7. Fraction of total demand for high-priority connections, NSFNET.

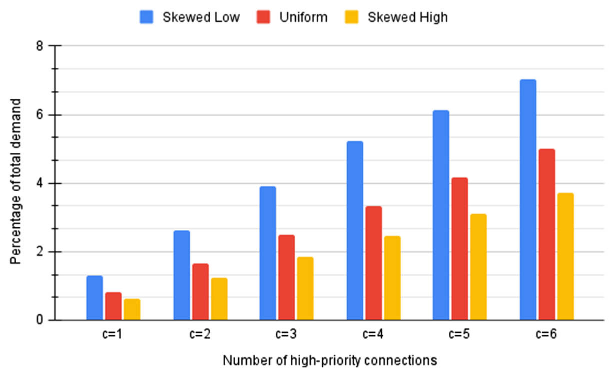

Fig. 8. Fraction of total demand for high-priority connections, GEANT2.

two. Therefore, for a given value of c, the percentage of total demand represented by the c high-priority (i.e., largest in our study) connections is smallest for the skewed high distribution, followed by the uniform and skewed low distribution; this is consistent with the results shown in the two figures. However, there are 91 $(= 14 \times 13/2)$ connections for the NSFNET but 496 $(= 32 \times 31/2)$ connections for the larger GEANT2 network. Therefore, as we observe in the two figures, c connections make up a smaller fraction of total demand for the GEANT2 network relative to NSFNET. Consequently, optimizing for c connections is expected to have a smaller relative improvement in solution quality for GEANT2 than NSFNET, consistent with the results we presented earlier.

Overall, we can draw several general conclusions regarding the performance of our RSA algorithm across the network topologies and traffic distributions used in our study: 1) solution quality improves with both path and connection diversity; 2) connection diversity has relatively higher benefits than path diversity; 3) the gains of path diversity level off after $k = 4$, consistent with earlier studies of alternate path routing; and 4) exhaustively exploring the solution space of a few high-priority connections pays off significantly even when these connections represent a relatively small fraction of total demand.

4. CONCLUDING REMARKS

We have studied two variants of the spectrum assignment problem, one without guard band constraints (i.e., SA) and one with such constraints (i.e., DSA). For each variant, we considered two objectives: min-max, which minimizes the highest spectrum slot index assigned to any connection, and min-frag, which minimizes the sum of the number of unused slots in spectrum fragments across all links. We have shown that the FF algorithm may be used to solve optimally the min-max SA and min-frag SA problems. Likewise, the DSA-FF algorithm, which operates similarly to FF but takes into consideration the guard band constraints, may be used to solve optimally the min-max DSA and min-frag DSA problems. We also showed that while the two objectives, min-max and min-frag, attempt to minimize fragmentation, they may lead to different solutions.

Our results transform all four spectrum assignment problems into permutation problems with a well-structured and symmetry-free solution space and show that seemingly disparate variants of spectrum assignment have a common underlying structure and may be tackled using similar approaches. Building upon this new insight, we developed the first spectrum symmetry-free algorithm that can be applied to all variants of the RSA problem. Our method explores the whole solution space of a subset of connections, and our simulation results indicate that connection diversity is more beneficial than path diversity. Our group is working on scaling this approach in terms of the size of solution space that can be explored. Our work represents the first step towards new algorithmic approaches to optical network design.

Funding. Directorate for Computer and Information Science and Engineering (CNS-1907142).

Acknowledgment. The author would like to thank Mr. Mithil Ghinaiya who carried out the simulation experiments in Section 3.B when he was a Masters student at North Carolina State University. Portions of this work were presented at IEEE GLOBECOM in 2023 as “First-fit: a universal algorithm for spectrum assignment” [32] and at IEEE ICC 2024 as “A universal spectrum symmetry-free algorithm for routing and spectrum assignment (RSA)” [33].

