Ain Shams Engineering Journal 11 (2020) 273–280

Contents lists available at ScienceDirect

Ain Shams Engineering Journal

journal homepage: www.sciencedirect.com

Electrical Engineering

Recursive algorithm for selecting optimum routing tables to solve offline routing and spectrum assignment problem

Mahmoud Fayez [a], Iyad Katib [b], George N. Rouskas [c], Tarek F. Gharib [a], Heba Khaleed [a], Hossam M. Faheem [a,*]

[a] Ain Shams University, Egypt
[b] King Abulaziz Universtiy, Saudi Arabia
[c] NC State University, United States

ARTICLE INFO

Article history:
Received 15 July 2019
Revised 9 October 2019
Accepted 10 October 2019
Available online 1 November 2019

Keywords:
RSA
Exhaustive routing search
RSA decomposition

ABSTRACT

The Routing and Spectrum Assignment (RSA) problem is NP-Hard so searching the entire problem space is not applicable. Many decomposition algorithms rely on reducing the search space in the routing space and applying heuristics algorithm in the spectrum assignment sub-problem. This is not necessarily a right solution as the ignored routing tables may lead to a better solution when they are used later as input to the Spectrum Assignment sub-problem. In this paper, we develop a new recursive decomposition approach for the RSA problem in optical networks. At the core of our approach is a new recursive branch and-bound procedure for carrying out an exhaustive search of the routing space in a scalable manner. This recursion effectively decouples the routing from the spectrum assignment part of the problem. Sequential generation of the full set of routing tables requires huge memory and very large processing time. Alternatively, our approach deploys multi-core architectures to generate the routing tables in parallel using OpenMP. Experimental results indicate that our recursive algorithm is quite efficient in searching the entire routing space for topologies representing large-scale wide area networks. Importantly, the decomposition may be more generally applied to any network design problem whose solution involves a search over both a routing and a resource allocation space. The main contributions for this paper are that we are able to generate all the search space in parallel in less than 1 min for 32-nodes network. Secondly, we are able to investigate all the routing tables, eliminate most of the search space, and select the promising routing tables that are proven to lead to a better solution in the Spectrum Assignment Sub-Problem

1. Introduction

Planning, deploying, and engineering the networks that make up the Internet infrastructure involves complex problems that we will refer to generally as "network design" problems. Effective and efficient solutions to network design problems are crucial to the operation and economics of the Internet and its ability to support critical and reliable communication services. This is especially true for optical networks that form the foundation of the global network infrastructure [1].

Optical network design problems have been the subject of numerous studies since the 1980s; for instance, we refer the reader to surveys of the routing and wavelength assignment (RWA) problem [2–4], the traffic grooming problem [5,6], the routing and spectrum assignment (RSA) problem[7,8], and network survivability [9]. Exact solution methods are based on integer linear program (ILP) models that have difficulties to scale to problem instances representing commercial networks. On the other hand, a wide range of heuristics are proposed to be applied to large networks; the challenge with most heuristic algorithms, however, is that generally there is little information regarding the accuracy of the solution [10]. Importantly, such solution approaches are often unique designs that are carefully customized to the problem in hand. Even though, it may be possible to adapt such approaches to other problem variants. Both the initial design and the modifications require expertise not only in the problem domain (i.e., networking and/or

* Corresponding author.
E-mail address: hmfaheem@cis.asu.edu.eg (H.M. Faheem).
Peer review under responsibility of Ain Shams University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.asej.2019.10.008
2090-4479/

graph theory) but also in a range of disciplines including discrete optimization, operations research, and/or mathematical programming. As a result, existing approaches require significant investment in human expertise and computational resources. This in turn limits our ability to explore the sensitivity of design decisions to forecast demands and capital/operating cost assumptions.

With this work we aim to make contributions that will lead to new ways of tackling network design problems. Specifically, we present a decomposition solution that is applicable to a wide range of problems by exploiting a feature common to all, namely, that the optimization process incorporates both a routing aspect and a resource allocation aspect. At the core of our solution is a new recursive branch-and-bound procedure for carrying out an exhaustive search of the routing space in a scalable manner. With an exhaustive routing search, our approach represents an exact decoupling of the routing and resource allocation aspects of the optimization, and hence, it has broad applications.

Following the introduction, in Section 2 we present a reference formulation for the RSA problem that we use to demonstrate our technique. In Section 3 we present a new decomposition approach for this problem, and in Section 4 we develop a scalable recursion for searching the routing space. We present an experimental study to demonstrate the effectiveness of the decomposition approach in Section 4.1, and we conclude the paper in Section 5.

2. ILP formulation for RSA

We consider a basic yet general version of the offline RSA problem that may be expressed using integer linear programming (ILP) formulations and is NP-hard [11]. There are generally two classes of ILP formulations depending on the types of variables used. In link-based formulations [12], the entities of interest (i.e., the decision variables) relate to individual links of the network, whereas in path-based formulations [13], variables relate to complete paths between network nodes. Link-based formulations express the optimization problem as a multi commodity flow problem and force the solver to consider the entire space of possible paths between any two nodes in the network. Path-based formulations, on the other hand, take as input a given set of paths for each pair of network nodes. Consequently, the solution that a path formulation produces is optimal only with respect to the input set of paths and can be no better than the optimal solution reached by an equivalent link formulation of the same problem.

