A GENETIC ALGORITHM WITH RANDOM KEYS FOR ROUTING AND WAVELENGTH ASSIGNMENT

THIAGO F. NORONHA, MAURICIO G.C. RESENDE, AND CELSO C. RIBEIRO

Abstract. The problem of routing and wavelength assignment (RWA) in wavelength division multiplexing (WDM) optical networks consists in routing a set of lightpaths and assigning a wavelength to each of them, such that lightpaths whose routes share a common fiber are assigned different wavelengths. This problem was shown to be NP-hard when the objective is to minimize the total number of wavelengths used. In this paper, we propose a genetic algorithm with random keys for routing and wavelength assignment with the goal of minimizing the number of different wavelengths used in the assignment. This algorithm extends the best heuristic in the literature by embedding it into an evolutionary framework. Computational results show that the new heuristic improves the state-of-the-art algorithms in the literature.

1. Introduction

Information is transmitted in optical networks through optical fibers as optical signals. Each link operates at a speed in the order of terabits per second, which is much faster than the currently available electronic devices for signal reception and transmission. *Wavelength division multiplexing* (WDM) allows more efficient use of the huge capacity of optical fibers, as far as it permits the simultaneous transmission of different channels along the same fiber, each of them using a different wavelength. An all-optical point-to-point connection between two nodes is called a *lightpath*. It is characterized by its route and the wavelength with which it is multiplexed. Two lightpaths may use the same wavelength, provided they do not share any common fiber. Such networks require a large number of available wavelengths, especially when wavelength conversion is not available.

Given an optical network and a set of lightpath to be established, the problem of *routing and wavelength assignment* (RWA) in WDM optical networks consists in routing the set of lightpaths and assigning a wavelength to each of them, such that lightpaths whose routes share a common fiber are assigned different wavelengths. Variants of RWA are characterized by different optimization criteria and traffic patterns, see e.g. [8, 27]. We consider the *min-RWA offline* variant, in which all lightpath requests are known beforehand and no wavelength conversion is available, i.e. a lightpath must be assigned the same wavelength on all fibers in its route. The objective is to minimize the total number of wavelengths used. This problem is also referred to as the *path coloring problem*. Erlebach and Jansen [10] showed that `min-RWA` is NP-hard.

Date: June 17, 2008.

Key words and phrases. Genetic algorithm, routing, wavelength assignment, optical networks, random keys.

AT&T Labs Research Technical Report.

The paper is organized as follows. Related work is reviewed in Section 2. The new heuristic is proposed in Section 3. Computational experiments are reported in Section 4. Concluding remarks are drawn in the last section.

2. Related work

Different heuristics have been proposed for solving `min-RWA`. Some approaches decompose the problem into two subproblems, the routing subproblem and the wavelength assignment subproblem [4, 17, 20, 23], while others tackle the two subproblems simultaneously [21, 25]. A functional classification of RWA heuristics can be found in [8].

Bannerjee and Mukherjee [4] tackled the problem in two phases. First, one route is computed for each lightpath by a randomized rounding algorithm. Then, a wavelength is assigned to each lightpath with a greedy heuristic for coloring the conflict graph. This graph is built with one vertex corresponding to each lightpath and an edge between every pair of vertices whose corresponding routes share a common fiber in the network. Hyytiä and Virtamo [17] followed the same decomposition strategy, but used different algorithms in each phase. The routes are computed by a shortest path algorithm and several heuristics are applied to solve the wavelength assignment subproblem. Their numerical results point to a tabu search [16] heuristic as the best for wavelength assignment.

Manohar, Manjunath, and Shevgaonkar [21] developed `Greedy-EDP-RWA`, the first heuristic to tackle both subproblems simultaneously. At each iteration, a subset of lightpaths is selected and routed with edge disjoint paths by the `BGAforEDP` heuristic for the *maximum edge disjoint path* (EDP) problem [19]. Then, all lightpaths in this subset are assigned the same wavelength, and the procedure is repeated with the remaining lightpaths. The authors reported that their algorithm was much faster than other algorithms in the literature and found solutions as good as those obtained by the other algorithms.

Li and Simha [20] proposed another two-phase decomposition strategy for solving `min-RWA`. First, one or more candidate routes are computed for each lightpath. Then, a precomputed route and a wavelength are assigned to each lightpath by solving an instance of the *partition coloring problem* (PCP) defined over a partitioned conflict graph. Vertices in this graph correspond to candidate routes and there is an edge between each pair of vertices whose associated routes share a common fiber. The set of vertices is partitioned such that all vertices associated with the same lightpath are placed in the same subset of the partition. The PCP consists in selecting one vertex (route) from each subset of the partition and assigning a color (wavelength) to each of the selected vertices, such that two selected vertices sharing an edge have different colors and the total number of colors used is minimum. Li and Simha [20] showed that the decision version of PCP is NP-complete. A branch-and-cut algorithm for solving the partition coloring problem was proposed in [11].

Noronha and Ribeiro [23] followed the same decomposition scheme suggested by Li and Simha [20], but proposed new algorithms for each phase. They call their scheme `2-EDR+TS-PCP`. First, at most two alternative routes are precomputed for each lightpath by an EDP-based heuristic called `2-EDR`. Then, a route (among those precomputed) and a wavelength are assigned to each lightpath by the tabu search heuristic `TS-PCP`. `TS-PCP` starts with a feasible initial solution S with D colors, provided by the greedy heuristic onestepCD [20]. Then, a new (possibly infeasible) solution S' using $D-1$ colors is built from S. Next, TS-PCP attempts to restore the feasibility of S'. If successful, the procedure is restarted from S' and attempts to improve the solution by removing another color. If a stopping criterion is satisfied and solution S' is still infeasible, the procedure halts and the best feasible solution found is returned. Computational experiments show that, when compared with the multi-start heuristic Greedy-EDP-RWA, 2-EDR+TS-PCP found better solutions within the same computation times and was also more robust and stable.

