Improved Construction Heuristics and Iterated Local Search for the Routing and Wavelength Assignment Problem

Kerstin Bauer[1], Thomas Fischer[1], Sven O. Krumke[2], Katharina Gerhardt[2], Stephan Westphal[2], and Peter Merz[1]

[1] Department of Computer Science
University of Kaiserslautern, Germany
{k_bauer,fischer,pmerz}@informatik.uni-kl.de
[2] Department of Mathematics
University of Kaiserslautern, Germany
{krumke,gerhardt,westphal}@mathematik.uni-kl.de

Abstract. This paper deals with the design of improved construction heuristics and iterated local search for the Routing and Wavelength Assignment problem (RWA). Given a physical network and a set of communication requests, the static RWA deals with the problem of assigning suitable paths and wavelengths to the requests. We introduce benchmark instances from the SND library to the RWA and argue that these instances are more challenging than previously used random instances. We analyze the properties of several instances in detail and propose an improved construction heuristic to handle 'problematic' instances. Our iterated local search finds the optimum for most instances.

1 Introduction

The *Routing and Wavelength Assignment* problem (RWA) deals with *Wavelength Division Multiplexed* (WDM) optical networks, where communication requests between nodes in a network have to be fulfilled by routing them on optical fiber links with a given capacity. Chlamtac *et al.* [1] showed the static RWA in general networks to be NP-complete since it contains the graph-coloring problem.

The problem is defined as follows: Given is a graph $G(V, E, W)$ with nodes V, arcs E and wavelengths W. An arc $e \in E$ is an optical fiber link in the physical network, where each wavelength $\lambda \in W$ is eligible. A request $r_i = (v_i^{\mathrm{s}}, v_i^{\mathrm{t}}, d_i)$ connects nodes v_i^{s} and v_i^{t} having a demand of $d_i \in \mathbb{N}^+$. For each unit of demand a lightpath between the request's endpoints has to be established. A *lightpath* is an optical path between two nodes created by the allocation of the same wavelength throughout the path of optical fiber links providing a 'circuit-switched' interconnection. Lightpaths have to fulfill two constraints: The *wavelength conflict constraint* defines that each wavelength on a physical link is used by at most one lightpath at the same time. The *wavelength continuity constraint* requires a lightpath to use the same wavelength on each link. There are problem variants

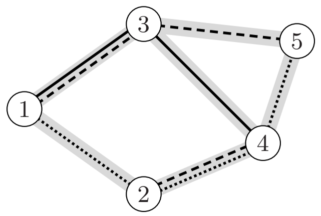


Fig. 1. Example for the RWA, using three wavelengths (different line styles) to route the requests $\{(1,5,2),(1,4,1),(2,4,1)\}$. The first request uses the dashed lightpath $\langle 1,3,5\rangle$ and dotted path $\langle 1,2,4,5\rangle$, the second request uses the solid path $\langle 1,3,4\rangle$ and the last request uses the dashed path $\langle 2,4\rangle$. Physical links are shaded in light gray.

relaxing these constraints: The first constraint can be relaxed by *multiple-fiber links* [2], the latter constraint can be relaxed by introducing *wavelength converters* [3] which change the wavelength of a lightpath at selected nodes.

The RWA can be seen as a *static* (*static lightpath establishment*, SLE) or a *dynamic* (*dynamic lightpath establishment*, DLE) problem. Furthermore, there are different types of *cost functions* available for the RWA. In the static case, a set of requests to be routed in parallel is known *a priori* and the objective is to find lightpaths for all requests minimizing the number of used wavelengths. In the dynamic case, time-bounded requests turn up over time, the routing has to be decided on-line, and the objective is to maximize the number of routed requests. In this paper, we focus on the static minimization problem.

First, we present related work on the static RWA and iterated local search. In Sec. 2 we give a formulation for the static RWA's lower bounds. Section 3 describes the used benchmark instances and our improved construction heuristics. In Sec. 4 we discuss a local search for the RWA, embedded into our iterated local search in Sec. 5. Finally, we draw conclusions and present ideas for future work.

1.1 Related Work

Ozdaglar and Bertsekas [4] present a Linear Programming (LP) approach to the RWA. Here, the RWA is represented as a *multicommodity network flow* problem with additional constraints. Additional constraints vary for setups with no, sparse, or full wavelength conversion. The LP formulations are similar to our formulation (Sec. 2), except that the authors use a piecewise linear cost function and limit the maximum number of wavelengths, whereas we assume constant link costs of 1 and impose no limit on the wavelengths to guarantee feasibility.