In this paper, we consider the following reference ILP formulation of the RSA problemP_{RSA}. Although this is a path-based formulation, we discuss shortly how it may be used to produce optimal solutions equivalent to a link-based formulation.

2.1. RSA Problem P_{RSA}

The proposed problem formulation does not consider neither the modulation technique nor the path distance as an input. Another assumption is that all traffic demands going from node s to node d will follow the same physical path and will use contiguous spectrum with only guard-bands between them. This assumption is valid as we are only targeting static RSA problem.

The RSA problem has the following inputs:

- **Network topology**: a connected graph $G(V,A)$ where V denotes the set of nodes and A denotes the set of arcs (directed links) in the network.
- **Traffic demands**: a traffic demand matrix $T = [t_{sd}]$, where t_{sd} is a non-negative value representing the amount of traffic from node s to node d, expressed in spectrum slots. Spectrum slots varies depending on bit-rate, modulation technique, and actual distance to be travelled. However, we are not considering those constraints in the current work.
- **Routing paths**: a set of paths $(K_{sd})$for source-destination pair *(s, d)*, such that all traffic from s to d is constrained to take path(s) in this set only; the paths are generated in advance and their number may vary based on the source-destination pair.

The RSA problem has the following constraints:

- **Spectrum contiguity**: each traffic demand is assigned contiguous slots in the spectrum.
- **Spectrum continuity**: each traffic demand is assigned the same spectrum slots on all links of its path.
- **Non-overlapping spectrum**: traffic demands that share the same link are assigned non-overlapping (distinct) spectrum slots.

The expected output of the RSA:

- **Routing Table**: A routing table that optimally fits with the selected spectrum assignment algorithm to achieve the optimum spectrum assignment for the given traffic demand.
- **Spectrum Assignment**: an assignment of spectrum slots to each traffic demand.

Objective

- Minimize the number of spectrum slots assigned on any network link while all constraints are satisfied. The objective function of the proposed problem is described in (1), (2), and (3) as minimization of the used spectrum slots. Also, we are minimizing the number of fragmented frequency slots.

$$UF(l,i) = \begin{cases} 1, & if\ \ FS(i)\ \ is\ utilized\ on\ link\ \ (l) \\ 0, & otherwise \end{cases} \tag{1}$$

$$FF(l,i) = \begin{cases} 1, & if\ \ FS_L(i)\ \ is\ fragmented\ on\ link\ \ (l) \\ 0, & otherwise \end{cases} \tag{2}$$

$$makespan = \max_{1 \le l \le |A|} \sum_{i=1}^{N} UF(l,i) + FF(l,i) \tag{3}$$

$$lower\ \ bound = \max_{0 \le l \le |A|} \sum_{i=1}^{N} UF(l,i) \tag{4}$$

where

- ***A*** is the set of directed links between nodes.
- ***N*** is the maximum number of frequency slices on each link, we assume this number to be very large as we are looking for the smallest number of slices on each link.

$\boldsymbol{FS_L}$ is frequency slice array on link (L)
Lower-bound

- In this work we consider the lower-bound is the maximum number of frequency slots used on any link without fragmentation. So for a given routing table (RT) we can calculate the lower-bound which any SA algorithm cannot perform better than it. The lower-bound is presented as in Eq. (4). Lower-bound can also be calculated before applying the SA algorithm as shown in (5), (6). The fragmentation may resulted from in efficient SA algorithm of the fact that this routing table and traffic demand matrix imposes this fragmentation.

$$RT(i,j,l) = \begin{cases} 1, & if \;\; traffice \;\; demand \;\; from \;\; i \;\; to \;\; j \;\; pass \;\; through \;\; link \;\; (k) \\ 0, & otherwise \end{cases} \tag{5}$$

$$lower \;\; bound = \max_{1 \le l \le |A|} \sum_{i,j=1,1}^{|T|,|T|} RF(i,j,l) * T[i,j] \tag{6}$$

where

- $\boldsymbol{A}$ is the set of directed links between nodes.
- $\boldsymbol{T}$ is matrix of traffic demands

The presented problem formulation is based on the paths of the routing table as shown in (5), but if we evaluated all routing tables and found the best make-span then we can find the optimum solution to the RSA problem. However, this requires very huge computation resources. And even if you ignore the routing tables whose lower bound is less than the current best make span this would require calculating the lower bound for each and every routing table which is exponential number [14]. This needs huge amount of computations. The alternative solution to all of that will be discussed in the algorithm section.

Typically, network design/optimization problems such as problem P_{RSA} are NP-hard and may be solved exactly only for small networks [9]. For larger topologies representative of Internet-scale networks, the ILP formulation cannot be solved to optimality within reasonable amounts of time (e.g., several hours). As a result, heuristic algorithms have generally been used to address the offline network optimization problems; the reader is referred to [7] for a comprehensive survey of algorithms for a range of RSA problem variants. Other approaches tackle the problem by manipulating the ILP formulation using decomposition or column generation techniques to address a main challenge, namely, the large number of possible paths to route the traffic demands.

