# On Distance-Adaptive Routing and Spectrum Assignment in Mesh Elastic Optical Networks

Sahar Talebi and George N. Rouskas

***Abstract*—The routing and spectrum assignment (RSA) problem has emerged as the key design and control problem in elastic optical networks. Distance-adaptive spectrum allocation exploits the tradeoff between spectrum width and reach to improve resource utilization by tailoring the modulation format to the level of impairments along the path. In this paper, we consider the distance-adaptive RSA (DA-RSA) problem with fixed alternate routing. We first show that the DA-RSA problem in networks of general topology is a special case of a well-studied multiprocessor scheduling problem. We then leverage insights from the scheduling theory to 1) present new results regarding the complexity of the DA-RSA problem and 2) build upon the list of scheduling concepts to develop a computationally efficient solution approach that is effective in utilizing the available spectrum resources.**

***Index Terms*—Distance-adaptive routing and spectrum assignment; Elastic optical networks; Multiprocessor scheduling; Network design; Spectrum assignment.**

## I. Introduction

Optical networking technologies underlie the delivery and availability of reliable and survivable Internet services. As optical transmission speeds approach 1 Tbps, new technologies, including flexible spectrum switches and bandwidth-variable transceivers [1], continue to drive novel capabilities at the optical layer. Elastic optical networks [2,3] offer the promise of harnessing the properties of *individual* optical devices to deliver novel features and capabilities at the *network* scale. Furthermore, using a finer spectrum granularity than conventional fixed-grid WDM technology, elastic networks have the potential to efficiently accommodate the ever-increasing traffic demands by tailoring both the modulation format and spectrum resources to the data rate and path impairments [2,4].

Elastic optical network technology is in the early stages of development and/or deployment, yet relevant network design techniques have been the subject of considerable research and development activities in recent years. The routing and spectrum assignment (RSA) problem [5] addresses the network-wide allocation and management of spectral resources and is fundamental to the design and control of elastic optical networks. The objective of RSA is to assign a spectrum and a physical path to each demand, so as to optimize spectrum utilization. Several aspects of the problem have been studied in the literature, including offline RSA [6,7], online RSA [8,9], distance-adaptive RSA (DA-RSA) [10,11], fragmentation-aware RSA (FA-RSA) [12], RSA and traffic grooming [13], and RSA with restoration [14]; for a recent survey of the literature, we refer the reader to [5]. Most existing studies approach the problem using classical network design techniques. Network design problems are notoriously hard, and optimal methods (e.g., integer programming formulations) do not scale to the topologies encountered in practice. This issue is even more pronounced in elastic optical networks, as the network designer has to take into account additional dimensions, including variable bandwidth demands (rather than single-wavelength ones, as in fixed-grid WDM) and tradeoffs in reach versus spectral efficiency.

In this paper, we provide new insights into the structure of the offline DA-RSA problem by relating it to a well-known problem of scheduling multiprocessor tasks on dedicated processors. We also present a computationally efficient solution approach for mesh (i.e., general topology) networks, based on list scheduling, that is effective in utilizing the available spectrum resources. Specifically, the remainder of the paper is organized as follows. In Section II, we define DA-RSA with fixed alternate routing and show that this problem is a special case of a multiprocessor scheduling problem in which a task may be executed by alternate sets of processors. Accordingly, we leverage the scheduling theory to investigate the complexity of the DA-RSA problem (in Section III) and to develop a list-scheduling algorithm to solve it (in Section IV). In Section V, we present the results of an experimental study to evaluate the list-scheduling algorithm on various network topologies and traffic distributions, and we conclude the paper in Section VI.

## II. DA-RSA in General Graphs As a Special Case of Multiprocessor Scheduling

The concept of distance-adaptive (DA) spectrum allocation was introduced in [15] to exploit the tradeoff between reach and spectrum width by tailoring the modulation

Manuscript received August 10, 2016; revised February 8, 2017; accepted March 10, 2017; published April 27, 2017 (Doc. ID 273341).

S. Talebi (e-mail: stalebi@ncsu.edu) and G. N. Rouskas are with Operations Research and Department of Computer Science, North Carolina State University, Raleigh, North Carolina 27695-8206, USA.

https://doi.org/10.1364/JOCN.9.000456

 

format to the level of impairments along the path so as to improve spectrum utilization. Specifically, for the same data rate, a high-level modulation format with a low SNR tolerance and narrow spectrum may be selected for a short path, whereas a low-level modulation with a high SNR tolerance and a wider spectrum may be used for a longer path [11].

We consider the following general definition of the DA-RSA problem with fixed alternate routing in elastic optical networks:

- *DA-RSA inputs:* 1) a directed graph $G = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes, and $\mathcal{A}$ is the set of arcs (directed links), 2) $k$ alternate routes, $r_{sd}^1, \ldots, r_{sd}^k$, from node $s$ to node $d$, where $k \geq 1$ is a small integer, 3) a spectrum demand matrix $T = [t_{sd}^l]$, such that i) $t_{sd}^l$ is the number of spectrum slots required to carry the traffic from source $s$ to destination $d$ along the $l$th route between the two nodes, $l = 1, \ldots, k$, and ii) spectrum demands may increase (but not decrease) with the path length, i.e.,

$$|r_{sd}^l| \leq |r_{sd}^h| \Rightarrow t_{sd}^l \leq t_{sd}^h, \tag{1}$$

  where $|r_{sd}^l|$ denotes the physical distance of the $l$th path between nodes $s$ and $d$.
- *DA-RSA objective:* select one of the $k$ possible routes for each spectrum demand and assign spectrum slots along all the arcs of this route such that the total amount of spectra used on any arc in the network is minimized.
- *DA-RSA constraints:* 1) spectrum contiguity: each demand is assigned contiguous spectrum slots; 2) spectrum continuity: each demand uses the same spectrum slots along all arcs of its route; and 3) non-overlapping spectrum: demands that share an arc are assigned non-overlapping parts of the available spectrum.

Now, consider the following multiprocessor scheduling problem $P|\text{set}_j|C_{\max}$, defined as [16]:

- $P|\text{set}_j|C_{\max}$ *inputs:* a set of $m$ identical processors; a set of $n$ tasks; a set $\text{set}_j = \{S_j^1, \ldots, S_j^k\}$ of $k$ alternative processor sets that may execute each task $j$, where $k$ is an integer; and the processing time $p_j^l$ of task $j$ when it is to be executed on processor set $S_j^l$, $l = 1, \ldots, k$.
- $P|\text{set}_j|C_{\max}$ *objective:* assign one of the $k$ sets of processors to execute each task, and schedule the tasks so as to minimize the makespan $C_{\max} = \max_j C_j$ of the schedule, where $C_j$ denotes the finish time of task $j$.
- $P|\text{set}_j|C_{\max}$ *constraints:* 1) no preemption is allowed, 2) all the processors in the selected set must work on task $j$ simultaneously, and 3) each processor may execute at most one task at any given time.