A memetic algorithm is presented by Sinclair [5] including a mutation operator, recombination (exchange a subset of paths between parents, reassign wavelengths) and two local search operators. The fitness function is rather complicated considering link length, link usage, node degree, and more. The first local search (called 'path reroute') reroutes a request in a wavelength with smaller index using one of the k-shortest paths, whereas the second local search ('path shift-out') assigns a path to another wavelength, but shifts out conflicting paths first. The network model uses the concept of several fibers on one physical link. Although this approach is said to perform well, the large population size (500) and the vast number of generations (100 000) question its efficiency.

Another approach for the RWA on general graphs is splitting the problem into subproblems for routing and wavelength assignment, respectively. An extensive overview on this approach is provided in [6] by Zang *et al.* and in [7] by Choi *et al.*

The static RWA is an NP-complete problem, for which we present an iterated local search algorithm. *Iterated Local Search* (ILS) [8] describes a class of algorithms that build a sequence of solutions using an embedded *local search* (LS) algorithm. To leave local optima, results of the embedded algorithm may be perturbated e. g. by a random mutation which moves the current solution to a different part of the search space allowing the LS to find other (hopefully better) local optima. LS [9] is an algorithm class defining for each solution a *neighborhood*, which is a subset of the solution space and contains all solutions which differ from the current solution in some selected aspect. In each iteration, the algorithm evaluates the current solution's neighborhood to choose a new solution and thus performs a random walk in the search space. Both neighborhood and moving strategy influence the LS's performance. Furthermore, LS algorithms are incomplete (finding optimal solutions is not guaranteed) and get stuck in local optima requiring diversification.

2 Lower Bounds

To evaluate our solutions' quality, we determined lower bounds by solving a relaxation of the RWA. Disregarding the wavelength continuity constraint, the RWA reduces to a multicommodity flow problem as follows. For each request $r \in R$ we have commodity which needs to be routed by means of flow through the network G. The mass balance d_v^r required for request r at node v is $-d_i$ if $v = s_i$, $+d_i$, if $v = t_i$ and zero otherwise. Let variable $x_{(i,j)}^r$ indicate the amount of flow from request r sent over the edge $(i,j) \in E$. Then, the problem of minimizing the maximum flow sent over an edge can be stated as:

$$\min \max_{(u,v)\in E} \sum_{r\in R} x_{(u,v)}^r + \sum_{r\in R} x_{(v,u)}^r$$

$$\sum_{u\in V:(u,v)\in E} x_{(u,v)}^r - \sum_{u\in V:(v,u)\in E} x_{(v,u)}^r = d_v^r \qquad \forall v \in V, r \in R \qquad (1)$$

$$x_{(u,v)}^r + x_{(v,u)}^r \leq d_v^r \qquad \forall r \in R, (i,j) \in E \qquad (2)$$

$$x_{(u,v)}^r \in \mathbb{N} \qquad \forall r \in R, (u,v) \in E$$

The objective is to minimize the maximum load of any edge in the network. Constraints (1) are flow-conservation constraints ensuring each request in the demand is fulfilled. Constraints (2) ensure that the flow for a request traverses an edge (u,v) only in one direction.

3 Benchmark Instances and Construction Algorithms

Solutions can be represented by a mapping from requests to sets of wavelength-path combinations. As our objective has the prerequisite of fulfilling all requests completely, an equivalent representation can be achieved by setting the requests' demands to 1, but allowing multiple requests for the same node pair. Therefore w. l. o. g., for the course of this paper every request has a demand of exactly one.

A proposal for a construction algorithm is given in [10]. Requests are processed iteratively by assigning each request a wavelength and a path depending on one of the strategies below. Initially, one wavelength is available, but the number of wavelengths is increased when no path can be found for a request.

First Fit (FF_RWA). Requests are ordered randomly and will be routed in the first wavelength with a feasible (shortest) path.

Best Fit (BF_RWA). Requests are ordered randomly and will be routed in the wavelength with the shortest feasible path.

First Fit Decreasing (FFD_RWA). Requests are sorted non-increasingly by the length of each request's shortest path in G and will be routed in the first wavelength with a feasible path.

Best Fit Decreasing (BFD_RWA). Requests are sorted non-increasingly by the length of each request's shortest path in G and will be routed in the wavelength with the shortest feasible path.