A typical decomposition approach tackles the problem by decomposing it into two simpler ILPs and solving them sequentially [15]. The first ILP addresses only the demand routing problem and generates network paths that can lead to a good solution. The second ILP addresses the spectrum assignment problem along the paths produced by the first ILP. This solution process iterates until the performance metric cannot be improved further. As the problem is divided into two sequential sub problems, the solution is not guaranteed to be optimal – under certain circumstances, this approach may not even lead to a feasible solution. Column generation techniques are also iterative in nature, and attempt to reach a near-optimal solution without generating (or considering) all possible paths [16,17].

Klinkowski et al. [18] presented a base line algorithm implemented on IBM CPLEX optimizer[19].The baseline algorithm used default CPLEX settings to solve the RSA problem for DT12 and BT22 using Branch and Bound (BB) technique with different number of connections (traffic demands). They proposed another enhancement in the same paper using Branch and Price technique which is using a k-shortest paths algorithm to select 30 lightpaths as initial seed for each node for BB algorithm. This reduced the computation time and memory requirements significantly.

Klinkowski et al. [18] succeeded in solving relatively large networks. They start with 30 lightpaths for each pair of network nodes. Different traffic demands between each pair of nodes may use different lightpaths. Column Generation is used to generate more lightpaths if needs. Their proposed work is useful and promising however their lower-bound is calculated on number of Frequency Slices that are used on any of the optical links, which ignores the fragmentation if any. Also the k-shortest paths used may not be the optimum if the Path P_{K+i} will lead to a better solution. The proposed technique has eliminated most of the light paths without any clear justification except that they are selected based on shortest path metric. Consequently, it can't be considered to yield the optimal solution.

Julian et al. [20] proposed column generation solver that maximizes the throughput of the network. There trials have focused on Spain and USA network topologies. The results of these experiments can't be compared with the most of the work done in the RSA domain as they are neither calculating the gap nor the solution quality. Instead, they provide insights if the traffic demands can be full-filed or not.

Both solutions [18] and [20] did not mention the number of the Frequency Slots (FSs) required and how they bench-marked their results. This makes the comparison between our approach and those approaches difficult as it will be discussed in the results section. Both solutions also did not mention the traffic size. They only mention how many traffic demands (connections) can be satisfied. In this work, We are clarifying the traffic generation algorithm to facilitate the comparison of the work with other future implementations.

3. Decomposition algorithm

Path-based formulations only pick set of paths as an input to the Routing Table Selector shown in Fig. 1. On the other hand the proposed decomposition algorithm has a preprocessing step to generate all possible paths between each pair in the network. This leads to selecting the optimum routing table. However, this preprocessing step is memory consuming and also requires a huge amount of processing time. The memory is optimized such that each path is represented by M-bits where M is the number of available links. Also the computation of each pair paths is done in parallel using OpenMP as they are totally independent tasks. After the preprocessing phase is finished the actual Recursive Routing Table Selector is started.

The proposed decomposition algorithm for the P_{RSA} presented in Fig. 1 starts with a routing table built on top of shortest path algorithm as an initial input. Afterwards, the spectrum assignment algorithm called Longest First Fit (LFF) [21] in this work is applied to calculate the makespan related to the traffic demand matrix. The resulted makespan is fed back to the routing table selector which in turn searches for another routing table that may lead to a better makespan. This is subject to the condition that the newly selected routing table will have a lower bound less than the current makespan. This operation is repeated recursively to skip routing tables that will not lead to better lower bound. For example, if a routing table starting with an entry consisting of one or more links and this entry already consumes the number of frequency slots larger than the current makespan, then the recursive selector will skip the rest of possible routing tables that start with the same entry. The pseudo code for the recursive routing table selector is shown in Fig. 2.

The main loop lines (15-23) in Fig. 2 starts updating the lower-bound of the partially constructed routing table based on one of the possible paths that will carry the traffic between nodes *s* and *d*. This will lead to increasing the lower-bound of the partially constructed routing table. If the lower-bound of the partially constructed routing table still less than the best makespan in hand, the main loop continues adding one more entry to the routing table. This process is repeated until either the partially routing table is discarded along with all its successors because its lower bound has exceeded the current makespan, or it becomes complete routing table which means it is a candidate to be investigated using the spectrum assignment algorithm. The RRTS function stores the