The next two lemmas show that the DA-RSA problem with fixed-alternate routing in networks of general topology is a special case of the $P|\text{set}_j|C_{\max}$ scheduling problem. Lemma 2.1 first shows that DA-RSA transforms to $P|\text{set}_j|C_{\max}$, and hence, any algorithm for the latter problem also solves the former. Lemma 2.2 shows by counter-example that the reverse result is not true, i.e., that there exist instances of $P|\text{set}_j|C_{\max}$ for which there is no corresponding instance of DA-RSA.

Lemma 2.1: DA-RSA with fixed-alternate routing in mesh networks transforms to $P|\text{set}_j|C_{\max}$.

*Proof.* Consider an instance of the DA-RSA problem with fixed-alternate routing on a general directed graph $G = (\mathcal{V}, \mathcal{A})$, a set of $k$ routes $\{r_{sd}^1, \ldots, r_{sd}^k\}$ for each source–destination pair $(s, d)$, and demand matrix $T = [t_{sd}^l]$, $l = 1, \ldots, k$. It is possible to construct an instance of $P|\text{set}_j|C_{\max}$ such that 1) there is a processor $i$ for every arc in $a_i \in \mathcal{A}$; 2) there is a task $j$ for each source–destination pair $(s, d)$; 3) there is a $\text{set}_j = \{S_j^1, \ldots, S_j^k\}$ for each task $j$ with $S_j^l = \{i{:}a_i \in \{r_{sd}^l\}\}$, where $(s, d)$ is the source–destination pair corresponding to task $j$; and 4) the processing time of task $j$ on processor set $S_j^l$ is $p_j^l = t_{sd}^l, l = 1, \ldots, k$. In this transformation, each arc in the DA-RSA problem maps to a processor in the scheduling problem, each spectrum demand to a task, each alternate route of a demand to one of the alternate processor sets of the corresponding task, and the number of spectrum slots along a route of a demand to the processing time of the task on the corresponding set of processors. Note that, because of Eq. (1), the processing times of each task $j$ in the $P|\text{set}_j|C_{\max}$ instance will obey this relationship:

$$|S_j^l| \leq |S_j^h| \Rightarrow p_j^l \leq p_j^h. \tag{2}$$

With this transformation, the spectrum contiguity constraint in allocating slots to a demand implies that processing the corresponding task in the constructed scheduling problem will continue with no preemption. The spectrum continuity constraint along the arcs of the route taken by a demand guarantees that all the processors within the set assigned to a task will execute this task simultaneously. Also, the non-overlapping spectrum constraint ensures that a processor works on one task at a time at most.

Finally, the total amount of spectra required for all the demands, using an arc of graph $G$ in the DA-RSA problem, is equivalent to the completion time of the last task executed on the corresponding processor. Accordingly, minimizing the spectrum use on any arc of the DA-RSA problem is equivalent to minimizing the makespan of the schedule in the corresponding problem $P|\text{set}_j|C_{\max}$. ∎

*Lemma 2.2:* There exist instances of $P|\text{set}_j|C_{\max}$ for which there is no corresponding instance of the DA-RSA problem with fixed-alternate routing.

*Proof.* By counter-example. Consider an instance of $P|\text{set}_j|C_{\max}$ with $m = 3$ processors labeled $P_1$, $P_2$, and $P_3$, and $n = 3$ tasks $\tau_1$, $\tau_2$, and $\tau_3$. Each task $\tau_j$ must be executed by a single set $S_j^1$ of processors (i.e., $k = 1$), as shown in the following table; the processing time of each task can be arbitrary:

| Task | $S_j^1$ |
|---|---|
| $\tau_1$ | $\{P_1, P_2\}$ |
| $\tau_2$ | $\{P_2, P_3\}$ |
| $\tau_3$ | $\{P_1, P_3\}$ |

The graph of the corresponding DA-RSA instance would have to consist of three directed links, $L_1$, $L_2$, and

$L_3$, corresponding to processors $P_1, P_2$, and $P_3$, respectively. Consider task $\tau_1$. Since this task must be executed simultaneously on processors $P_1$ and $P_2$, the corresponding demand in the DA-RSA instance must be routed along the two arcs $L_1$ and $L_2$. Suppose that the path of this demand is the directed pair of arcs $\langle L_1, L_2 \rangle$; if the path is the directed pair of arcs $\langle L_2, L_1 \rangle$, then similar arguments may be used to reach the same conclusion. Now consider task $\tau_2$. Similarly, the path of the corresponding demand can be either a) $\langle L_2, L_3 \rangle$ or b) $\langle L_3, L_2 \rangle$. In case (a), the graph of the DA-RSA instance would have to be the directed three-link chain network $\langle L_1, L_2, L_3 \rangle$. In case (b), the graph of the DA-RSA instance would have to be a three-link network in which links $L_1$ and $L_3$ feed into link $L_2$. Consequently, there is no feasible path for the spectrum demand corresponding to the third task $\tau_3$ in either graph, as it is not possible for the traffic on link $L_1$ to continue onto link $L_3$, or vice versa. Therefore, an instance of DA-RSA does not exist. ∎

## III. Complexity Results

The problem $P2|\text{set}_j|C_{\max}$, in which the number of processors is fixed to $m = 2$, is NP-hard [17]. Moreover, it has been shown that, unless $P = \text{NP}$, no constant-ratio polynomial time approximation algorithm exists for the general problem $P|\text{set}_j|C_{\max}$ [18]. However, since the DA-RSA problem is a special case of $P|\text{set}_j|C_{\max}$, it is possible that polynomial or approximation algorithms exist for special topologies or spectrum demand matrices. In this section, we present theoretical results on the complexity of the DA-RSA problem.

Before we proceed, we introduce two definitions. First, we let $K_N$ denote a *complete digraph* with $N$ nodes. Since every pair of distinct nodes in $K_N$ is connected by a pair of distinct arcs, one in each direction, the total number of arcs in the graph is equal to $N(N-1)$. Second, in the context of multiprocessor scheduling, we refer to tasks as *compatible* if they can be executed simultaneously, i.e., if the processor sets assigned to the tasks are pairwise disjointed. We now have the following lemma.

*Lemma 3.1:* The DA-RSA problem on complete digraphs $K_N, N \geq 2$ is solvable in polynomial time.

*Proof.* From Lemma 2.1, the multiprocessor scheduling problem instance corresponding to a DA-RSA instance on $K_N$ contains $N(N-1)$ processors, one for each arc of $K_N$. Let us select the shortest path (i.e., a direct arc) for each spectrum demand in the DA-RSA instance. Then, each task in the scheduling instance is to be executed on its own distinct processor. Therefore, the problem reduces to that of scheduling a set of single-processor tasks that are pairwise compatible. Since all tasks may be executed in parallel, the makespan of the schedule is equal to the processing time of the longest task. Recall that all instances of $P|\text{set}_j|C_{\max}$ constructed from an instance of the DA-RSA problem are such that the processing times of the tasks satisfy Eq. (2). Therefore, this makespan is optimal. ∎

Although DA-RSA may be solved in polynomial time on a complete digraph using shortest-path routing, as the above lemma implies, the next three results show that there exists a computational cliff such that slightly modifying the complete digraph renders the problem intractable. Specifically, we prove that the DA-RSA problem on general topologies derived by deleting arcs from a complete digraph is NP-complete.