Experimental results in [10] indicate that BFD_RWA finds best results. Here, all test instances were constructed by the Erdős-Rényi random graph model $G(n, p)$ [11], where each possible edge (i, j) is chosen with probability $p = \frac{\delta}{n-1}$ independently from all other edges (where $n = |V|$ and δ is the expected node degree) and only connected graphs are accepted. In a second step the graph is made bidirectional by replacing each undirected edge by a pair of antiparallel directed edges. The demand between each pair of nodes (i, j) ((i, j) and (j, i) are considered as different node pairs) was set to 1 by a given probability between 0.2 and 1.0. Similar instances have been used in other publications, e. g. [12,13].

Our own preliminary experiments with this graph model and demand matrix (in our setting, however, edges can be used in both directions and thus are not replaced by a pair of unidirectional edges) indicate that this type of instance is completely uninteresting due to two facts: First, the demand is so small that construction algorithms as above already give (near-)optimal solutions. In a preliminary experiment, we generated a $G(n, p)$ instance with 50 nodes, $\delta = 3$, and probability for a request between a node pair of $p = 0.4$, where the BFD_RWA construction heuristic found a solution whose quality equals the lower bound (Sec. 2). Second, the underlying graph is 'pathological' in most cases. We call a graph pathological if it is only 1-edge connected, i. e. it contains at least one bridge. To illustrate the high probability of getting a pathological graph by the Erdős-Rényi random graph model, consider a graph $G(100, \delta{=}3)$. Here it is rather improbable that a graph with a connectivity of degree > 1 will be generated. The probability of one special node being a leaf is already $\left(1 - \frac{3}{99}\right)^{98} \cdot \left(\frac{3}{99}\right)^{1} \cdot 99 = 0.147$ indicating that the probability of an at least 2-edge connected graph is very small [14]. Every request depending on such a bridge automatically requires an additional wavelength increasing the

Table 1. Properties of instances from SNDlib [15]. 'Pairs' describes the number of node pairs communicating with each other, 'Requests' summarizes the demand volume.

Instance	Nodes	Edges	Pairs	Requests	Lower Bound
`atlanta`[†]	15	22	210	6840	1256
`france`[†]	25	45	300	10008	1060
`germany50`	50	88	662	2365	147
`janos-us-ca`[†]	39	122	1482	10173	1288
`newyork`	16	49	240	1774	85
`nobel-eu`	28	41	378	1898	304
`nobel-germany`	17	26	121	660	85
`nobel-us`	14	21	91	5420	670
`norway`	27	51	702	5348	543
`pdh`	11	34	24	4621	214
`polska`	12	18	66	9943	1682
`zib54`	54	81	1501	12230	705

[†] Instance has been modified, see text for details.

total number of wavelengths, although previously allocated wavelengths still have a lot of free links.

Instead of generating random instances as described above, we used standard benchmark instances from the SNDlib collection [15]. This collection consists of 22 networks and a set of associated models which can be used as problem instances for the Survivable Network Design problem. The data was derived from industrial and research background. A network is described by nodes, (physical) links, demands, and other, for our problem irrelevant planning data. From this collection we chose 12 networks as listed in Tab. 1. We restricted our selection as the other instances were either uninteresting for sophisticated algorithms or exceeded our available system capacities. To include some large instances, we scaled down the demand matrices of instances `atlanta`, `france`, and `janos-us-ca` by a factor of 20, 10, and 200, respectively. For the benchmark instances, capacity constraints and predefined admissible paths were not used and edge costs were fixed to 1. Edges were interpreted to be bidirectional and all demands between node pairs (i, j) and (j, i) were summed up to a unidirectional demand. The graph of instance `nobel-us` corresponds to the well-known NSF network with 14 nodes.

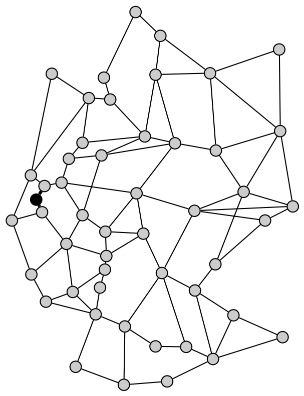


Fig. 2. Instance `germany50` based on 50 cities in Germany

Our preliminary experiments indicated that the structure of the graph is not the only limiting factor for the solution quality. Additional problems arise if the requests' load is unbalanced distributed among the node pairs. Given a node with degree δ, in each wavelength this node can handle at most δ requests, especially requests starting or ending in this node. In instance `germany50` (Fig. 2), the highlighted node has a node degree of 2 and 43 requests start here with a summarized demand of 293. This leads to the lower bound $\lceil \frac{293}{2} \rceil = 147$, which equals the solution found by our iterated local search (Sec. 5) and the lower bound given by the multicommodity flow solution (Tab. 1).