REFERENCES

1. J. M. Simmons, *Optical Network Design and Planning* (Springer, 2008).
2. B. Jaumard, C. Meyer, and B. Thiongane, “Comparison of ILP formulations for the RWA problem,” Opt. Switch. Netw. **4**, 157–172 (2007).
3. M. Jinno, B. Kozicki, H. Takara, *et al.*, “Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network,” IEEE Commun. Mag. **48**(8), 138–145 (2010).
4. K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, “Elastic bandwidth allocation in flexible OFDM–based optical networks,” J. Lightwave Technol. **29**, 1354–1366 (2011).
5. M. Klinkowski, P. Lechowicz, and K. Walkowiak, “Survey of resource allocation schemes and algorithms in spectrally-spatially flexible optical networking,” Opt. Switch. Netw. **27**, 58–78 (2018).
6. S. Talebi, F. Alam, I. Katib, *et al.*, “Spectrum management techniques for elastic optical networks: a survey,” Opt. Switch. Netw. **13**, 34–48 (2014).
7. R. Dutta and G. N. Rouskas, “A survey of virtual topology design algorithms for wavelength routed optical networks,” Opt. Netw. **1**, 73–89 (2000).
8. H. Wang and G. N. Rouskas, “Hierarchical traffic grooming: a tutorial,” Comput. Netw. **69**, 147–156 (2014).
9. R. Dutta and G. N. Rouskas, “Traffic grooming in WDM networks: past and future,” IEEE Netw. **16**, 46–56 (2002).
10. B. Chen, G. N. Rouskas, and R. Dutta, “Clustering methods for hierarchical traffic grooming in large-scale mesh WDM networks,” J. Opt. Commun. Netw. **2**, 502–514 (2010).
11. C. Castillo, G. N. Rouskas, and K. Harfoush, “Efficient resource management using advance reservations for heterogeneous grids,” in *22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS 2008)*, April 2008.
12. Y. Xiong, Y. Li, B. Zhou, *et al.*, “SDN enabled restoration with triggered precomputation in elastic optical inter-datacenter networks,” J. Opt. Commun. Netw. **10**, 24–34 (2018).
13. F. Tang, G. Shen, and G. N. Rouskas, “Crosstalk-aware shared backup path protection in multi-core fiber elastic optical networks,” J. Lightwave Technol. **39**, 3025–3036 (2021).
14. S. Talebi, E. Bampis, G. Lucarelli, *et al.*, “Spectrum assignment in optical networks: a multiprocessor scheduling perspective,” J. Opt. Commun. Netw. **6**, 754–763 (2014).
15. F. Bertero, M. Bianchetti, and J. Marenco, “Integer programming models for the routing and spectrum allocation problem,” Trans. Oper. Res. **26**, 465–488 (2018).
16. L. Velasco, M. Ruiz, and M. Klinkowski, *The Routing and Spectrum Allocation Problem* (Wiley, 2017), Chap. 3, pp. 43–60.
17. J. Wang, M. Shigeno, and Q. Wu, “ILP models and improved methods for the problem of routing and spectrum allocation,” Opt. Switch. Netw. **45**, 100675 (2022).
18. G. N. Rouskas and C. Bandikatla, “Recursive first fit: a highly parallel optimal solution to spectrum allocation,” J. Opt. Commun. Netw. **14**, 165–176 (2022).
19. B. Jaumard, C. Meyer, and B. Thiongane, “ILP formulations for the routing and wavelength assignment problem: symmetric systems,” in *Handbook of Optimization in Telecommunications* (Springer, 2006), pp. 637–677.
20. J. Simmons and G. N. Rouskas, “Routing and wavelength (spectrum) allocation,” in *Springer Handbook of Optical Networks*, B. Mukherjee, ed. (Springer, 2020).
21. H. Zang, J. P. Jue, and B. Mukherjee, “A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks,” Opt. Netw. **1**, 47–60 (2000).
22. Y. Zhu, G. N. Rouskas, and H. G. Perros, “A comparison of allocation policies in wavelength routing networks,” Photon. Netw. Commun. **2**, 267–293 (2000).
23. D. S. Johnson, A. J. Demers, J. D. Ullman, *et al.*, “Worst-case performance bounds for simple one-dimensional packing algorithm,” SIAM J. Comput. **3**, 235–299 (1974).
24. M. R. Garey, R. L. Graham, and J. D. Ullman, “Worst-case analysis of memory allocation algorithms,” in *4th Annual ACM Symposium on Theory of Computing* (1972), pp. 143–150.
25. M. R. Garey, R. L. Graham, D. S. Yao, *et al.*, “Resource constrained scheduling as generalized bin packing,” J. Comb. Theory Ser. A **21**, 257–298 (1976).
26. G. Dosa and J. Sgall, “First fit bin packing: a tight analysis,” in *30th International Symposium on Theoretical Aspects of Computer Science (STACS)* (2013), pp. 538–549.
27. G. N. Rouskas, S. Gupta, and P. Sharma, “Experimental evaluation of a symmetry-free parallel algorithm for spectrum allocation,” in *IEEE GLOBECOM*, December 2022.
28. H. Wu, F. Zhou, Z. Zhu, *et al.*, “On the distance spectrum assignment in elastic optical networks,” IEEE/ACM Trans. Netw. **25**, 2391–2404 (2017).
29. R. Ramaswami and K. Sivarajan, “Routing and wavelength assignment in all-optical networks,” IEEE/ACM Trans. Netw. **3**, 489–500 (1995).
30. S. Talebi and G. N. Rouskas, “On distance-adaptive routing and spectrum assignment in mesh elastic optical networks,” J. Opt. Commun. Netw. **9**, 456–465 (2017).

31. NC State, “Henry2 Linux Cluster,” https://projects.ncsu.edu/hpc/About/ComputeResources.php.
32. G. N. Rouskas, “First-fit: a universal algorithm for spectrum allocation,” in *IEEE GLOBECOM 2023*, December 2023.
33. G. N. Rouskas and M. Ghinaiya, “A universal spectrum symmetry-free algorithm for routing and spectrum assignment (RSA),” in *IEEE International Conference on Communications (ICC)*, June 2024.