*Theorem 3.1:* DA-RSA on a digraph $K'_4$ obtained by deleting the two arcs[1] between any pair of nodes of $K_4$ is NP-complete.

*Proof.* Consider the four-node digraph $K'_4$ obtained from $K_4$ after removing the two arcs between nodes 1 and 3, as shown in Fig. 1; note that, because of symmetry, the proof holds if the arcs between any pair of nodes of $K_4$ are removed. Let $\{1, 2, 3, 4, 5, 1', 2', 3', 4', 5'\}$ represent the ten arcs of this network, as labeled in the figure. Following the transformation described in Lemma 2.1, we transform an instance of DA-RSA on digraph $K'_4$ to an instance of $P|\text{set}_j|C_{\max}$ with $m = 10$ processors, and we represent each processor using the same label as the corresponding arc of $K'_4$.

Let $\mathcal{P}_4$ represent the set of $P|\text{set}_j|C_{\max}$ instances corresponding to DA-RSA instances defined on digraph $K'_4$. By construction, each DA-RSA instance on $K'_4$ transforms to a unique instance of $\mathcal{P}_4$ and, therefore, the reverse is also true, i.e., each instance of $\mathcal{P}_4$ transforms back to a unique instance of DA-RSA on $K'_4$. We now show that the scheduling problem $\mathcal{P}_4$ is NP-complete; since $\mathcal{P}_4$ transforms to DA-RSA on $K'_4$, the latter problem is NP-complete as well.

The proof is by reduction from the PARTITION problem [19], which is defined as follows:

*Definition 3.1 (PARTITION):* Given a set of $k$ integers $A = \{a_1, a_2, \ldots, a_k\}$, such that $B = \sum_{j=1}^{k} a_j$, does there exist a partition of $A$ into two sets, $A_1$ and $A_2$, such that $\sum_{a_j \in A_1} a_j = \sum_{a_j \in A_2} a_j = \frac{B}{2}$?

Given an instance of PARTITION, we create an instance of $\mathcal{P}_4$ with the ten processors labeled $\{1, 2, 3, 4, 5, 1', 2', 3', 4', 5'\}$, as shown in Fig. 2. Specifically, for each $a_j \in A$, we create a task $\tau_j$ with processing time $p_j = a_j$ and $\text{set}_j = \{\{2\}, \{5, 3'\}, \{1', 4', 3'\}\}$. Furthermore, we create the eleven tasks listed in Table I.

Each task created for the $\mathcal{P}_4$ instance corresponds to a demand between a pair of nodes in the DA-RSA problem on $K'_4$. For instance, consider task $T_a$ in Table I. Referring to Fig. 1, task $T_a$ corresponds to a demand from node 1 to node 4 in $K'_4$, and the three alternate sets of processors for the task (i.e., the sets in the third column of the first row in Table I) correspond to the three paths from 1 to 4 in $K'_4$, i.e., $\langle 4' \rangle$, $\langle 1, 5 \rangle$, and $\langle 1, 2, 3 \rangle$, respectively. Also, since the processing time of each task is independent of the set of processors on which the task is executed, Eq. (2) is satisfied.

If set $A$ can be partitioned into $A_1$ and $A_2$ such that $\sum_{a_j \in A_1} = \sum_{a_j \in A_2} = B/2$, then there exists a feasible schedule for the $\mathcal{P}_4$ problem, as shown in Fig. 2 with $C_{\max} = 3B$. This schedule is also optimal since its makespan is equal to the processing time of the longest task.

[1]In typical telecommunication networks, two nodes are directly connected using two links, one in each direction. Hence, in this and the next theorem, we only consider the case of removing both arcs between a pair of nodes. Although it is possible to extend the results to the case of deleting one arc at a time, we will consider this task in future works.

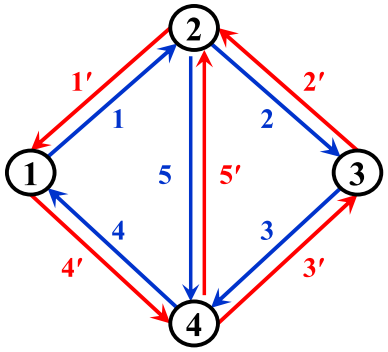


Fig. 1. Digraph $K_4'$ obtained after removing the two arcs between nodes 1 and 3 from $K_4$.

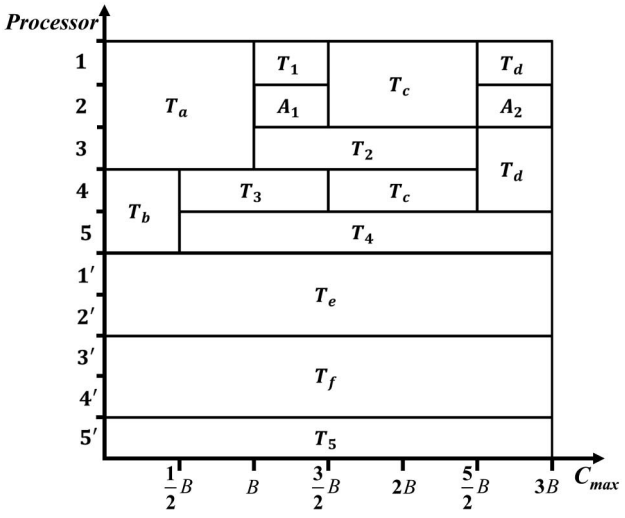


Fig. 2. Feasible schedule with $C_{\max} = 3B$ for the $\mathcal{P}_4$ problem instance of Theorem 3.1.

TABLE I
TASKS CREATED IN THE TRANSFORMATION OF PARTITION $\mathcal{P}_4$

| Task $j$ | $p_j$ | set$_j$ |
|---|---|---|
| $T_a$ | $B$ | $\{\{4'\},\{1,5\},\{1,2,3\}\}$ |
| $T_b$ | $B/2$ | $\{\{1'\},\{5,4\},\{2,3,4\}\}$ |
| $T_c$ | $B$ | $\{\{3'\},\{5',2\},\{4,1,2\}\}$ |
| $T_d$ | $B/2$ | $\{\{2'\},\{3,5'\},\{3,4,1\}\}$ |
| $T_e$ | $3B$ | $\{\{3,4\},\{2',1'\},\{2',5,4\},\{3,5',1'\}\}$ |
| $T_f$ | $3B$ | $\{\{1,2\},\{4',3'\},\{1,5,3'\},\{4',5',2\}\}$ |
| $T_1$ | $B/2$ | $\{\{1\},\{4',5'\},\{4',3',2'\}\}$ |
| $T_2$ | $3B/2$ | $\{\{3\},\{2',5\},\{2',1',4'\}\}$ |
| $T_3$ | $B$ | $\{\{4\},\{5',1'\},\{3',2',1'\}\}$ |
| $T_4$ | $5B/2$ | $\{\{5\},\{2,3\},\{1',4'\}\}$ |
| $T_5$ | $3B$ | $\{\{5'\},\{4,1\},\{3',2'\}\}$ |