We classify requests depending on overloaded edges as 'evil' requests. The influence of evil requests can best be observed for instances where an unfavorable combination of graph and request set leads to some heavily overloaded edges. These observations motivate a new approach of the construction heuristic. In this new approach, evil requests are preferably routed during the construction phase. For our experiments we consider three different sorting strategies. Other combinations of the components Len, Anti and Evil are also possible but will not be considered further.

Len. Requests are ordered non-increasingly by the length of their shortest path, equals BFD_RWA.
AntiEvil. Evil requests are routed first, otherwise using a random sorting.
LenAntiEvil. Requests are first sorted by Len and within each set of requests with equal shortest path length by AntiEvil.
AntiEvilLen. Requests are first sorted by AntiEvil and within the evil and non-evil requests, respectively, by Len.
Shuffle Requests are ordered in a random fashion.

To locate evil requests for a given graph and set of requests, we propose the following method. In the first step, an initial solution is constructed with the standard BFD_RWA algorithm and all edges $e \in E'$ heavily used in marginally used wavelengths are taken as candidates for overloaded edges. We define marginally used wavelengths as wavelengths whose usage lies below a multiple k of the average path length. In a second step requests are marked as evil requests, if they cannot be routed in $G \setminus E'$ (Fig. 3). Here, $u(s, \lambda)$ calculates the actual load of wavelength λ (number of used physical links), use(s, W', e) describes how often edge e is used in solution s restricted to wavelengths W', and routeable(G, r) checks if request r is physically routeable in G.

```
1: function FindEvilRequests(Graph $G$, Requests $R$, $k \in \mathbb{R}$)
2:     $s \leftarrow$ BFD_RWA$(G, R)$                          ▷ Construct a BFD_RWA solution
3:     $n \leftarrow k \cdot \lceil \text{avg}_{r \in R}\, l(p_{G,s}(r)) \rceil$     ▷ $n$ is the $k$ times the average path length in $s$
4:     $E \leftarrow \emptyset$                                  ▷ Set of possibly overloaded edges
5:     $W' \leftarrow \{\lambda \in W : u(s, \lambda) < n\}$            ▷ Set of marginally used wavelengths
6:     for all $e \in E$ do
7:         if use$(s, W', e) > |W'| - 2$ then      ▷ is $e$ used in $> |W'| - 2$ wavel. of $W'$?
8:             $E' \leftarrow E' \cup \{e\}$                         ▷ store overloaded edge
9:     return $\{r \in R : \neg$routeable$(G \setminus E', r)\}$
```

Fig. 3. Determination of Evil Requests

Table 2. Construction heuristic's results (minimum and average over 50 runs)

	Len		LenAntiEvil		AntiEvilLen		AntiEvil		Shuffle	
Instance	min	avg	min	avg	min	avg	min	avg	min	avg
`atlanta`	1415	1473.7	1417	1474.0	1424	1518.0	1460	1556.5	1457	1524.8
`france`	1091	1115.1	1095	1120.2	1094	1131.7	1110	1175.3	1105	1164.6
`germany50`	185	193.2	185	190.8	165	166.4	161	164.3	163	172.9
`janos-us-ca`	1781	1806.7	1764	1791.3	1495	1507.2	1523	1567.3	1522	1611.7
`newyork`	92	96.9	91	97.4	91	96.4	95	100.5	93	101.3
`nobel-eu`	304	304.2	304	304.0	304	304.0	304	305.5	304	310.3
`nobel-germany`	91	94.0	92	94.1	91	94.5	93	98.8	92	99.7
`nobel-us`	827	892.1	812	885.7	836	890.1	805	877.7	835	898.6
`norway`	555	564.8	557	564.7	559	568.5	570	586.8	570	585.0
`pdh`	267	298.0	275	296.4	271	298.7	257	297.0	268	291.6
`polska`	1750	1836.1	1747	1861.6	1788	1943.1	1842	2010.7	1855	1971.9
`zib54`	824	903.7	816	884.2	825	892.2	827	947.6	854	983.3

3.1 Experimental Results

All problem instances in Tab. 1 were solved by the construction heuristics Len, LenAntiEvil, AntiEvilLen, AntiEvil and Shuffle. Each experiment was repeated with 50 seeds, the results (minimum and average) are summarized in Tab. 2.