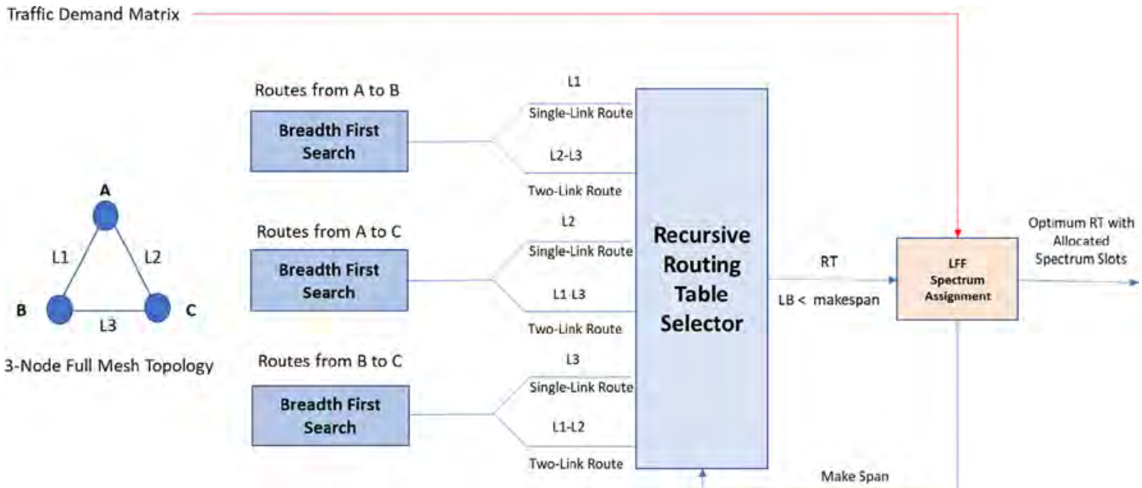

Fig. 1. P_{RSA} RSA problem decomposition example of 3-Node network.

Algorithm 1 Recursive Routing Table Selector
RRTS(s, d, R, LB, $Makespan$, K_{sd})

Input:
  s: source node, initial value is 1
  d: destination node, initial value is 2
  R: Partially Constructed Routing Table
  LB: lower bound of Partially Constructed Routing Table
  $Makespan$: best solution so far (global variable)
  $\mathbf{K_{sd}}$: All possible routes between s and d nodes
Output:
  R: Fully Constructed Routing Table

BEGIN
  if $s > |v|$ **then** {R is a complete configuration}
    Return R
  end if
  if $d > |v|$ **then** {Continue with next source node}
    $s = s + 1$
    $d = 1$
    RRTS(s, d, R ,LB**)**
  end if
  if $s = d$ **then** {Continue with next destination node}
    $d = s + 1;$
    RRTS(s, d, R, LB**)**
  end if
  //Main Recursion
  for $k = 1$; $k \le |K_{sd}|$; $k++$ **do**
    $newR$ = add path p_{sd}^{k} to R; //**Update *R***
    $newLB$ = update LB;
    if $newLB < MS$ **then**
      RRTS(s, $d + 1$, $newR$, $newLB$);
    else
      return {skip this partial constructed routing table and its successors}
    end if
  end for
END

Fig. 2. Pesdo code for Recursive Routing Table Selector.

state of the return routing table so if it is called again it will continue searching from the last processed routing table.

We emphasize that, in the worst case, the RRTS recursion may be forced to generate all, or close to all, possible routing tables and hence take exponential time to complete. However, the experimental results we present in the next section indicate that, in practice, the RRTS algorithm will need to explore only a tiny fraction of the routing space. Thus, RRTS represents a scalable solution to large RSA problems.

4. Experimental results

We have evaluated the performance of the RRTS algorithm by carrying out simulation experiments with a large number of RSA problem instances. Each problem instance is characterized by three parameters:

1. The network topology
2. the set k_{sd} of paths for each source-destination pair *(s,d)*
3. The probability distribution used to generate the traffic demand matrix.

In our evaluation study, we have used two network topologies:

- NSFNet 14-node topology shown in Fig. 3.
- GEANT2 32-node topology shown in Fig. 4.

We used DFS to generate the $|K_{sd}| = k$ shortest acyclic paths for each source–destination pair, k = 1; 2; 3, as well as all acyclic paths between each node pair, as we discussed earlier; we denote the latter as k = all in the figures we present later in this section. In all simulations we assume symmetric routing of demands, i.e., traffic from *s* to *d* takes the same path as traffic from *d* to *s*. We assume that the network supports data rates of 10, 40, 100, 400, and 1000 Gbps. For a given problem instance, we generate a random value for the traffic demand between a pair of nodes based on one of the following three distributions:

1. Uniform: each of the five rates {10; 40; 100; 400; 1000} is selected with equal probability
2. Skewed low: the rates above are selected with probability 0:30; 0:25; 0:20; 0:15, and 0:10, respectively (i.e., lower data rates have higher probability to be selected)
3. Skewed high: the five rates are selected with probability 0:10; 0:15; 0:20; 0:25, and 0:30, respectively (i.e., higher data rates have higher probability to be selected).

We assume that all traffic will use QPSK modulation and all routes will be considered regardless of the actual distance in KM of any route, although applying this constraint would reduce the problem size significantly, however we did not consider it in our experiments.

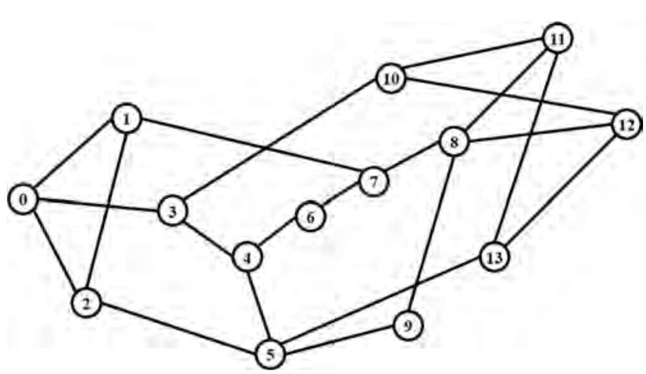

Fig. 3. The 14-node NSFNet topology.