Skorin-Kapov [25] proposed the current state-of-the-art heuristics for min-RWA. Each wavelength is associated with a different copy of a bidirected graph $G = (V, A)$ that represents the physical topology of the optical network. Vertices in V and arcs in A represent the network nodes and fibers, respectively. Lightpaths that are arc-disjointly routed on the same copy of G are assigned the same wavelength. Copies of G are associated with the bins and lightpaths with the items of an instance of the *bin packing problem* [3]. Therefore, min-RWA can be reformulated as the problem of packing all the lightpath requests in a minimum number of bins.

Let $\mathcal{T}$ denote the set of lightpath requests and, for $i = 1, \dots, |\mathcal{T}|$, let *min-length*$(i)$ be the number of hops in the path with the smallest number of arcs between the endnodes of ligthpath i in G. The min-length values will only be used for sorting the lightpaths in the heuristics described next, even though the lightpaths are not necessarily routed on shortest paths. This occurs because whenever a lightpath is routed on a copy of G (or, equivalently, placed in the corresponding bin), all arcs in its route are deleted from this copy to avoid that other lightpaths use them. Therefore, the next lightpaths routed in this copy of G might be routed on a path that is not a shortest path in the original graph G.

Four min-RWA heuristics based on classical bin packing heuristics were developed in [25]: (i) FF-RWA, based on the *first fit* heuristic, (ii) BF-RWA, based on the *best fit* heuristic, (iii) FFD-RWA, based on the *first fit decreasing* heuristic, and (iv) BFD-RWA, based on the *best fit decreasing* heuristic. Computational results show that FFD-RWA and BFD-RWA outperform Greedy-EDP-RWA [21]. However, as we note below, the reported running times turned out to be high.

Noronha, Ribeiro, and Resende [22] studied algorithms and data structures for the efficient implementation of the heuristics in [25] and reevaluated their behavior on a broader set of test instances. The best results were obtained by BFD-RWA. The longest running times of the best implementation of BFD-RWA took less than three seconds, while the times reported for the same heuristic in [25] took up to eight minutes on the same instances and the same Pentium IV 2.8 GHz hardware.

The pseudo-code of BFD-RWA is presented in Figure 1. The inputs are the graph G, the set $\mathcal{T}$ of lightpath requests, a vector $\pi = [\ \pi(1), \dots, \pi(|\mathcal{T}|)\ ]$ describing the order in which the lightpaths are considered ($\pi(i) \in \{1, \dots, |\mathcal{T}|\}$ and $\pi(i) \neq \pi(j)$, for any $i, j = 1, \dots, |\mathcal{T}|$), and the value d of the maximum number of arcs in each route. This bound was introduced in [21] to avoid lightpaths routes that use a large number of arcs in a given copy of G. As suggested in [25], the maximum number of links in each route was set to be the maximum between the square root of the number of links in the network and the diameter of G, defined as the longest shortest path between any two nodes in G. The output is a set S of tuples (p_i, ω_i), for $i = 1, \dots, |\mathcal{T}|$, where p_i is the route associated with lightpath i and ω_i is the wavelength with which it is multiplexed.

BFD-RWA is applied with the vector $\pi = [\ \pi(1), \ldots, \pi(|\mathcal{T}|)\ ]$ corresponding to the lightpaths taken in non-increasing order of their min-length values, i.e., $\pi(k) = \text{argmax}\{\textit{min-length}(i) : i \in \mathcal{T} \setminus \{\pi(1), \pi(2), \ldots, \pi(k-1)\}\}$, for any $k = 1, \ldots, |\mathcal{T}|$. The intuition to use this ordering is that long lightpaths are harder to be routed and therefore should be routed first. The sets S (pairs formed by the route and the wavelength assigned to each lightpath) and Ω (copies of G or, equivalently, bins) are initialized in line 1. The lightpaths are routed one at a time and assigned a wavelength in lines 2 to 12. If there is no feasible paths available for arc-disjointly routing lightpath $\pi(i)$ using at most d arcs in any of the copies of G in Ω, then a new copy is created in line 5 and added to set Ω in line 6. The copy of the graph G in which lightpath $\pi(i)$ can be routed using the smallest number of arcs is found in line 8. In line 9, lightpath $\pi(i)$ is placed in route $p_{\pi(i)}$ associated with the shortest path between the endnodes of $\pi(i)$ in this copy of G and assigned wavelength $\omega_{\pi(i)}$. The pair $(p_{\pi(i)}, \omega_{\pi(i)})$ is added to the current partial solution in line 10 and all arcs in route $p_{\pi(i)}$ used by lightpath $\pi(i)$ are deleted from this copy of G in line 11.

```
procedure BFD-RWA($G, \mathcal{T}, d, \pi$)
1.  Set $S \leftarrow \emptyset$ and $\Omega \leftarrow \emptyset$;
2.  for $i = 1, \ldots, |\mathcal{T}|$ do
3.      if there is no (arc-disjoint) path available for routing $\pi(i)$ with less
        than $d$ arcs in any of the copies of $G$ in $\Omega$
4.      then do
5.          Create a new copy of $G$;
6.          Add to $\Omega$ the new copy of $G$;
7.      end-if
8.      Find the copy of $G$ in $\Omega$ where lightpath $\pi(i)$ can be routed with the
        smallest number of arcs;
9.      Let $p_{\pi(i)}$ be the shortest path between the endnodes of lightpath $\pi(i)$
        in this copy of $G$ and $\omega_{\pi(i)}$ be its corresponding wavelength;
10.     $S \leftarrow S \cup (p_{\pi(i)}, \omega_{\pi(i)})$;
11.     Delete all arcs in path $p_{\pi(i)}$ from this copy of $G$;
12. end-for;
13. return S;
end BFD-RWA.
```

FIGURE 1. Pseudo-code of the BFD-RWA heuristic.