Conversely, suppose that there exists a feasible schedule $\mathcal{S}$ with $C_{\max} \leq 3B$. Since tasks $T_5$, $T_e$, and $T_f$ have lengths equal to $3B$, they must be executed in parallel, i.e., they must be assigned to processor sets that are pairwise disjointed. Specifically, assigning $T_5$ on either of its two-processor sets would create a schedule of length longer than $3B$; hence, it must be executed on processor $5'$, as shown in Fig. 2. As a result, $T_e$ and $T_f$ *must* be scheduled on processor sets $\{2', 1'\}$ and $\{4', 3'\}$, respectively. Given this assignment, the five processors $1', 2', 3', 4'$, and $5'$ are busy in the interval $[0, 3B]$. Therefore, the remaining tasks must be executed on processors 1, 2, 3, 4, and 5 to guarantee that the length of the schedule $C_{\max} \leq 3B$.

We also note that the processing time of $T_4$ is $5B/2$, so this task *must* be executed on processor 5 to ensure that the makespan does not exceed $3B$. Consequently, $T_b$ is the only task that can be scheduled on processor set $\{4, 5\}$ to keep $C_{\max} \leq 3B$. In turn, this observation implies that task $T_a$ *must* be executed on set $\{1, 2, 3\}$.

Without loss of generality, suppose $T_a$ is executed before $T_d$ in schedule $\mathcal{S}$; otherwise, similar arguments can be used to reach the same conclusion. Among the remaining tasks compatible with $T_a$, $T_b$ is the only one that *must* be executed in parallel with $T_a$ to yield a makespan of no more than $3B$. Next, tasks $T_3$ and $T_4$ *must* be scheduled immediately after the completion of $T_b$. As $T_a$ completes at time $B$, earlier than $T_3$, tasks $T_1$ and $T_2$ *must* be scheduled right after $T_a$; otherwise, the schedule length would be greater than $3B$. Similarly, as soon as $T_1$ and $T_3$ complete at time $3B/2$, $T_c$ *must* be executed in parallel with $T_2$ and $T_4$. Finally, we observe that $T_d$ *must* start at time $5B/2$ to ensure that the makespan does not exceed $3B$. The corresponding schedule of tasks is shown in Fig. 2 and is such that only the intervals $[B, 3B/2]$ and $[5/2B, 3B]$ can be used to execute the PARTITION jobs. Thus, a partition of set $A$, i.e., a solution to the PARTITION problem, exists. ∎

Theorem 3.1 shows that removing the two arcs between any pair of nodes of $K_4$ renders the DA-RSA problem on the resulting graph $K_4'$ NP-complete. The following theorem shows that removing two pairs of arcs from $K_5$ yields a problem that is also NP-complete.

*Theorem 3.2:* DA-RSA on a digraph $K_5'$ obtained by deleting the two arcs between any two pairs of nodes of $K_5$ is NP-complete.

*Proof.* The two pairs of nodes in $K_5$ whose arcs are removed may or may not have one node in common. We investigate each of these cases separately.

Figure 3(a) illustrates the digraph $K_5'$ after removing the two arcs between nodes 1 and 3 and nodes 2 and 4. Consider a DA-RSA instance in which the spectrum demands to and

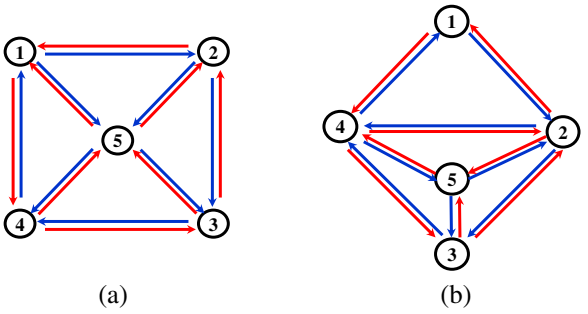


Fig. 3. Digraph $K_5'$ created from $K_5$ by removing two pairs of arcs: (a) arcs between nodes 1 and 3 and nodes 2 and 4 (i.e., with no common node) and (b) arcs between nodes 1 and 3 and nodes 1 and 5 (i.e., with one common node).

from node 5 are very large, i.e., $t^l_{5j} = t^l_{j5} = M, j = 1, \ldots, 4$, where $M$ is a large number. In this case, an optimal solution must be such that 1) all these large demands must be routed across the corresponding direct arc and 2) the arcs that carry the large demands are not used to carry any other traffic. The DA-RSA problem for the remaining node pairs $(i,j), i,j = 1, \ldots, 4, i \neq j$, is equivalent to the DA-RSA on a four-node bidirectional ring. In an earlier work [20], we showed that the RSA problem on four-node bidirectional rings is NP-complete; since RSA is a special case of DA-RSA, the latter is also NP-complete on rings with four or more nodes.

Now consider the case where the two pairs of nodes in $K_5$ whose arcs are removed have one node in common. Let node 1 be the common node. If we remove the two arcs between nodes 1 and 5 and nodes 1 and 3, the result will be the digraph $K'_5$ shown in Fig. 3(b); due to symmetry, this proof applies to any node as the common node and any other two nodes for removing arcs. Let the spectrum demands to and from node 5 in Fig. 3(b) be very large, i.e., $t^l_{5j} = t^l_{j5} = M, j = 2, \ldots, 4$. As we observed in the previous case, these large demands must use the direct arcs to/from node 5, and in turn, these direct arcs may not be used to carry other traffic. DA-RSA for the remaining demands is defined on a digraph identical to the one in Fig. 1, a problem we proved to be NP-complete in Theorem 3.1. ∎

We now provide the following complexity result for the DA-RSA problem on general graphs.

*Lemma 3.2:* Let $G$ be a digraph. If there exists a vertex-induced subgraph of $G$ that is a ring of four or more nodes, then the DA-RSA problem on $G$ is NP-complete.

*Proof.* Let $R$ be a vertex-induced subgraph of $G$ that is a ring of four or more nodes. Consider an instance of DA-RSA on $G$ with the following spectrum demands: 1) arbitrary, between nodes of subgraph $R$; 2) equal to a large number, $M$, between adjacent nodes not in the subgraph $R$; and 3) equal to zero, between non-adjacent nodes not in the subgraph $R$. Similar to the observations in the previous theorem, in the optimal solution, each arc of $G$ that is not part of the subgraph $R$ only carries the traffic between directly connected nodes. Hence, this instance reduces to a DA-RSA subproblem on ring $R$ with four or more nodes, which, according to the results in [20], is NP-complete. ∎

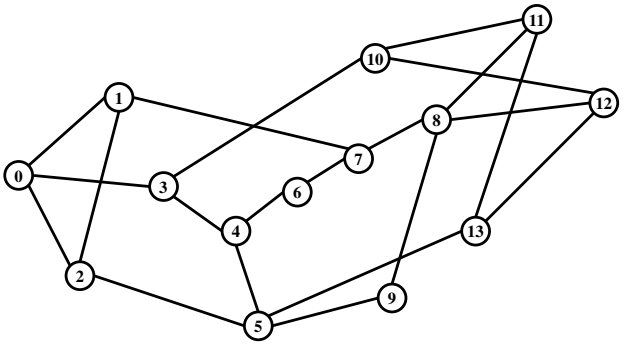


Fig. 4. NSFNet topology.