As can be seen in Tab. 2, there exists no clear preference between the sorting strategies Len and LenAntiEvil (except for `germany50` and `janos-us-ca`, the confidence intervals overlap). Instances `germany50`, `janos-us-ca`, `nobel-us`, and `pdh` perform better with the sorting strategy AntiEvil than with Len or LenAntiEvil regarding the best solution, however, only for `janos-us-ca` and `germany50` there is a significant difference (99 %). E. g. the best solution for `germany50` using AntiEvil is 13.0 % better than using Len or LenAntiEvil. We observed that for these instances the least used wavelengths in solutions built using Len contain only a few, similar paths indicating that the underlying request matrix is unbalanced and thus yields some heavily overloaded edges. E. g. in one selected solution for `germany50`, a request having a demand of 76 uses 32 wavelengths used by no other request to route 63 lightpaths. As long as there exist few evil requests, AntiEvilLen nearly matches with the sorting strategy Len and thus has quite a similar but slightly worse performance. For the instances where AntiEvil performs well, no clear preference can be made between AntiEvil and AntiEvilLen. Thus, for unknown instances our experiments indicate first to try both strategies Len and AntiEvil and then depending on which strategy performs better to construct the final solutions either with Len/LenAntiEvil or with AntiEvil/AntiEvilLen, respectively. However, there may be a large variance between different solutions from the same construction heuristic. E. g. for `nobel-us`, the variance on the number of wavelengths ranges between 24.2 and 40.5 (not shown in Tab. 2). This suggests to construct several solutions to confirm the decision for the best construction heuristic.

```
1: procedure ShiftPaths(Solution $s$, Requests $R$)
2:     for all $r \in R$ do                                        ▷ for each request ...
3:         $W' \leftarrow \{\lambda \in W | \Psi_{G,s}(r,\lambda) \neq \emptyset\}$ ▷ find wavelengths in which $r$ can be routed
4:         $\lambda \leftarrow \arg\max_{\lambda \in W'} u(s,\lambda)$          ▷ find wavelength $\lambda$ with maximum load
5:         if $u(s,\lambda) > u(s,\lambda_s(r))$ then      ▷ compare usage of $\lambda$ and request's wavel.
6:             $\lambda_s(r) \leftarrow \lambda$                              ▷ set request's new wavelength
7:             $p_{G,s}(r) \leftarrow \Psi_{G,s}(r,\lambda)$ ▷ request's path set to shortest path in new wavel.
```

Fig. 4. Local Search moving path to wavelengths with higher load

4 Local Search

Although there is a large variety in solution quality among different request sorting strategies and random seeds, in many cases the results are considerably worse than the lower bounds motivating our local search (LS).

The general idea of our LS is to shift requests from less used wavelengths to more often used wavelengths by looking for alternative (possibly longer) paths. The algorithm (Fig. 4), which operates on a solution s and the set of requests R, works as follows: For each request $r \in R$, the set of all wavelengths $W' \subseteq W$ in which r is routeable is determined. Among all wavelengths in W', the wavelength λ with highest load is chosen. If the load for λ is larger than the load for the request's current wavelength, then the request's wavelength is set to λ and the request's path is updated with the shortest path in λ. Function $\Psi_{G,s}(r,\lambda)$ calculates the shortest path in G for request r routed in λ and function $u(s,\lambda)$ is introduced in Sec. 3. We define $\Psi_{G,s}(r,\lambda) = \emptyset$, iff there exists no such path.

4.1 Experimental Results

To evaluate the effectiveness of the LS, we performed multistart experiments using the same setup as described in Sec. 3.1. We applied our LS ShiftPaths to the 50 initial solutions of each construction heuristic setup until a local optimum was reached. The results are summarized in Tab. 3 (best of 50).

When comparing our multistart LS algorithm to the construction heuristics, the former performs only slightly better than the underlying construction heuristic. E. g. for instance `germany50`, the construction heuristic's best solution using Len is 185 (average 193.2), but the multistart LS's best solution is only 184. A significance analysis (99 % confidence interval) shows that the multistart LS improves the construction heuristics only in 5 cases (`france`+Shuffle, `germany50`+AntiEvil/AntiEvilLen, `norway`+AntiEvil/Shuffle).