3. Genetic algorithm with random keys

Since the number of lightpaths is usually much greater than the diameter of the graph, there are many lightpaths with the same min-length value. We observed in computational experiments that the way these ties are broken in line 1 of Figure 1 influences the quality of the solutions produced by BFD-RWA. Therefore, it is natural to embed BFD-RWA into a multi-start procedure that breaks ties at random and takes the best solution found over all iterations. The algorithm we call MS-BFD stops when a maximum number of iterations is reached or when a solution at least as good as a given target is found. In place of breaking ties completely at random, this paper proposes an evolutionary algorithm that *learns* how to break ties in a way that turns out to be better than random tie breaking, as we will see in Section 4.

The genetic algorithm with random keys (`GA-RWA`) is based on Bean [5] and is motivated by its successful applications to many combinatorial optimization problems [6, 7, 9, 12, 14, 13, 15]. The algorithm evolves a population of chromosomes that are used to break ties. These chromosomes are vectors of real numbers (called keys) in the range $[0, 1]$ that are randomly generated in the initial population. We use the *parameterized uniform crossover* scheme proposed in [26] to combine two parent solutions and produce an offspring solution. In this scheme the offspring inherits each of its keys from the best fit of the two parents with probability 0.7 and from the least fit parent with probability 0.3. This genetic algorithm does not make use of the standard mutation operator, where parts of the chromosomes are changed with small probability. Instead, the concept of *mutants* is used. In each generation, a fixed number of mutant solutions are introduced in the population. They are generated in the same way as the initial population. As with mutation, mutants serve the role of helping the procedure escape from local optima. The fitness of the chromosome is given by the cost of the solution found by a decoding heuristic that receives as input the random-keys vector and outputs a feasible solution with its corresponding cost.

At each new generation, the population is partitioned into two sets: TOP and $REST$. Consequently, the size of the population is $|TOP| + |REST|$. The best solutions are kept in TOP while the others are placed in $REST$. As illustrated in Figure 2, the chromosomes in TOP are copied, without change, to the population of the next generation. The new mutants are placed in set BOT. The remaining elements of the new population are obtained by crossover with one parent randomly chosen from TOP and the other from $REST$. $|REST| - |BOT|$ offspring solutions are created this way. The sizes of sets TOP, $REST$, and BOT are parameters that must be tuned.

In the genetic algorithm `GA-RWA`, there is one key associated with each lightpath. The decoding consists of two steps. First, the lightpaths are sorted in non-increasing order of the sum of their min-length and key values. Therefore, the relative order between lightpaths with the same min-length value is defined by their keys. The resulting order is used as the vector π in `BFD-RWA` (see the pseudo-code in Figure 1). The number of wavelengths found by `BFD-RWA` using this order is used as the fitness of the chromosome. The algorithm stops when a maximum elapsed time is reached or when a solution as good as a given target is found. Through this evolutionary process, `GA-RWA` identifies the relationships between keys and good solutions, converging to better solutions faster than `MS-BFD`, as we will see in the next section.

4. Computational experiments

Three sets of test instances were used in the computational experiments. Sets Y and Z were proposed in [22], while set W is a collection of the most studied realistic instances in the literature. All network topologies are connected and each link corresponds to a pair of bidirected fibers. The traffic matrices are asymmetric, i.e. there might be a lightpath request from a node i to a node j and not from j to i. A description of each problem set is presented below.

Instances in set Y were randomly generated with 100 nodes. The probability P_e that there is a link (a pair of bidirected arcs) between a pair of nodes is equal to 0.03, 0.04, or 0.05, and the probability P_l that there is a lightpath request between a pair of nodes is equal to 0.2, 0.4, 0.6, 0.8, or 1.0. The nodes in networks with P_e

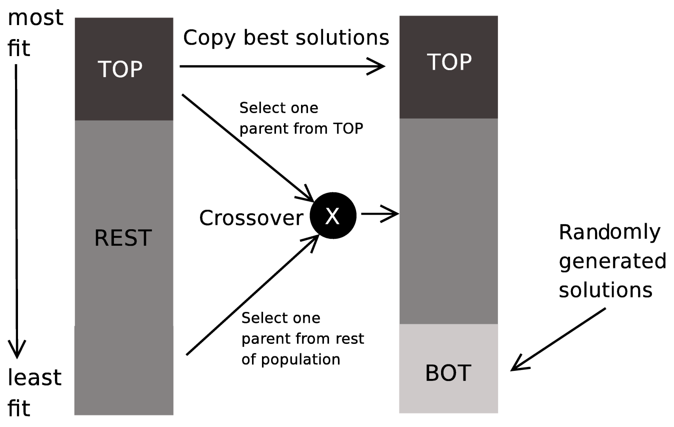
most fit
least fit
TOP
REST
Copy best solutions
Select one parent from TOP
Crossover
X
Select one parent from rest of population
TOP
BOT
Randomly generated solutions

FIGURE 2. Illustration of the transitional process between consecutive generations of the genetic algorithm with random keys. Current population is sorted from best to worst fitness. The best fit individuals are place in set TOP while the other are placed in set $REST$. Individuals in set TOP are copied unchanged to next population. Mutant individuals are generated at random in a similar way as the initial population is generated and are placed in set BOT in the next population The remaining individuals of the next population are generated by repeatedly applying the crossover operator to randomly selected individuals from the set TOP in the current population and randomly selected individuals from the set $REST$ in the current population.

equal to 0.04 or 0.05 have degree at least 2. The diameters of the networks with P_e equal to 0.05, 0.04, and 0.03 are, respectively, 5, 6, and 7. Fifteen groups with five instances each were randomly generated combining each possible value of P_e and P_l.

The set Z of instances is based on $n \times m$ grids embedded on the torus. Each node is connected only to its nearest four nodes. Figure 3 gives the example of a 3×4 grid. Five grid networks with approximately 100 nodes (10×10, 8×13, 6×17, 5×20, 4×25) were generated. For each of them, five traffic matrices were randomly generated with probability P_l equal to 0.2, 0.4, 0.6, 0.8, or 1.0 that there is a lightpath request between a pair of nodes.

Set W is a collection of the most studied realistic instances in the literature. Their topologies and traffic matrices follow the patterns and sizes of real telecommunication networks. The topology of the `Finland` network was obtained from [17] and its traffic matrix was the same used in [23]. Instances `ATT` and `ATT2` are the same used in [22]. The networks `EON`, `NSF`, and `NSF2` and their respectively traffic matrices were downloaded from [18].