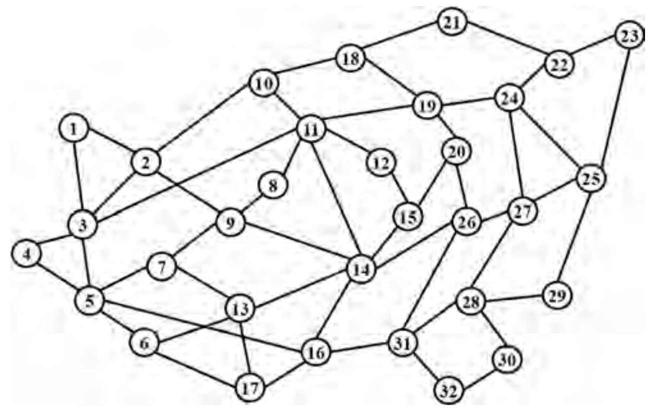

Fig. 4. The 32-node GEANT2 topology.

Each traffic distribution (normal, low skew, high skew) is generated 100 times and the experiments are repeated 100 time for each distribution and for each network.

The results shown in Fig. 5 shows the percentage of trials that RRTS was able to finish the compete search space. Other trials that failed to finish the complete search space are also listed including the percentage of the fragmentation. RRTS fails to complete the search space may be caused by one of the following reasons:

1. The heuristics Spectrum Assignment algorithm is not efficient in those trials.
2. The successors of the tables are not eliminated due to that fact that the skip condition was not satisfied.

To find which reason causes this we need to solve those trials with spectrum assignment using brute-force. However the gain of such solution is negligible as we know that the fragmented slots are less than 3.5% as showed in Fig. 5 which is very small percentage.

The main metric we consider is makespan that the decomposition algorithm returns as the (best) solution value. Let $RRTS_k$ denotes the RRTS algorithm that generates k paths for each node pair in the path computation step (Step 1), k = 1; 2; 3, all, and let SOL_k denotes the corresponding solution. This metric provides insight into the impact of the number k of alternate paths on the use of spectrum resources in the network.

Since the RRTS algorithm uses the LFF heuristic for spectrum allocation, it is important to compute a lower bound in order to evaluate the quality of the solution obtained by the SA algorithm. Consider a complete routing configuration R generated by RRTS within the if statement of Lines 2-4 in Fig. 2.

The lower bound $LB(R)$ given by expression (6) on this configuration is a lower bound on the optimal solution of this RSA instance. Therefore, we modify the algorithm to calculate not only the LFF solution on R, but also the lower bound $LB(R)$. Then, the lower bound LB_k for algorithm $RRTS_k$ is taken as the minimum of the lower bounds $LB(R)$ over all complete configurations generated by this algorithm. Finally, we compute the overall lower bound as: $LB = \min_k\{LB_k\}$. Ofcourse we are sure there are no other lower-bound in the complete search space less than the LB we found.

The metric we use to characterize the quality of the solution constructed by algorithm $RRTS_K$ is the ratio:

$$Q_k = \frac{SOL_k}{LB} \tag{7}$$

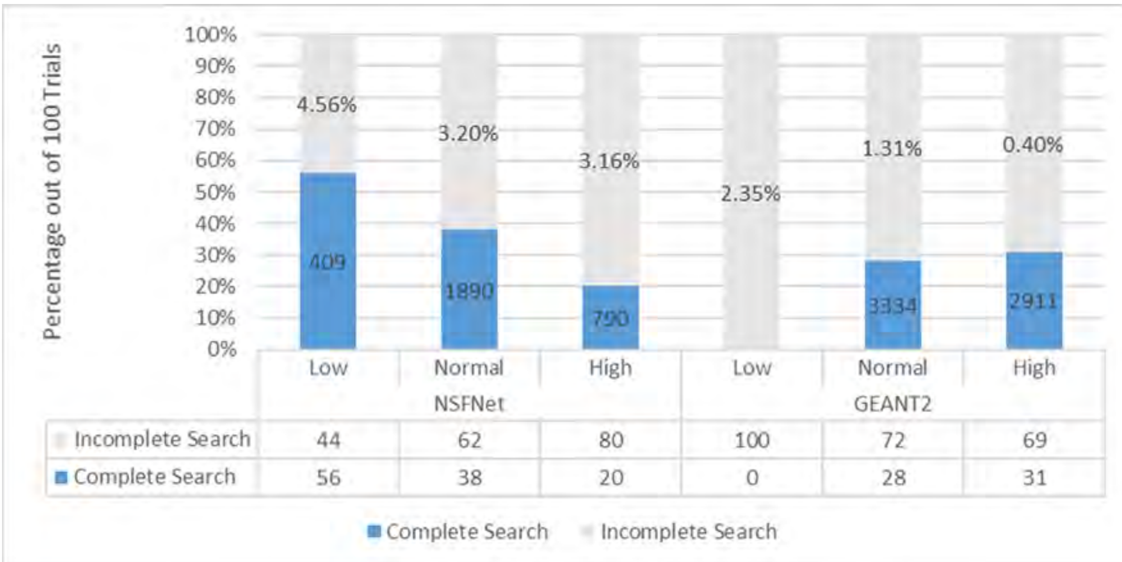

Fig. 5. RRTS Solution Performance; the value on bars corresponds either to the average computation time (for complete search bars) or fragmentation percentage (for Incomplete search).