This observation matched our expectations, as both the construction heuristics and the LS follow similar strategies. The only difference is that the construction heuristics prefer shortest paths, whereas the LS allows to reroute requests using longer paths if the load on wavelengths gets changed in respect of the optimization criterion. Differences in the quality of solutions created by the various construction heuristics cannot be compensated by the multistart LS.

Table 3. Multistart local search's results (best of 50 runs)

Instance	Len	LenAntiEvil	AntiEvilLen	AntiEvil	Shuffle
`atlanta`	1414	1417	1424	1418	1430
`france`	1086	1092	1085	1102	1096
`germany50`	184	185	164	159	160
`janos-us-ca`	1781	1764	1494	1497	1495
`newyork`	92	90	91	92	92
`nobel-eu`	304	304	304	304	304
`nobel-germany`	91	91	91	91	89
`nobel-us`	827	804	808	787	802
`norway`	554	554	555	556	560
`pdh`	263	259	266	254	256
`polska`	1748	1747	1787	1827	1854
`zib54`	800	787	785	785	810

5 Iterated Local Search

As shown above, local search alone is not sufficient to considerably improve the quality of the initial solutions. Thus, in order to escape from local optima, we introduce an iterated local search (ILS) combining the local search with a mutation (perturbation) operator which randomly changes paths within an already existing solution (Fig. 5). In each mutation step, two wavelengths λ_1 and λ_2 are randomly chosen and an arbitrary request whose path p is routed in the wavelength with lower usage (here, λ_2) is taken. Paths preventing p from being routed in wavelength λ_1 are removed from the solution and p is routed in λ_1. The function $\text{rem}(s, \lambda_1, p)$ determines the paths to be removed and stores the paths' requests in R'. To restore a valid solution, all requests in R' are rerouted in the first possible wavelength.

We applied different mutation strategies to the ILS, where the strength is defined by a percentage of the number of paths. Mutation strategies were either constant (ranging between 1 % and 25 %) or variable, where the mutation strength started with an initial high mutation rate (either 10 % or 25 %) and decreased linearly (step width 1 % or 2 %) until reaching a strength of 1 %.

Within our ILS, we accept new solutions after mutation and local search iff it is better (by length-lex ordering on wavelength usage vectors) than the previous best solution, otherwise the previous best solution is restored. Length-lex ordering sorts vectors first by length and then by lexicographic ordering.

5.1 Experimental Results

Due to similarities of Len and LenAntiEvil, we restrict the following discussion to Len, AntiEvil, AntiEvilLen, and Shuffle. The number of generations was limited to 50 as for most instances optimal solutions were found within this bound. Experiments were repeated 50 times.

```
 1: procedure MUTATE(Solution $s$, Requests $R$, $strength$)
 2:     for $i = 1 \ldots strength$ do
 3:         $(\lambda_1, \lambda_2) \leftarrow$ RandomWavelengths
 4:         if $u(s, \lambda_1) < u(s, \lambda_2)$ then
 5:             swap$(\lambda_1, \lambda_2)$
 6:         $r \leftarrow \mathrm{rInWL}(s, \lambda_2, R)$        ▷ get a request $r$ with path $p$ in wavelength $\lambda_2$
 7:         $p \leftarrow p_{G,s}(r)$
 8:         $R' \leftarrow \mathrm{rem}(s, \lambda_1, p)$       ▷ remove paths stopping $p$ from being routed in $\lambda_1$
 9:         $p_{G,s}(r) \leftarrow p$
10:         $\lambda_s(r) \leftarrow \lambda_1$
11:         for all $r \in R'$ do                              ▷ for all currently unrouted paths
12:             $\lambda_s(r) \leftarrow \arg\min_{\lambda \in W} \Psi_{G,s}(r, \lambda) \neq \emptyset$
13:             $p_{G,s}(r) \leftarrow \Psi_{G,s}(r, \lambda)$     ▷ route request on the first possible wavelength
```