Heuristics `2-EDR+TSPCP` [23], `BFD-RWA` [25], `MS-BFD`, and `GA-RWA` were implemented in C++ and compiled with the `gnu gcc` compiler version 4.0.3 with no compiler code optimization. One should expect running times to improve by activating code optimization. The first and third batch of experiments were performed on a 3.4 GHz Pentium IV, while the second batch was performed on a 1.5 GHz

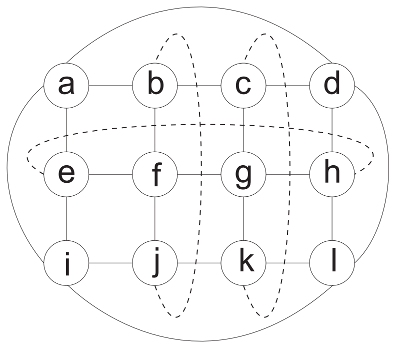

FIGURE 3. Example of a 3×4 grid topology.

TABLE 1. Experimental results with BFD-RWA and MS-BFD for test set Y.

				BFD-RWA		MS-BFD	
Group	nodes	P_e	P_l	gap (%)	time (s)	gap (%)	time (s)
Y.3.20	100	0.03	0.2	14.1	0.1	11.8	73.3
Y.4.20	100	0.04	0.2	14.7	0.1	11.1	59.7
Y.5.20	100	0.05	0.2	7.6	0.1	3.2	48.6
Y.3.40	100	0.03	0.4	11.3	0.2	9.3	174.8
Y.4.40	100	0.04	0.4	10.6	0.2	8.8	142.3
Y.5.40	100	0.05	0.4	5.0	0.1	2.7	108.0
Y.3.60	100	0.03	0.6	8.6	0.3	7.3	305.4
Y.4.60	100	0.04	0.6	8.9	0.3	7.8	245.2
Y.5.60	100	0.05	0.6	5.6	0.2	2.9	180.9
Y.3.80	100	0.03	0.8	7.3	0.5	6.1	455.5
Y.4.80	100	0.04	0.8	8.0	0.4	7.0	368.1
Y.5.80	100	0.05	0.8	5.3	0.3	3.4	263.0
Y.3.100	100	0.03	1.0	6.7	0.7	6.0	630.2
Y.4.100	100	0.04	1.0	8.0	0.5	6.9	500.7
Y.5.100	100	0.05	1.0	4.0	0.4	3.0	359.9
			Average:	8.4	0.3	6.5	261.0

Intel Itanium 2. CPU times are reported in seconds. Solution quality is displayed as the relative gap [UB-LB]/LB between the cost UB of the solution provided by the heuristic and a lower bound LB for the optimal cost, which is calculated as suggested in [4].

The first batch of experiments addresses the performances of the greedy heuristic BFD-RWA and of the multi-start heuristic MS-BFD for all the 112 instances in sets Y, Z, and W. The stopping criterion of MS-BFD was set to 1000 iterations.

Numerical results for set Y are reported in Table 1. The first four columns give the name, the number of nodes, the value of P_e, and the value of P_l for each

Table 2. Experimental results with `BFD-RWA` and `MS-BFD` for test set Z.

				`BFD-RWA`		`MS-BFD`	
Group	nodes	degree	P_l	gap (%)	time (s)	gap (%)	time (s)
`Z.4`×25.20	100	4	0.2	4.5	0.1	3.0	99.9
`Z.5`×20.20	100	4	0.2	3.3	0.1	1.9	84.1
`Z.6`×17.20	102	4	0.2	6.8	0.1	4.5	73.2
`Z.8`×13.20	104	4	0.2	9.1	0.1	9.1	57.4
`Z.10`×10.20	100	4	0.2	19.3	0.1	18.5	46.2
`Z.4`×25.40	100	4	0.4	3.3	0.2	2.4	218.1
`Z.5`×20.40	100	4	0.4	4.4	0.2	3.0	184.8
`Z.6`×17.40	102	4	0.4	5.7	0.2	4.5	160.8
`Z.8`×13.40	104	4	0.4	7.9	0.1	6.3	124.4
`Z.10`×10.40	100	4	0.4	16.5	0.1	15.7	97.1
`Z.4`×25.60	100	4	0.6	3.1	0.4	2.1	371.1
`Z.5`×20.60	100	4	0.6	3.0	0.3	2.6	310.8
`Z.6`×17.60	102	4	0.6	4.8	0.3	3.9	272.2
`Z.8`×13.60	104	4	0.6	6.7	0.2	5.2	207.6
`Z.10`×10.60	100	4	0.6	15.6	0.2	14.3	159.2
`Z.4`×25.80	100	4	0.8	2.3	0.6	1.2	529.3
`Z.5`×20.80	100	4	0.8	2.9	0.5	2.0	446.5
`Z.6`×17.80	102	4	0.8	3.7	0.4	3.2	394.0
`Z.8`×13.80	104	4	0.8	5.3	0.3	4.2	300.3
`Z.10`×10.80	100	4	0.8	13.6	0.3	12.6	225.7
`Z.4`×25.100	100	4	1.0	2.7	0.7	2.2	703.2
`Z.5`×20.100	100	4	1.0	3.2	0.6	2.8	596.6
`Z.6`×17.100	102	4	1.0	3.7	0.6	3.2	526.9
`Z.8`×13.100	104	4	1.0	4.9	0.5	4.2	416.1
`Z.10`×10.100	100	4	1.0	14.7	0.3	13.5	295.1
			Average:	6.8	0.3	5.8	276.0

group of five instances. The next two columns present the average gaps and the average execution times of `BFD-RWA` over five runs for each of the five instances, with different seeds for the random number generator [24]. The same results are reported for `MS-BFD` in the last two columns. We observe that, as expected, `MS-BFD` improved solution quality with respect to `BFD-RWA`. The average gap obtained with `MS-BSD` was only 6.5%, while the gap obtained with `BFD-RWA` was 8.4%. Moreover, `MS-BFD` provided optimality certificates (i.e., solutions for which the lower and upper bounds match) for 36 of the 75 instances in set Y, while `BFD-RWA` found optimal solutions for only 24 of them.