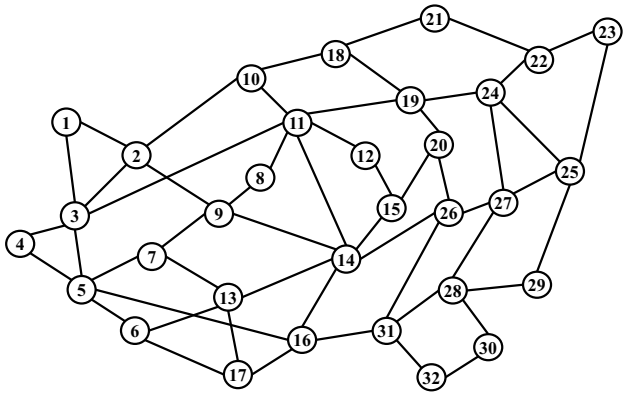


Fig. 5. GEANT2 topology.

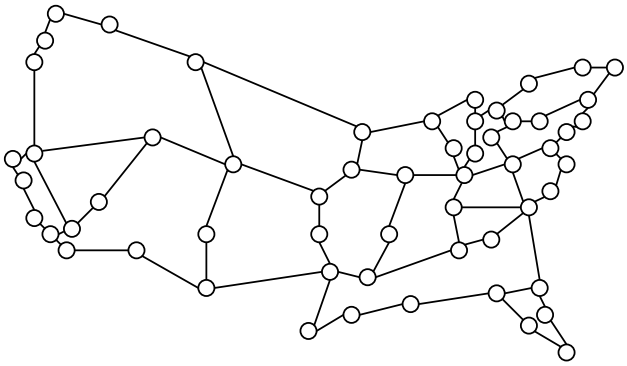

Fig. 6. 60-node network topology derived from the CORONET CONUS.

Note that typical telecommunications networks are connected; hence, they are highly likely to include subgraphs that are rings and satisfy the condition of the above lemma. For instance, all three topologies in Figs. 4–6 that we use for our experimental study have this property.

## IV. List-Scheduling Algorithm for $Pm|\text{set}_j|C_{\max}$

In this section, we propose a list-scheduling (LS) algorithm for the $Pm|\text{set}_j|C_{\max}$ problem. Since DA-RSA is a special case of $Pm|\text{set}_j|C_{\max}$, this algorithm can be used to solve the DA-RSA problem in networks of general topology. This is accomplished in three steps: 1) the DA-RSA instance at hand is first transformed to an instance of $Pm|\text{set}_j|C_{\max}$ following the process described in Lemma 2.1, 2) the LS algorithm is applied to construct a schedule that solves the scheduling instance, and 3) the schedule is transformed back to a solution of the DA-RSA instance.

The input to the LS algorithm is a list of tasks $L$, along with their corresponding $k$ alternate sets of processors. Tasks in the list are sorted in decreasing order of the processing time on their smallest processor set; ties are broken by the size (i.e., the number of processors) of their smallest processor set, and further ties are broken arbitrarily. For each task, its alternate processor sets are sorted in increasing order of their size.

**List Scheduling Algorithm for** $Pm|set_j|C_{max}$
**Input:** A list $L$ of $n$ tasks on $m$ processors, each task $j$ defined by the set $set_j = \{S_j^1, \ldots, S_j^k\}$ of $k$ alternative processor sets on which it may be executed, and the corresponding processing times $\{p_j^1, \ldots, p_j^k\}$
**Output:** A schedule of tasks, i.e., the time $T_j$ when task $j$ starts execution, along with the set $S_j$ of processors assigned to it and the corresponding processing time $p_j$

```
begin
  1.  t ← 0   //Scheduling instant
  2.  F ← {1, ..., m}   //Set of currently idle processors
  3.  while list L ≠ ∅ do
  4.      j ← first task in list L
  5.      S_j ← ∅   //Set of processors to execute task j
  6.      p_j ← 0   //Processing time of task j
  7.      for z ← 1 to k
  8.          if S_j^z ⊆ F then
  9.              S_j ← S_j^z
  10.             p_j ← p_j^z
  11.             T_j ← t
  12.             F ← F \ S_j
  13.             Remove the task j from list L
  14.             break
  15.     while not at the end of list L or F ≠ ∅ do
  16.         i ← first task in list
  17.         for w ← 1 to k
  18.             if S_i^w ⊆ F then
  19.                 S_i ← S_i^w
  20.                 p_i ← p_i^w
  21.                 T_i ← t
  22.                 F ← F \ S_i
  23.                 Remove the task i from list L
  24.                 break
  25.     end while  // no more tasks may start at time t
  26.     j ← the first task executing at time t to complete
  27.     t ← T_j + p_j
  28.     F ← F ∪ S_j
  19. end while
end
```

Fig. 7. LS algorithm to select one set $S_j$ and its corresponding processing time $p_j$ to execute each task $j$ of the $Pm|\text{set}_j|C_{\max}$ problem.

At each scheduling instant $t$, the algorithm scans the list $L$ to find the first task $j$ and processor set $S_j^l$ that is compatible with the tasks already executing at this time $t$. This set $S_j^l$ of processors is selected to execute task $j$ starting at time $t$, and the algorithm removes the task from $L$. The algorithm updates the set of free processors at time $t$ and continues scanning list $L$, repeating the above process until no other compatible task is found. Then, the algorithm advances $t$ to the earliest time $t' > t$ at which one of the currently executing tasks will be completed, releases the set of processors assigned to the just-completed task, and repeats the above actions for time $t'$. The algorithm continues in this manner until all tasks in list $L$ have been scheduled.

A pseudocode description of the LS algorithm is provided in Fig. 7. Both the outer and inner **while** loops of the algorithm take at most $O(n)$ time in the worst case, where $n$ is the number of tasks in the scheduling problem. Both **for** loops take time $O(k)$ in the worst case, where $k$ is the number of alternate processor sets. Therefore, the running time complexity of the LS algorithm is $O(kn^2)$. Since the number of tasks corresponds to the number of spectrum demands, the complexity of the algorithm when applied to the DA-RSA problem is $O(kN^4)$, where $N$ is the number of nodes and $k$ is the number of alternate paths.

## V. Numerical Results

We have evaluated the performance of the LS algorithm by carrying out simulation experiments with a large number of DA-RSA problem instances. Each problem instance is characterized by three parameters: 1) the network topology, 2) the number $k$ of shortest paths for each source–destination pair, and 3) a randomly generated spectrum demand matrix.