Clearly, $Q_k \leq 1$; the closer Q_k is to 1, the better the solution, i.e., the closer it is to the optimal one.

4.1. Results and discussions

The two Figs. 6 and 7 plot the average quality ratio Q_k in expression (7) for the NSFNet and GEANT2 topologies, respectively; note that the lower bound LB in (3) is independent of the number alternative paths (K) between each pair of nodes. Each figure includes three curves, each curve representing results for problem instances with spectrum demand matrices generated by the uniform, skewed low, and skewed high distributions, respectively. Each data point in these figures is the average of 100 random problem instances generated for the stated parameters (i.e., network topology, number of alternavie paths (k) between pair, and traffic demand distribution).

As we can see, the solution quality improves (i.e., the ratio Q_k decreases) with the number alternavie paths (k) of paths increase. Note that k = 1 in the figures corresponds to spectrum assignment along the shortest paths (i.e., a single routing configuration), while as k increases the algorithm considers a significantly larger set of routing configurations and hence is able to find increasingly better solutions. This improvement in solution quality as k increases is observed for both topologies and across all three traffic distributions. For the smaller NSFNet topology, we also observe that with as few as k = 3 paths most benefits of the routing search are realized, and there is little improvement even when we consider all possible paths between each node pair. For the larger GEANT2

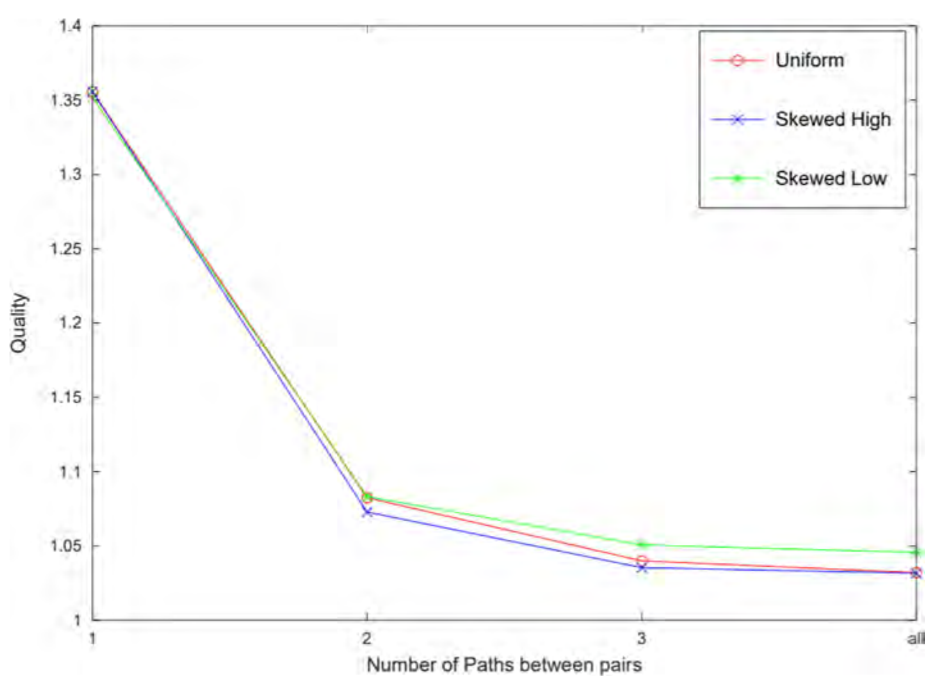

Fig. 6. Average ratio of solution quality (Q), versus the number alternative paths (k) between each pair of nodes, for NSFNet Network Topology.

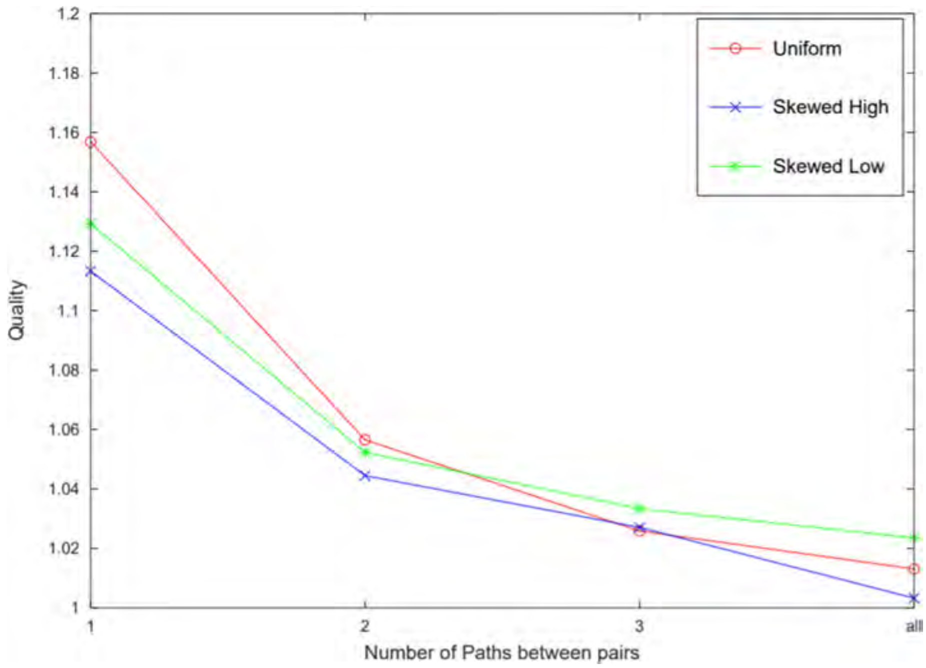

Fig. 7. Average ratio of solution quality (Q), versus the number alternative paths (k) between each pair of nodes, for GEANT2 Network Topology.