Table 2 presents the numerical results for set Z. The first four columns give the name, the number of nodes, the degree of the nodes, and the value of P_l for each instance. The next two columns present the average gaps and the average execution times over five runs of `BFD-RWA`. The same information is reported for `MS-BFD` in the last two columns. Although the average solution gaps for set Z were smaller

TABLE 3. Experimental results with BFD-RWA for test set W.

			Lightpaths		BFD-RWA		MS-BFD	
Instance	nodes	links	total	max.	gap (%)	time (s)	gap (%)	time (s)
Finland	31	51	930	1	3.0	0.0	2.2	9.6
EON	20	39	374	2	0.0	0.0	0.0	2.1
ATT	90	137	359	5	32.0	0.0	24.0	19.8
ATT2	71	175	4456	34	2.1	0.2	0.7	140.8
NSF.1	14	21	284	3	6.4	0.0	4.5	0.9
NSF.3	14	21	258	3	8.2	0.0	4.5	0.8
NSF.12	14	21	551	6	8.9	0.0	4.7	1.8
NSF.48	14	21	547	6	3.4	0.0	2.0	1.8
NSF2.1	14	22	284	3	5.7	0.0	0.0	0.8
NSF2.3	14	22	258	3	6.7	0.0	0.0	0.8
NSF2.12	14	22	551	6	6.3	0.0	2.9	1.7
NSF2.48	14	22	547	6	1.5	0.0	0.0	1.7
				Average:	7.0	0.0	3.8	15.2

than those for set Y, no optimality certificate was obtained by either heuristic due to the high symmetry of the networks in this set. For this set, the average improvement in solution quality observed for MS-BFD with respect to BFD-RWA was only 1%. However, we highlight the fact that for most min-RWA instances, while it can be easy to find a solution with D wavelengths (for some $D > 0$), it can be often difficult to find one using $D - 1$ wavelengths. Therefore, any improvement in the number of wavelengths can be significant.

The numerical results for instance set W are reported in Table 3. The first three columns give the name, the number of nodes, and the number of links for each instance, while the next two columns give the total number of lightpath requests of the instance and the maximum number of lightpath requests from the same node. The sixth and seventh columns display the average gaps and the average computation times over five runs of BFD-RWA, respectively. The same data is reported for MS-BFD in the last two columns. For this set of instances, the solution gaps obtained with MS-BFD were almost half as large as those found by BFD-RWA. Furthermore, MS-BFD provided optimality certificates for six out of the twelve instances (EON, ATT2, NSF.48, NSF2.1, NSF2.3, and NSF2.48), while BFD-RWA found optimal solutions only for two of them (EON and NSF2.48).

The second batch of experiments evaluates and compares the performances of algorithms MS-BFD and GA-RWA. Six versions of algorithm GA-RWA were evaluated, with different values for the population size and the number of chromosomes in sets TOP, $REST$, and BOT. The three first versions (V1, V2, and V3) have their population size equal to n, while the other versions (V4, V5, and V6) have their population size equal to $2 \times n$, where n is the number of nodes in the network. Versions V1 and V4 have $|TOP| < |BOT|$ (with $|TOP| = 0.01 \times n$ and $|BOT| = 0.02 \times n$), while V2 and V5 have $|TOP| > |BOT|$ (with $|TOP| = 0.25 \times n$ and

$|BOT| = 0.05 \times n$). The other two versions V3 and V6 have $|TOP| = |BOT| = 0.15 \times n$.

For the next experiments, a subset of the hardest among the 112 instances studied were selected as follows. First, we increased the stopping criterion of MS-BFD to 10,000 iterations and performed five runs for each instance with different seeds for the random number generator. The cost of the best solution found in the five runs was set as the target cost. Then, we selected the instances for which MS-BFD could not find a solution at least as good as the target within the first 1,000 iterations. Thirty instances were selected: 14 from set Y, ten from set Z, and six from set W. We notice that the remaining instances are not necessarily easy just because MS-BFD could find a solution at least as good as the target within the first 1,000 iterations, since this target could still be far from the cost of an optimal solution.

MS-BFD and the six versions of GA-RWA were run 200 times on each instance with different seeds for the random number generator. The heuristics were made to stop whenever a solution with cost smaller than or equal to the target was found. The computational results are presented in Table 4. The first column presents the name of the instance. The next column gives the gap (target $-$ LB)/LB. The third column reports the average time for MS-BFD to find a solution at least as good as the target. The average times to reach the target for the six versions of GA-RWA are displayed in the last six columns as a percent deviation from the corresponding MS-BFD time. Among the six versions of GA-RWA, V2 resulted in the best choice of parameters. This variant improved the performance of MS-BFD by 23% on average. For the hardest instance (Z.4x25.60), MS-BFD took on average 13,583 seconds to reach the target, while GA-RWA took only 60% of this time. Among the 30 instances tested, BFD-RWA was faster than V2 only on two instances (Z.10x10.20 and Finland). Considering only the set W of realistic instances, the time-to-target values of GA-RWA were on average approximately two thirds of those of MS-RWA.

The same results can be observed in Figures 4 to 6, where we plotted the empirical probability distributions of the time-to-target-solution-value observed for MS-BFD and GA-RWA for six instances from those in Table 4. To plot the empirical distribution for both heuristics, we followed the methodology described in [1, 2]. We associated with the i-th smallest running time t_i a probability $p_i = (i - \frac{1}{2})/200$, and plotted the points $z_i = (t_i, p_i)$, for $i = 1, \ldots, 200$. The more to the left is a plot, the better is the algorithm corresponding to it. We observe that GA-RWA outperforms MS-BFD for all the instances plotted in Figures 4 to 6 except Z.10x10.20 (Figures 5b). However, the plots are very close for this instance, while GA-RWA is much better than MS-BFD for the others.