### A. Topology and Shortest Paths

In our evaluation study, we have used three general topology networks of varying sizes and average nodal degrees:

- the 14-node, 42-arc (directed link) NSFNet shown in Fig. 4;
- the 32-node, 108-arc GEANT2 topology depicted in Fig. 5; and
- the 60-node, 154-arc network topology illustrated in Fig. 6 and adapted from CORONET CONUS [21].

We used Yen's algorithm [22] to compute the $k$ loop-less shortest paths, $k = 1, \ldots, 7$, between each pair of nodes in each topology. Yen's algorithm takes time $O(N^3)$, where $N$ is the number of nodes. For the experiments we present in this section, we assumed that all links have unit weight for the purposes of computing the shortest paths.

### B. Spectrum Demand Matrix

For each DA-RSA problem instance, we randomly generate a spectrum demand matrix in two steps: traffic demand generation and DA spectrum allocation.

1) *Traffic Demand Generation:* We assume that the elastic optical network supports the following data rates (in Gbps): 10, 40, 100, 400, and 1000. Therefore, in the first step, the traffic rates between every pair of nodes are drawn from one of three probability distributions:

- *Distance-independent:* each value in the set $\{10, 40, 100, 400, 1000\}$ is selected with equal probability.
- *Distance-increasing:* the probability assigned to each value in the set $\{10, 40, 100, 400, 1000\}$ depends on the length of the shortest path between the source and destination nodes, such that the probability of higher values in the set *increases* with the length of the shortest path.
- *Distance-decreasing:* the probability assigned to higher values in the set $\{10, 40, 100, 400, 1000\}$ *decreases* with the length of the shortest path between the source and destination nodes.

2) *Distance-Adaptive Spectrum Allocation:* In the second step, we determine the number $t_{sd}^l$ of spectrum slots

required for the traffic demand to be carried on the $l$th alternate path, $l = 1, \ldots, k$, from source $s$ to destination $d$. In DA spectrum allocation, the number of slots depends on both the data rate and the length of the path [2,15]. We adopt the parameters of the study in [15] and assume a slot width of 12.5 GHz and three modulation formats:

- *Paths with up to 4 links:* the 64-QAM modulation format is used such that data rates of 10, 40, 100, 400, and 1000 Gbps require 1, 1, 2, 6, and 14 spectrum slots, respectively.
- *Paths with 5–9 links:* the 16-QAM modulation format applies, such that rates of 10, 40, 100, 400, and 1000 Gbps are assigned 1, 1, 2, 8, and 20 slots, respectively.
- *Paths with 10 or more links:* the QPSK modulation format is utilized, and data rates of 10, 40, 100, 400, and 1000 Gbps are allocated 1, 2, 4, 16, and 40 spectrum slots, respectively.

### C. Evaluation Metrics

The first metric we consider is the maximum number of spectrum slots on any link in the network required by the solution to a DA-RSA problem instance obtained by the LS algorithm. We denote this value as $\text{MaxSlots}_{\text{LS}}$; as the reader may recall, this value is equivalent to the length of the schedule constructed by the LS algorithm for the corresponding scheduling problem instance. This metric can provide insight into the impact of the number $k$ of alternate paths or the traffic rate distribution on the use of spectrum resources in the network.

In order to evaluate the quality of the LS algorithm, and since the optimal solution cannot be obtained in polynomial time, it is important to compute a lower bound (LB). Let $D_q^{\text{in}}$ and $D_q^{\text{out}}$ denote the in- and out-degrees of node $q$. A simple LB for the DA-RSA problem can be calculated as follows:

$$\text{LB} = \max\left\{\max_s \sum_d t_{sd}/D_s^{\text{out}}, \max_d \sum_s t_{sd}/D_d^{\text{in}}\right\}, \quad (3)$$

where $t_{sd}$ in the above expression is the spectrum demand for the traffic from $s$ to $d$ along the shortest path between the two nodes. The metric we use to characterize the LS algorithm is the ratio

$$R = \text{MaxSlots}_{\text{LS}}/\text{LB}. \quad (4)$$

Clearly, $R \geq 1.0$; the closer $R$ is to 1.0, the better the performance of the algorithm. We note, however, that the LB in Eq. (3) only considers the spectrum demands in and out of each node and does not account for the interactions of these demands along the links of the network; therefore, we expect the bound to be loose.

The figures we present in the next section report the average values for either $\text{MaxSlots}_{\text{LS}}$ or $R$. Specifically, each data point on these figures is the average of 10 replications of a random experiment; in turn, each replication is the average of 30 random instances generated for the stated parameters (i.e., topology, number $k$ of paths, and traffic rate distribution). The figures also report 95% confidence intervals, which can be seen to be narrow.

### D. Results and Discussion

The three Figs. 8–10 plot the maximum number of spectrum slots, $\text{MaxSlots}_{\text{LS}}$, as a function of the number $k$ of alternate paths, for the NSFNet, GEANT2, and 60-node topologies, respectively. Each figure includes three curves, each representing results for problem instances with spectrum demand matrices generated by the distance-independent, distance-increasing, and distance-decreasing distributions, respectively.

We first observe that the amount of spectrum increases with the size of the network, reflecting the corresponding increase in traffic demands due to the larger number of source–destination pairs. Nevertheless, the overall behavior of the curves is consistent across the three traffic distributions and network topologies. Specifically, the amount of spectrum resources is high for shortest-path routing

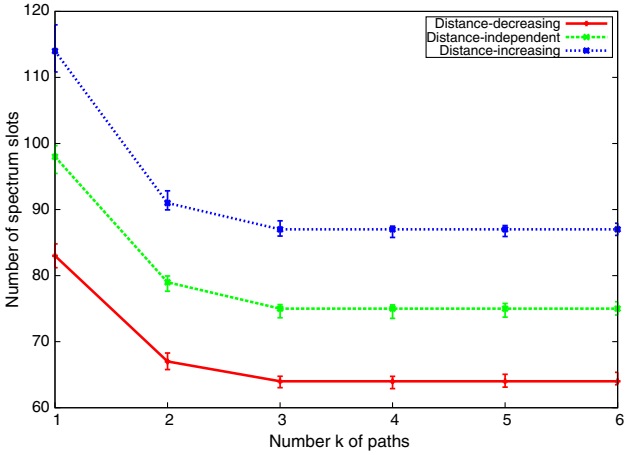


Fig. 8. Spectrum slots versus number $k$ of paths in NSF.

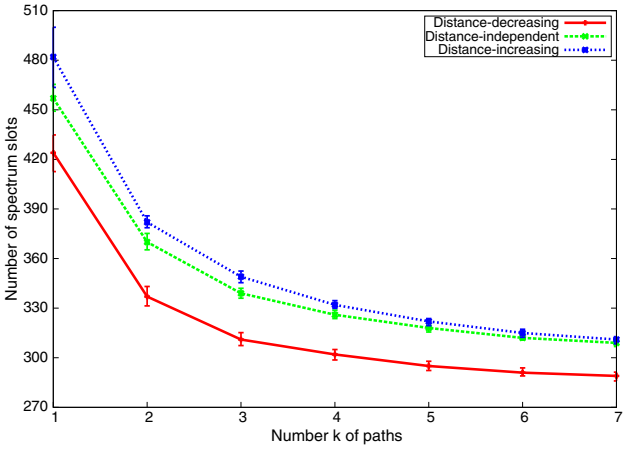


Fig. 9. Spectrum slots versus number $k$ of paths in GEANT2.