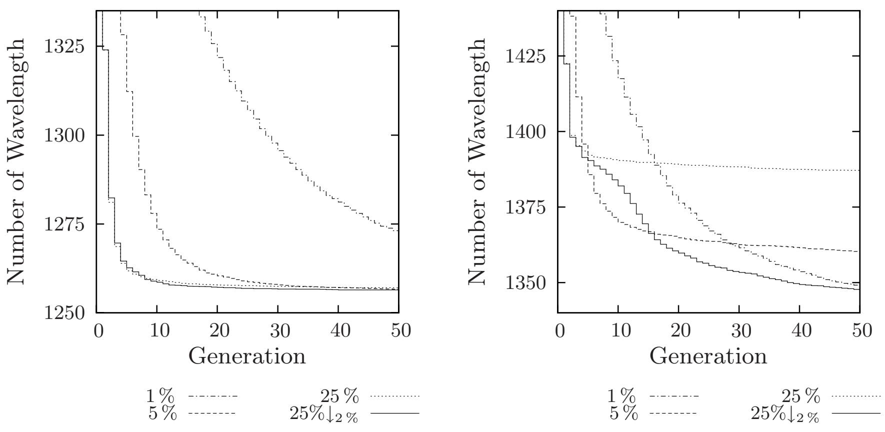
Fig. 5. Mutation shifting path between wavelengths

Whereas LS without mutation is not powerful enough to improve initial solutions, the ILS always improves them to the optimum for many instances (reaching the LB from Sec. 2 and Tab. 1). Best results were achieved with a variable mutation strategy, where the initial strength was set to 25 % decreasing by 2 % each generation ($25\%\downarrow_{2\,\%}$, Tab. 4). In each setup, the ILS resulted in significantly better results than the multistart LS (99 % confidence intervals).

Regarding the convergence towards the optimal solution, we observed two different patterns. In the first case, the optimal solution was reached in a few generations by setups with strong mutation, whereas setups with weak mutation converged much slower. Instance `atlanta` (Fig. 6a) is an example for this behavior, where the optimal solution is reached after about 15 generations for strong mutation (25 %), after 40 generations for mutation strength 5 % and not

Table 4. Iterated Local Search's results with minimum and average over 50 runs for the mutation strategy starting at 25 % and decreasing by step width 2 % ($25\%\downarrow_{2\,\%}$)

		Len		AntiEvilLen		AntiEvil		Shuffle	
Instance	LB	min	avg	min	avg	min	avg	min	avg
`atlanta`	1256	$\underline{1256}$	1256.5	$\underline{1256}$	1256.4	$\underline{1256}$	1256.3	$\underline{1256}$	$\underline{1256.2}$
`france`	1060	1061	1062.5	$\underline{1060}$	$\underline{1062.4}$	$\underline{1060}$	1062.7	1061	1062.9
`germany50`	147	$\underline{147}$	147.8	$\underline{147}$	147.6	$\underline{147}$	148.0	$\underline{147}$	$\underline{147.5}$
`janos-us-ca`	1288	1343	1351.4	$\underline{1337}$	$\underline{1347.5}$	1340	1357.8	1342	1358.4
`newyork`	85	$\underline{85}$	85.1	$\underline{85}$	85.2	$\underline{85}$	$\underline{85.0}$	$\underline{85}$	85.2
`nobel-eu`	304	$\underline{304}$	$\underline{304.0}$	$\underline{304}$	$\underline{304.0}$	$\underline{304}$	$\underline{304.0}$	$\underline{304}$	$\underline{304.0}$
`nobel-germany`	85	$\underline{85}$	$\underline{86.2}$	86	86.5	$\underline{85}$	86.4	86	86.4
`nobel-us`	670	$\underline{684}$	689.3	685	689.3	$\underline{684}$	689.3	$\underline{684}$	$\underline{688.6}$
`norway`	543	$\underline{543}$	$\underline{543.0}$	$\underline{543}$	543.1	$\underline{543}$	543.1	$\underline{543}$	$\underline{543.0}$
`pdh`	214	$\underline{215}$	217.3	$\underline{215}$	217.2	216	217.3	$\underline{215}$	$\underline{217.0}$
`polska`	1682	$\underline{1682}$	1682.2	$\underline{1682}$	$\underline{1682.1}$	$\underline{1682}$	$\underline{1682.1}$	$\underline{1682}$	$\underline{1682.1}$
`zib54`	705	709	711.3	709	711.5	$\underline{708}$	$\underline{710.0}$	$\underline{708}$	710.2



(a) atlanta using Len **(b)** janos-us-ca using AntiEvilLen

Fig. 6. Performance plots for two ILS setups using different mutation strategies

reached for weak mutation (1 %) within 50 generations. In the second case, the optimal solution was not reached by our ILS. Again, our mutation strategies performed differently. Stronger mutations let the ILS converge faster, but, however, get stuck in weak local optima. Weaker mutation lead to slow convergence, but end eventually in better solutions. Using dynamic mutation, our ILS converges fast during the initial phase and later is able to continuously improve its current solution. An example for this behavior is shown in Fig. 6b for problem instance janos-us-ca. The bump for line '$25\%{\downarrow_{2\,\%}}$' is due to the dynamic mutation strategy and varies for different parameters.