The last experiments address the performances of algorithms 2-EDR+TS-PCP and GA-RWA with a time limit of 10 minutes on the computation time. For each instance, Table 5 displays its name, the lower bound for the cost of the optimal solution, and the minimum, average, and maximum solution gaps obtained with 2-EDR+TS-PCP and GA-RWA over five runs with different seeds for the random number generator. The average gaps observed with GA-RWA were smaller than those with 2-EDR+TS-PCP for 29 out of the 30 instances. The average gap over all instances in Table 5 is only 9.3% for GA-RWA, which almost doubles to 17.1% in the case of 2-EDR+TS-PCP. GA-RWA converges faster and finds better solutions than 2-EDR+TS-PCP, specially on the largest instances. This is due to the fact that the number of nodes in the conflict graph that must be colored by TS-PCP grows with the number of lightpaths

Table 4. Time-to-target experiment for MS-BFD and different versions of GA-RWA.

Instance	target gap (%)	MS-BFD time (s)	GA-RWA/MS-BFD (%) V1	V2	V3	V4	V5	V6
Y.4.20.4	5.3	118.0	97.2	96.4	99.3	90.5	91.0	90.4
Y.3.40.5	11.3	2913.4	86.8	75.3	81.7	84.0	90.8	81.7
Y.3.60.5	11.7	1307.4	92.8	91.8	95.0	98.3	88.6	95.8
Y.4.60.5	18.4	233.6	96.7	97.2	103.0	86.9	103.6	100.6
Y.5.60.1	9.1	2262.7	90.3	73.6	80.3	86.9	68.8	74.2
Y.3.80.1	15.1	3036.2	87.9	78.3	102.8	97.3	90.2	93.0
Y.3.80.5	8.7	632.4	89.4	84.3	87.3	95.3	87.9	99.9
Y.4.80.1	55.3	194.1	75.8	78.3	78.9	89.0	88.1	85.8
Y.4.80.5	15.4	981.1	73.4	62.4	84.7	82.9	88.0	83.5
Y.5.80.1	9.3	3066.4	62.8	41.8	59.0	52.2	42.1	50.5
Y.5.80.2	0.0	8538.7	66.8	53.1	60.8	63.1	45.4	59.8
Y.4.100.1	18.4	238.3	93.8	89.1	98.9	75.8	80.7	84.3
Y.5.100.1	5.5	754.7	81.5	80.1	77.4	86.8	79.7	81.2
Y.5.100.2	1.4	1330.4	62.2	54.3	64.7	64.7	62.9	67.2
Z.10x10.20	14.8	329.6	120.5	118.5	119.0	110.6	102.2	98.2
Z.6x17.40	3.6	388.8	104.3	82.9	98.2	89.0	83.8	95.1
Z.4x25.60	1.6	13583.2	65.8	59.8	70.3	67.3	68.1	65.8
Z.10x10.60	13.0	1282.3	90.1	94.1	95.7	91.4	90.6	92.9
Z.4x25.80	1.2	739.9	76.4	72.7	78.2	75.3	86.5	82.7
Z.5x20.80	2.0	145.6	81.3	77.1	86.6	89.6	91.7	87.2
Z.6x17.80	2.9	422.8	69.5	72.5	70.9	76.8	73.7	75.2
Z.8x13.80	3.9	604.9	94.1	76.4	84.6	91.5	78.6	98.6
Z.10x10.80	11.7	6522.1	89.7	94.7	94.1	90.0	94.2	83.6
Z.5x20.100	2.4	3181.5	101.4	96.6	89.4	102.3	105.9	112.5
Finland	0.0	237.0	89.3	101.7	100.0	94.7	98.6	95.2
ATT	20.0	24.9	74.9	64.8	63.6	65.7	63.4	65.9
ATT2	0.0	1584.6	62.2	54.5	63.7	72.4	53.1	73.0
NSF.3	0.0	355.7	110.5	78.8	86.3	81.9	78.0	87.7
NSF.12	2.6	32.8	97.2	62.6	70.4	57.4	50.3	58.1
NSF2.12	0.0	1370.3	83.5	46.4	64.0	52.3	31.9	42.9
Average:	8.8	1880.4	85.6	77.0	83.6	82.1	78.6	82.1

requests. Instances in sets Y and Z have up to 9,900 lightpath requests. Since procedure 2-EDR selects up to two alternative routes for each request, the conflict graphs that must be colored by heuristic TS-PCP have up to 19,800 nodes. If we consider the average gaps on the instances from set W, which are much smaller than those in sets Y and Z, the average for 2-EDR+TS-PCP is 4.5%, while the same value for GA-RWA is 4.0%.

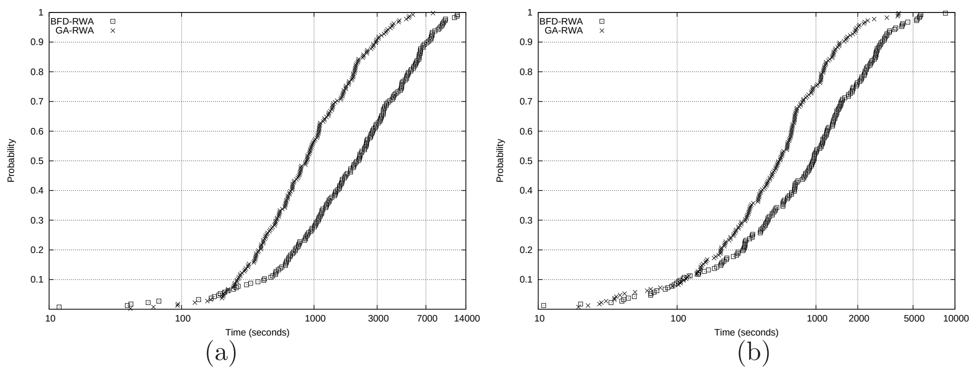

FIGURE 4. Time-to-target plots for two instances in set Y: (a) instance `Y.5.80.1` with target gap equal to 9.3%, and (b) instance `Y.5.100.2` with target gap equal to 1.4%.