($k = 1$) but drops sharply (between 20% and 50%, depending on the distribution and topology) when demands may be routed along one of $k = 2$ alternate paths. As the number $k$ of alternate paths increases further, the number of spectrum slots decreases more slowly and eventually levels off, indicating the diminishing returns of employing each additional path.

A final observation from the three figures is that the solution to the DA-RSA problem is highly sensitive to the traffic demand distribution. Specifically, everything else being equal, the distance-increasing distribution requires more spectrum than the distance-independent distribution, which, in turn, is more resource-intensive than the distance-decreasing distribution. This result can be explained by the fact that demands between nodes that are far away from each other consume more spectral resources in the network than the same demands between two nearby nodes due to 1) the larger number of links in the paths they travel and 2) the wider spectrum that is required to carry the demand if the length of its path crosses the threshold into a lower-level modulation with a high SNR tolerance.

Let us now turn our attention to the three Figs. 11–13, which plot the average ratio $R$ in Eq. (4) against the number $k$ of paths for each of the three network topologies; again, each figure includes three plots, one per demand distribution. Note that the LB in Eq. (3) is independent of the number $k$ of alternate paths for each demand. Since the number of required slots, $\text{MaxSlots}_{\text{LS}}$, decreases with $k$, as seen in the previous three figures, we expect $R$ to decrease as well, and this is exactly what we observe in Figs. 11–13.

Nevertheless, there is an important difference between the three figures that plot the absolute value of spectrum slots required and the ones that show the average ratio. Specifically, we observe that there are significant gaps between the various curves in each of Figs. 8–10 which, as we explained above, are due to the combined effects of the demand distribution and DA spectrum allocation. On the other hand, the curves of the various distributions in Figs. 11–13 are closer to each other, and the average ratios of the three distributions converge to similar values. Recall that the LB in Eq. (3) depends on the demands in and out of each node in the network, and hence, it depends

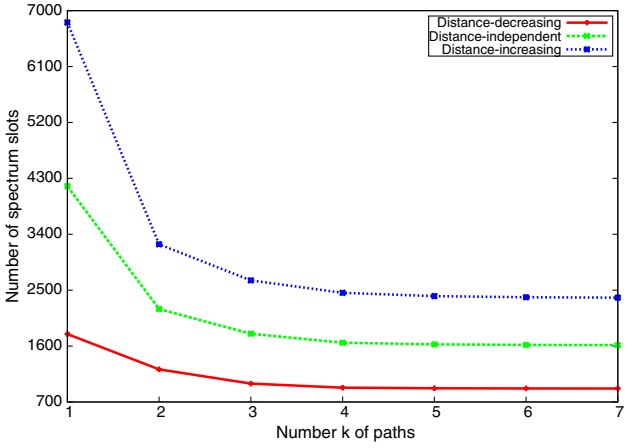


Fig. 10. Spectrum slots versus number $k$ of paths in 60-node network.

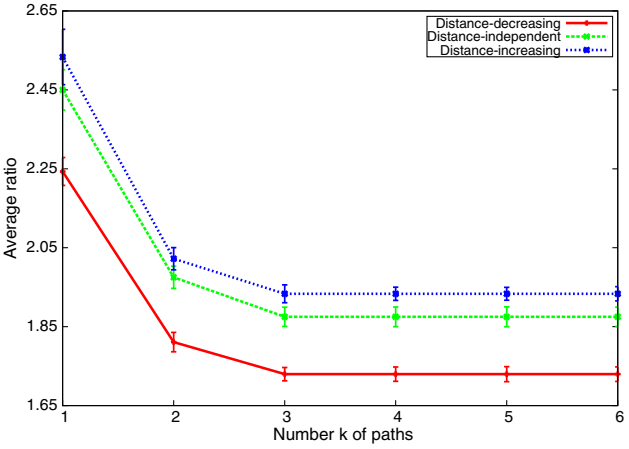


Fig. 11. Average ratio versus number $k$ of paths in NSF.

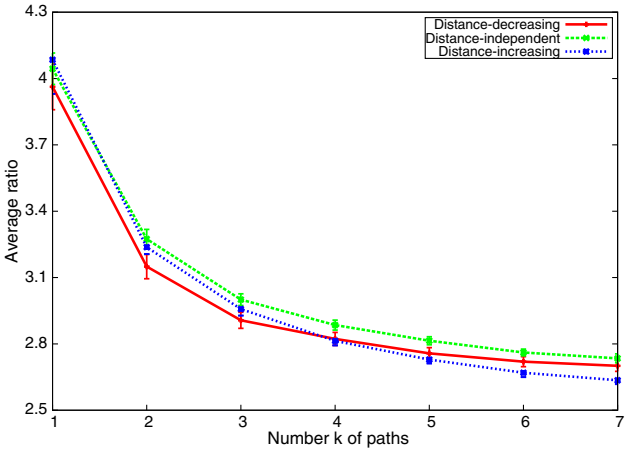


Fig. 12. Average ratio versus number $k$ of paths in GEANT2.

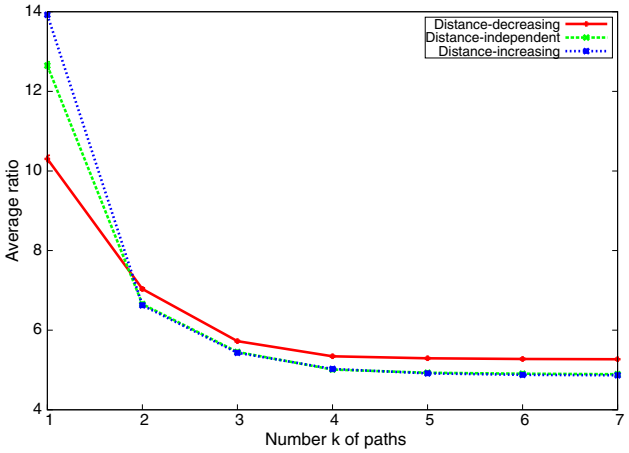


Fig. 13. Average ratio versus number $k$ of paths in 60-node network.

on the traffic distribution. Therefore, the behavior of the curves in Figs. 11–13 is a strong indication that, for the topologies and distributions we considered in this study, the LS algorithm is capable of exploiting alternate paths to construct solutions that move towards the LB, regardless of the absolute value of the spectrum slots required in each problem instance.