Table 1
Running time of RRTS (sec).

	K = 2			K = 3			K = all		
	Uniform	Skewed High	Skewed Low	Uniform	Skewed High	Skewed Low	Uniform	Skewed High	Skewed Low
NSFNet	1222	279	1381	1513	2231	2614	5183	5919	3398
GEANT2	12,102	13,294	13,668	12,799	13,661	13,391	11,174	10,715	12,204

network, on the other hand, there is a substantial improvement in solution quality as we move from k = 3 to all paths. Finally, we note that, while RRTS searches the entire routing space, it uses a heuristic spectrum assignment algorithm. As a result, the RRTS algorithm may not obtain the optimal solution even when it is given as input all the paths between each node pair. On the other hand, we also emphasize that the lower bound may not be achievable, and hence, the ratio Q_k for optimal solution may be strictly larger than 1.0. In any case, we observe that the solutions obtained for all paths achieve a ratio that is very close to 1.0.

Let us now turn our attention to the running time of the RRTS algorithm. We run the algorithm on the Aziz Supercomputer facility at King Abdulaziz University, which includes 492 Intel Xeon nodes comprising almost twelve thousand cores.

For each of the two topologies, we first run DFS to compute all possible paths between every pair of nodes. The path computation for each topology was run in parallel on 24 cores, and took negligible time for NSFNet and approximately 11 sec for GEANT2. Each problem instance was then run on a single core, but all 300 instances for each topology and value of k (i.e., 100 instances per traffic distribution) were run in parallel by reserving 300 cores.

Table 1 lists the running time (in seconds) of the RRTS algorithm, averaged over the 100 problem instances for the stated topology, number k of paths, and traffic distribution. It worth mentioning that K = 2 has initial makespan input calculated for the shortest path routing table, and output makespan of K = 2 was used as input makespan for K = 3, and output makespan of k = 3 was used as input makespan for K = all.

To appreciate how efficient the RRTS algorithm is, we note that for a network with N nodes and k paths per node, the number of possible routing configurations L(N, k) under the symmetric routing we consider in this paper is $O(K^{N(N-1)/2})$. In other words, just for k = 2 paths, the size of the routing space is $O(2^{91})$ for NSFNet and $O(2^{496})$ for GEANT2. Nevertheless, the RRTS algorithm was able to eliminate huge swaths of the routing space and complete the search (including the time to run the LFF spectrum assignment heuristic for each routing configuration examined) in less than two hours for NSFNet and less than four hours for GEANT2. In other words, the algorithm scales well to problem instances representative of topologies encountered in practice.

5. Concluding remarks

In this work we have made the following contributions:

- Develop a new recursive algorithm to iterate over all routing tables and find the routing table that can lead to a better makespan.
- The new recursive algorithm is converging to better solution over time even if it was not able to finish the whole search space. The converging rate is very good as it reaches a fragmentation ratio of 4.56% for NSFNet in two hours and 2.35% for GENAT2.

Our current efforts are directed to replace the spectrum assignment algorithm with a new recursive algorithm to find exact solution to the SA problem. Also, we are working on parallelizing the proposed algorithm in this paper to utilize multicore architecture in the RRTS algorithm not only in table generation part but also in the recursive part of the algorithm.