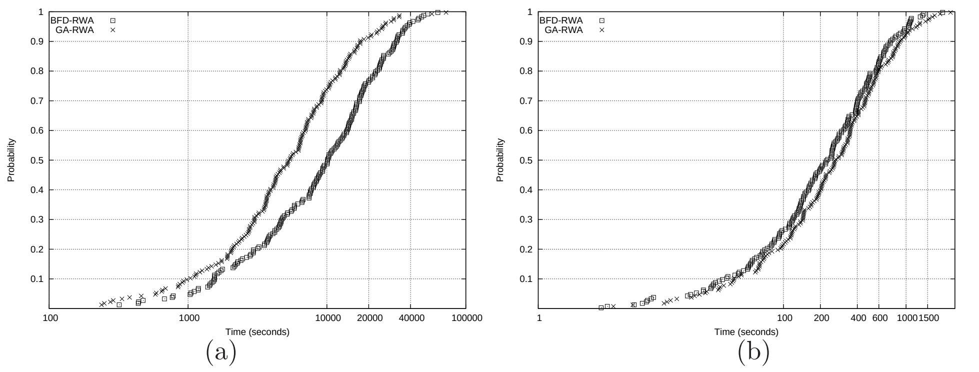

FIGURE 5. Time-to-target plots for two instances in set Z: (a) instance `Z.4x25.60` with target gap equal to 1.6%, and (b) instance `Z.10x10.20` with target gap equal to 14.8%.

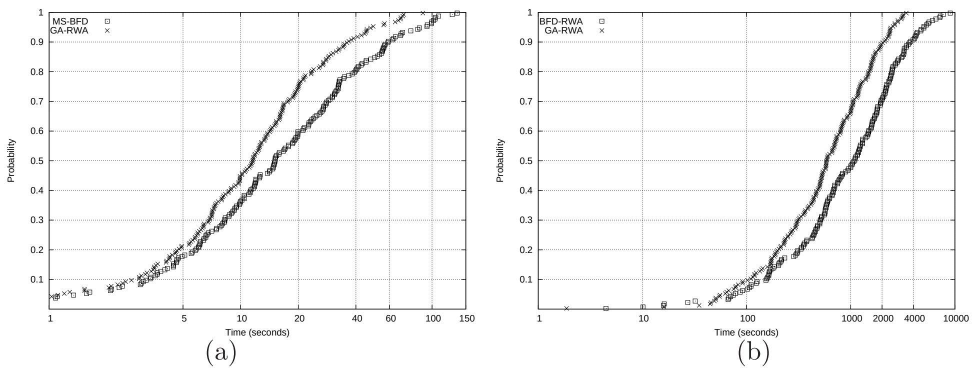

FIGURE 6. Time-to-target plots for two instances in set W: (a) instance `ATT` with target gap equal to 20.0%, and (b) instance `ATT2` with target gap equal to 0.0%.

Table 5. Solution gaps with GA-RWA and 2-EDR+TS-PCP within 10 minutes of processing time.

		2-EDR+TS-PCP			GA-RWA		
Instance	LB	min (%)	avg (%)	max (%)	min (%)	avg (%)	max (%)
Y.4.20.4	19	10.5	13.7	15.8	5.3	6.3	10.5
Y.3.40.5	53	13.2	14.7	15.1	11.3	12.8	13.2
Y.3.60.5	77	13.0	15.1	16.9	11.7	12.5	13.0
Y.4.60.5	49	24.5	25.3	26.5	18.4	18.4	18.4
Y.5.60.1	33	18.2	21.2	24.2	9.1	9.7	12.1
Y.3.80.1	106	17.9	18.7	20.8	15.1	15.5	16.0
Y.3.80.5	104	12.5	12.7	13.5	8.7	8.8	9.6
Y.4.80.1	47	63.8	65.1	68.1	55.3	55.3	55.3
Y.4.80.5	65	21.5	22.2	23.1	15.4	16.0	16.9
Y.5.80.1	43	20.9	21.9	23.3	9.3	11.2	11.6
Y.5.80.2	59	6.8	8.1	8.5	1.7	1.7	1.7
Y.4.100.1	76	38.2	40.0	43.4	18.4	18.4	18.4
Y.5.100.1	55	23.6	30.5	36.4	5.5	5.5	5.5
*Y.5.100.*2	73	16.4	21.1	30.1	1.4	1.6	2.7
Z.10x10.20	27	22.2	25.2	25.9	14.8	15.6	18.5
Z.6x17.40	84	7.1	7.6	8.3	3.6	4.0	4.8
Z.4x25.60	192	5.7	5.9	6.3	1.6	2.0	2.1
Z.10x10.60	77	20.8	21.3	22.1	13.0	13.2	14.3
Z.4x25.80	257	9.7	10.5	11.3	1.2	1.3	1.6
Z.5x20.80	205	14.6	15.5	17.6	2.0	2.0	2.0
Z.6x17.80	171	15.2	16.8	19.3	2.9	3.0	3.5
Z.8x13.80	129	14.7	15.8	17.1	3.9	3.9	3.9
Z.10x10.80	103	26.2	27.6	29.1	11.7	12.4	12.6
Z.5x20.100	250	2.8	8.6	15.6	2.8	2.8	2.8
Finland	46	2.2	2.2	2.2	0.0	0.4	2.2
ATT	20	10.0	11.0	15.0	20.0	20.0	20.0
ATT2	113	0.9	1.1	1.8	0.0	0.0	0.0
NSF.3	22	4.5	4.5	4.5	0.0	0.9	4.5
NSF.12	38	2.6	4.2	7.9	2.6	2.6	2.6
NSF2.12	35	2.9	4.0	5.7	0.0	0.6	2.9
	Average:	15.4	17.1	19.2	8.9	9.3	10.1

5. Concluding remarks

We proposed a simple, robust, and efficient genetic algorithm for min-RWA. It extends the best heuristic in the literature by embedding it into an evolutionary framework. The numerical results showed that the new heuristic improved the state-of-the-art heuristics in the literature. They also showed the robustness of the genetic algorithm, since all versions of GA-RWA (using different parameter settings) obtained good and similar results.