Finally, we note that the average ratio of the LS algorithm increases with the size of the network, from around 1.8 for the NSFNet to around 2.7 for GEANT2 and about 5 for the 60-node network (these ratio values are for the largest number of alternate paths shown in the figures). This increase is partly due to the heuristic nature of the LS algorithm: as the size of the problem increases, the size of the solution space increases exponentially, whereas the set of solutions examined by the algorithm increases polynomially; hence, the probability of finding good-quality solutions decreases. However, we argue that a significant part of the increase in the ratio is due to the increase in the gap between the LB and optimal solution as the network size increases. In particular, as the network size grows, the spectrum allocation is affected by the interaction of an increasing number of traffic demands over an increasing number of paths and links. Since Eq. (3) for the LB does not account for these interactions, we expect that LB becomes looser and further disconnected from the optimal. Therefore, we conjecture that the LS algorithm performs significantly better relative to the LB than Figs. 11–13 suggest, especially for the GEANT2 and 60-node networks.

Overall, the results in this section indicate that the LS algorithm is effective in using a small number of alternate paths (i.e., $k = 5, 6$) to utilize spectrum resources efficiently by balancing the traffic demands across the network links.

## VI. Concluding Remarks

We have shown that the distance-adaptive routing and spectrum assignment (DA-RSA) problem with fixed alternate routing in mesh networks transforms to a well-known processor scheduling problem. We have also developed a computationally efficient algorithm that builds upon list-scheduling concepts to jointly tackle the RSA aspects of DA-RSA. Our work explores the tradeoffs involved in DA-RSA algorithm design and opens up new research directions in leveraging the vast literature in scheduling theory to address important and practical problems in network design.

## Acknowledgment

This work was supported by the National Science Foundation under grant CNS-1113191.

## References

[1] G. Zhang, M. De Leenheer, A. Morea, and B. Mukherjee, “A survey on OFDM-based elastic core optical networking,” *IEEE Commun. Surv. Tutorials*, vol. 15, no. 1, pp. 65–87, 2013.

[2] O. Gerstel, M. Jinno, A. Lord, and S. J. B. Yoo, “Elastic optical networking: A new dawn for the optical layer?” *IEEE Commun. Mag.*, vol. 50, no. 2, pp. s12–s20, 2012.

[3] M. Jinno, H. Takara, and B. Kozicki, “Dynamic optical mesh networks: Drivers, challenges and solutions for the future,” in *35th European Conf. Optical Communication (ECOC)*, Sept. 20–24, 2009, paper 7.7.4.

[4] G. Shen and M. Zukerman, “Spectrum-efficient and agile CO-OFDM optical transport networks: Architecture, design, and operation,” *IEEE Commun. Mag.*, vol. 50, no. 5, pp. 82–89, 2012.

[5] S. Talebi, F. Alam, I. Katib, M. Khamis, R. Khalifah, and G. N. Rouskas, “Spectrum management techniques for elastic optical networks: A survey,” *Opt. Switching Netw.*, vol. 13, pp. 34–48, July 2014.

[6] Y. Wang, X. Cao, and Y. Pan, “A study of the routing and spectrum allocation in spectrum-sliced elastic optical path networks,” in *Proc. IEEE INFOCOM*, 2011, pp. 1503–1511.

[7] A. N. Patel, P. N. Ji, J. P. Jue, and T. Wang, “Routing, wavelength assignment, and spectrum allocation algorithms in transparent flexible optical WDM networks,” *Opt. Switching Netw.*, vol. 9, no. 3, pp. 191–204, 2012.

[8] X. Wan, L. Wang, N. Hua, H. Zhang, and X. Zheng, “Dynamic routing and spectrum assignment in flexible optical path networks,” in *Optical Fiber Communication Conf. and the Nat. Fiber Optic Engineers Conf. (OFC/NFOEC)*, Mar. 6–11, 2011, paper JWA55.

[9] X. Wang, Q. Zhang, I. Kim, P. Palacharla, and M. Sekiya, “Blocking performance in dynamic flexible grid optical networks—What is the ideal spectrum granularity?” in *37th European Conf. Optical Communication (ECOC)*, Sept. 18–22, 2011, paper Mo.2.K.6.

[10] B. Kozicki, H. Takara, Y. Sone, A. Watanabe, and M. Jinno, “Distance-adaptive spectrum allocation in elastic optical path network (SLICE) with bit per symbol adjustment,” in *Optical Fiber Communication Conf. and the Nat. Fiber Optic Engineers Conf. (OFC/NFOEC)*, Mar. 21–25, 2010, paper OMU3.

[11] S. Yang and F. Kuipers, “Impairment-aware routing in translucent spectrum-sliced elastic optical path networks,” in *17th European Conf. Networks and Optical Communications (NOC)*, June 20–22, 2012.

[12] K. Wen, Y. Yin, D. J. Geisler, S. Chang, and S. J. B. Yoo, “Dynamic on-demand lightpath provisioning using spectral defragmentation in flexible bandwidth networks,” in *37th European Conf. Optical Communication (ECOC)*, Sept. 18–22, 2011, paper Mo.2.K.4.

[13] Y. Zhang, X. Zheng, Q. Li, N. Hua, Y. Li, and H. Zhang, “Traffic grooming in spectrum-elastic optical path networks,” in *Optical Fiber Communication Conf. and the Nat. Fiber Optic Engineers Conf. (OFC/NFOEC)*, Mar. 6–10, 2011, paper OTuI1.

[14] A. Eira, J. Pedro, and J. Pires, “Optimized design of shared restoration in flexible-grid transparent optical networks,” in *Optical Fiber Communication Conf. and the Nat. Fiber Optic Engineers Conf. (OFC/NFOEC)*, 2012, paper JTh2A37.

[15] M. Jinno, B. Kozicki, H. Takara, A. Watanabe, Y. Sone, T. Tanaka, and A. Hirano, “Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network,” *IEEE Commun. Mag.*, vol. 48, no. 8, pp. 138–145, 2010.

[16] L. Bianco, J. Blazewicz, P. Dell’Olmo, and M. Drozdowski, “Scheduling multiprocessor tasks on a dynamic configuration of dedicated processors,” *Ann. Oper. Res.*, vol. 58, no. 7, pp. 493–517, 1995.

[17] M. Kubal, "The complexity of scheduling independent two-processor tasks on dedicated processors," *Inf. Process. Lett.*, vol. 24, no. 3, pp. 141–147, 1987.

[18] L. Torres, A. Miranda, and J. Chen, "On the approximability of multiprocessor task scheduling problems," in *13th Annu. Int. Symp. Algorithms and Computation*, 2002, vol. 2518, pp. 403–415.

[19] M. R. Garey and D. S. Johnson, *Computers and Intractability*, New York, NY: W. H. Freeman and Co., 1979.

[20] S. Talebi, E. Bampis, G. Lucarelli, I. Katib, and G. N. Rouskas, "On routing and spectrum assignment in rings," *J. Lightwave Technol.*, vol. 33, no. 1, pp. 151–160, Jan. 2015.

[21] "DARPA Core Optical Networks (CORONET) Continental United States (CONUS) topology," [Online]. Available: http://monarchna.com/CORONET_CONUS_Topology.xls.

[22] J. Y. Yen, "Finding the *K* shortest loopless paths in a network," *Management Sci.*, vol. 17, no. 11, pp. 712–716, 1971.