References

[1] Simmons JM. Optical network design and planning. Springer; 2014.
[2] Rouskas GN. Routing and wavelength assignment in optical WDM networks. Wiley Encycl Telecommun 2003.
[3] Dutta R, Rouskas GN, et al. A survey of virtual topology design algorithms for wavelength routed optical networks. Opt Netw Mag 2000;1(1):73–89.
[4] Jaumard B, Meyer C, Thiongane B. Comparison of ILP formulations for the RWA problem. Opt Switch Netw 2007;4(3–4):157–72.
[5] Dutta R, Rouskas GN. Traffic grooming in WDM networks: past and future. IEEE Netw 2002;16(6):46–56.
[6] Wang H, Rouskas GN. Hierarchical traffic grooming: a tutorial. Comput Netwo 2014;69:147–56.
[7] Klinkowski M, Lechowicz P, Walkowiak K. Survey of resource allocation schemes and algorithms in spectrally-spatially flexible optical networking. Opt Switch Netw 2018;27:58–78.
[8] Shirin Abkenar F, Ghaffarpour Rahbar A. Study and Analysis of Routing and Spectrum Allocation (RSA) and Routing, Modulation and Spectrum Allocation (RMSA) Algorithms in Elastic Optical Networks (EONs). Opt Switch Netw 2017;23:5–39.
[9] Zhou D, Subramaniam S. Survivability in optical networks. IEEE Netw 2000;14(6):16–23.
[10] Jaumard B, Daryalal M. Efficient spectrum utilization in large scale RWA problems. IEEE/ACM Trans Netw 2017;25(2):1263–78.
[11] Talebi S, Alam F, Katib I, Khamis M, Salama R, Rouskas GN. Spectrum management techniques for elastic optical networks: a survey. Opt Switch Netw 2014;13:34–48.
[12] Liu Z, Rouskas GN. Election algorithms for link-based ILPs and applications to RWA in mesh networks. University of Athens; 2013.
[13] Liu Z, Rouskas GN. A fast path-based ILP formulation for offline RWA in mesh optical networks. IEEE Global Communications Conference (GLOBECOM), 2012. p. 2990–5.
[14] Garey MR, Johnson DS. Computers and intractability, vol. 29. wh freeman New York; 2002.
[15] Christodoulopoulos K, Tomkos I, Varvarigos EA. Routing and spectrum allocation in OFDM-based optical networks with elastic bandwidth allocation. In: 2010 IEEE Global Telecommunications Conference GLOBECOM. p. 1–6.
[16] Jaumard B, Meyer C, Thiongane B. On column generation formulations for the RWA problem. Discret Appl Math 2009;157(6):1291–308.
[17] Dawande M, Gupta R, Naranpanawe S, Sriskandarajah C. A traffic-grooming algorithm for wavelength-routed optical networks. INFORMS J Comput 2007;19(4):565–74.
[18] Klinkowski M, Żotkiewicz M, Walkowiak K, Pióro Michałand Ruiz M, Velasco L. Solving large instances of the RSA problem in flexgrid elastic optical networks. IEEE/OSA J Opt Commun Netw 2016;8(5):320–30.
[19] IBM, "No Title."; 2010.
[20] Enoch J, Jaumard B. Towards optimal and scalable solution for routing and spectrum allocation. Electron Notes Discret Math 2018;64:335–44.
[21] Fayez M, Katib I, Rouskas GN, Faheem HM. Scheduling-inspired spectrum assignment algorithms for mesh elastic optical networks. In: Adv. Comput. Commun. Networks From Green, Mobile, Pervasive Netw. to Big Data Comput. p. 225.

Mahmoud Fayez is a PhD student and associate lecturer at Ain Shams University. I received the BSc. degree in Computer Systems from the University of Ain Shams, Egypt, in 2007, and the MSc in Computer Science from Ain Shams University, Egypt, in 2014. In 2008, I joined the Department of Computer Systems at Ain Shams university as a Teaching Assistant. I became associate lecturer in 2014. His current research interests include CUDA, MPI, OpenMP, Machine Learning, Computer Vision and Elastic Optical Network.

Iyad Katib is an Associate Professor with the Computer Science Department and the current Vice Dean and the College Council Secretary of the Faculty of Computing and Information Technology (FCIT) in King Abdulaziz University (KAU). He is also the Director of KAU High Performance Computing Center. Iyad received his Ph.D. and MS degrees in Computer Science from University of Missouri-Kansas City in 2011 and 2004, respectively. He received his BS degree in Statistics/Computer Science from King Abdul Aziz University in 1999. His current research interest is on Computer Networking and High Performance Computing.

George N. Rouskas is an Alumni Distinguished Graduate Professor with the Department of Computer Science at North Carolina State University. Since January 2014, I serve as Director of Graduate Programs for the department. I received an undergraduate degree in Computer Engineering from the National Technical University of Athens, and MS and PhD degrees in Computer Science from the Georgia Institute of Technology. During the 2000-2001 academic year, I spent a sabbatical term at Vitesse Semiconductor, Morrisville, NC. I have been an invited Professor at King Abdulaziz University, Saudi Arabia, Paris VI University, France, and University of Evry, France, on several occasions. My research interests are in the broad field of computer networking -- specifically, optical networks, Internet architectures and protocols, network design and optimization, performance modeling, and scheduling.

Tarek F. Gharib is a Professor and Chairman of Information Systems in the Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt. He received his Ph.D. degree in Theoretical Physics from the University of Ain Shams. His research interests include data mining techniques, Bioinformatics, graph and sequential data mining and information retrieval. He has published over 30 papers on data mining. He received the National Science Foundation Award in 2001.

Heba Khaled is an assistant professor at the Department of Computer Systems Faculty of Computer and Information Science Ain Shams University, Cairo, Egypt. She obtained the M.Sc. degree in Computer Science from the same faculty in 2008; the thesis title was "Analysis and design of DNA sequence alignment algorithm using parallel processing". She obtained the Ph.D. in Computer Science in August 2014 from the same faculty. The Ph.D. thesis was "Hybrid Model for solving DNA sequence Alignment using Graphics Processing Units". Her main research interests are in the areas of parallel processing and Architectures, High performance computing, Natural Language processing and Bioinformatics.

Hossam Faheem is a professor of computer systems at the faculty of computer and inforaiton sciences – Ain Shams University. He has a BSc of computer engineering in 1992, MSc of computer engineering in 1995, and PhD of computer engineering in 2000. His research interests include Parallel Processing, High Performance Computing, Networking, and Multi Agent Based Systems.