Despite the fact that for most instances of `min-RWA` it may be very difficult to find a solution with $D - 1$ wavelengths, our algorithm was able to improve the solutions obtained by other approaches even when they were close to the optimal solution. The average solution gap observed with `GA-RWA` was almost one half of that with `2-EDR+TS-PCP`, while the average CPU time was 77% of that of `MS-BFD`. In addition, our heuristic and the lower bound of [4] made it possible to provide optimality certificates for 46 out of the 112 test instances.

References

[1] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. Probability distribution of solution time in GRASP: An experimental investigation. *Journal of Heuristics*, 8:343–373, 2002.

[2] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. TTTPLOTS: A Perl program to create time-to-target plots. *Optimization Letters*, 1:355–366, 2007.

[3] A.C.F. Alvim, C.C. Ribeiro, F. Glover, and D.J. Aloise. A hybrid improvement heuristic for the one-dimensional bin packing problem. *Journal of Heuristics*, 10:205–229, 2004.

[4] D. Bannerjee and B. Mukherjee. Practical approach for routing and wavelength assignment in large wavelength routed optical networks. *IEEE Journal on Selected Areas in Communications*, 14:903–908, 1995.

[5] J.C. Bean. Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 2:154–160, 1994.

[6] L.S. Buriol, M.G.C. Resende, Ribeiro C.C., and M. Thorup. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. *Networks*, 46:36–56, 2005.

[7] L.S. Buriol, M.G.C. Resende, and M. Thorup. Survivable IP network design with OSPF routing. *Networks*, 49:51–64, 2007.

[8] J.S. Choi, N. Golmie, F. Lapeyrere, F. Mouveaux, and D. Su. A functional classification of routing and wavelength assignment schemes in DWDM networks: Static case. In *Proceedings of the 7th International Conference on Optical Communication and Networks*, pages 1109–1115, Paris, 2000.

[9] M. Ericsson, M.G.C. Resende, and P.M. Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. *Journal of Combinatorial Optimization*, 6:299–333, 2002.

[10] T. Erlebach and K. Jansen. The complexity of path coloring and call scheduling. *Theoretical Computer Science*, 255:33–50, 2001.

[11] Y. Frota, T.F. Noronha, N. Maculan, and C.C. Ribeiro. A branch-and-cut algorithm for partition coloring. In *Proceedings of The International Network Optimization Conference*, Spa, 2007.

[12] J.F. Gonçalves, J.J.M. Mendes, and M.G.C. Resende. A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research*, 167:77–95, 2005.

[13] J.F. Gonçalves, J.J.M. Mendes, and M.G.C. Resende. A genetic algorithm for the resource constrained multi-project scheduling problem. *European Journal of Operational Research*, 189:1171–1190, 2008.

[14] J.F. Gonçalves, J.J.M. Mendes, and M.G.C. Resende. A random key based genetic algorithm for the resource constrained project scheduling problems. *Computers and Operations Research*, 36:92–109, 2009.

[15] J.F. Gonçalves and M.G.C. Resende. An evolutionary algorithm for manufacturing cell formation. *Computers and Industrial Engineering*, 47:247–273, 2004.

[16] A. Hertz and D. de Werra. Using tabu search techniques for graph coloring. *Computing*, 39:345–351, 1987.

[17] E. Hyytia and J. Virtamo. Wavelength assignment and routing in WDM networks. In *Fourteenth Nordic Teletraffic Seminar*, pages 31–40, Copenhagen, 1998.

[18] B. Jaumard. Network and traffic data sets for optical network optimization. Online publication in `http://users.encs.concordia.ca/~bjaumard`, last visited on January 3th, 2008.

[19] J.M. Kleinberg. *Approximation algorithms for disjoint paths problems.* PhD thesis, MIT, Cambridge, 1996.

[20] G. Li and R. Simha. The partition coloring problem and its application to wavelength routing and assignment. In *Proceedings of the First Workshop on Optical Networks*, Dallas, 2000.

[21] P. Manohar, D. Manjunath, and R.K. Shevgaonkar. Routing and wavelength assignment in optical networks from edge disjoint path algorithms. *IEEE Communications Letters*, 5:211–213, 2002.

[22] T.F. Noronha, M.G.C. Resende, and C.C. Ribeiro. Efficient implementation of heuristics for routing and wavelength assignment. In C.C. McGeoch, editor, *Proceedings of the 7th International Workshop on Experimental Algorithms*, volume 5038 of *Lecture Notes in Computer Science*, pages 169–180. Springer, Provincetown, Mass., 2008.

[23] T.F. Noronha and C.C. Ribeiro. Routing and wavelength assingn by partition coloring. *European Journal of Operational Research*, 171:797–810, 2006.

[24] L. Schrage. A more portable Fortran random number generator. *ACM Transactions on Mathematical Software*, 5:132–138, 1979.

[25] N. Skorin-Kapov. Routing and wavelength assignment in optical networks using bin packing based algorithms. *European Journal of Operational Research*, 177:1167–1179, 2007.

[26] W. Spears and K. deJong. On the virtues of parameterized uniform crossover. In R. Belew and L. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 230–236, San Mateo, 1991. Morgan Kaufman.

[27] H. Zang, J.P. Jue, and B. Mukherjee. A review of routing and wavelength assignment approaches for wavelength-routed optical wdm networks. *Optical Networks Magazine*, 1:47–60, 2000.

(T.F. Noronha) Department of Computer Science, Catholic University of Rio de Janeiro, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ 22453-900, Brazil
E-mail address: `tfn@inf.puc-rio.br`

(M.G.C. Resende) Algorithms and Optimization Research Department, AT&T Labs Research, 180 Park Avenue, Room C241, Florham Park, NJ 07932 USA.
E-mail address, M.G.C. Resende: `mgcr@research.att.com`

(C.C. Ribeiro) Department of Computer Science, Universidade Federal Fluminense, Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil
E-mail address: `celso@ic.uff.br`