For 8 out of 12 benchmark instances our algorithm is able to find optimal solutions, as it reaches the lower bound. Here, we can argue that the use of wavelength converters will not result in solutions with less wavelengths, as the lower bound is based on a relaxation assuming wavelength converters at every node. For pdh and zib54, near-optimal solutions are found, and with more generations our algorithm can find optimal solutions. Only for janos-us-ca and nobel-us our ILS is not able to reach solutions close to the lower bound. Reasons may be that these two problem instances are harder than the other instances or that the lower bounds are considerably below the optimal solution.

6 Conclusion

In this paper we improved an existing construction heuristic and developed an iterated local search for the static RWA. We adopted problem instances from the SND library for the RWA and argued that these benchmark instances are more interesting than previously used random instances.

As for some benchmark instances the BFD_RWA construction heuristic performed badly, we subsequently suggested alternative request sorting strategies resulting in considerably better initial solutions. To further improve these solutions, we introduced an ILS which finds provable optimal solutions for eight instances and near-optimal solutions for two more instances.

Future work will focus on run-time optimizations for large instances. We are evaluating a multi-level approach based on the scaling mechanism we used for large problems in this paper. Furthermore, using don't-look-bits on wavelengths or requests to restrict the search space is another promising concept.

References

1. Chlamtac, I., Ganz, A., Karmi, G.: Lightnet: Lightpath based solutions for wide bandwidth wans. In: INFOCOM, pp. 1014–1021 (1990)
2. Xu, S., Li, L., Wang, S.: Dynamic routing and assignment of wavelength algorithms in multifiber wavelength division multiplexing networks. IEEE J. Sel. Areas Comm. 18(10), 2130–2137 (2000)
3. Chlamtac, I., Faragó, A., Zhang, T.: Lightpath (Wavelength) Routing in Large WDM Networks. IEEE J. Sel. Areas Comm. 14(5) (1996)
4. Ozdaglar, A.E., Bertsekas, D.P.: Routing and wavelength assignment in optical networks. IEEE/ACM Trans. Netw. 11(2) (2003)
5. Sinclair, M.C.: Minimum cost routing and wavelength allocation using a genetic-algorithm/heuristic hybrid approach. In: Proc. 6th IEE Conf. Telecom. (1998)
6. Zang, H., Jue, J.P., Mukherjee, B.: A Review of Routing and Wavelength Assignment Approaches for Wavelength-Routed Optical WDM Networks. Optical Networks Magazine 1(1), 47–60 (2000)
7. Choi, J.S., Golmie, N., Lapeyrere, F., Mouveaux, F., Su, D.: A Functional Classification of Routing and Wavelength Assignment Schemes in DWDM networks: Static Case. In: Proc. of the 7th International Conference on Optical Communications and Networks, OPNET 2000, pp. 1109–1115 (2000)
8. Lourenço, H.R., Martin, O.C., Stützle, T.: Iterated Local Search. In: Glover, F., Kochenberger, G. (eds.) Handbook of Metaheuristics. International Series in Operations Research & Management Science, vol. 57, pp. 321–353 (2002)
9. Hoos, H.H., Stützle, T.: Stochastic Local Search: Foundations and Applications. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann, San Francisco (2004)
10. Skorin-Kapov, N.: Routing and Wavelength Assignment in Optical Networks using Bin Packing Based Algorithms. EJOR 177(2), 1167–1179 (2007)
11. Erdős, P., Rényi, A.: On Random Graphs I. Publ. Math. Debrecen 6, 290–297 (1959)
12. Manohar, P., Manjunath, D., Shevgaonkar, R.K.: Routing and Wavelength Assignment in Optical Networks From Edge Disjoint Path Algorithms. IEEE Communications Letters 6(5), 211–213 (2002)
13. de Noronha, T.F., Resende, M.G.C., Ribeiro, C.C.: A Random-Keys Genetic Algorithm for Routing and Wavelength Assignment. In: Proc. of the Seventh Metaheuristics International Conference (MIC 2007) (2007)
14. Bollobás, B.: Random Graphs. Cambridge University Press, Cambridge (2001)
15. Orlowski, S., Pióro, M., Tomaszewski, A., Wessäly, R.: SNDlib 1.0–Survivable Network Design Library. In: Proc. of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium (2007), `http://sndlib.zib.de`