Branch-and-Bound Algorithms for Constrained Paths and Path Pairs and Their Application to Transparent WDM Networks

Franz Rambach
Student of the TUM
Telephone: 0049–89–12308564
Email: rambach@in.tum.de

Dominic A. Schupke
Siemens, Corporate Technology
Optical Networks and Transmission
CT IC ONT, Otto-Hahn-Ring 6
81730 Munich, Germany

***Abstract*— We introduce three novel algorithms for routing in WDM networks. These schemes are based on the "Best-First Branch-and-Bound" algorithm and take constraints into account. An exact algorithm for the unprotected routing problem in WDM networks, one exact algorithm, and a heuristic for the routing with 1+1 protection are described. By simulations with realistic optical networks and grid topologies we can demonstrate that our algorithms exhibit short computation times.**

I. Introduction

In every communication network we have the problem how to route a message. Often the searched path should fulfill constraints additionally. Let us focus on WDM networks. We assume a connection request between a source node S and a target node T should be established. Furthermore a given bit error rate (BER) and/or availability should be guaranteed for this connection. Hence we have a routing problem, which should also satisfy given constraints. In this paper we stick to the case of linear constraints, which can model the BER approximately and the availability exactly. However the problem is easily extended to monotonically increasing constraint functions. The described problem is called the unprotected routing problem in the following. For high available connections the request can force not only one, but two disjoint paths to be set up. These two paths should be node or link disjoint. This protection approach is called 1+1 protection and thus we call it the 1+1 protection problem.

The operator of a network wants to minimize the cost for every connection that is set up. We must find one feasible path for the unprotected routing problem, as well as minimize a given monotonically increasing cost function among all feasible paths. In the case of the 1+1 protection problem we minimize the sum of the cost function concerning the two searched paths. In addition to that we try to minimize the costs for the wavelengths, which exist in the system, too. Now an additional constraint can occur for the 1+1 protection problem: It can be required to route the two searched paths on the same wavelength.

Recapitulatory the unprotected problem can be formulated in the following way: An optical network, two nodes S and T, linear constraints and a cost function are given. The problem is to find the cheapest path w.r.t. to a cost function among all paths that fulfill the constraints. The 1+1 protection problem can be formulated in a similar way. Both problems are NP-complete as long as they have at least one constraint [1], [2].

We show that a constraint which enforces that a path has an availability of more than a_g can be transformed into a linear constraint. For simplicity we assume that every node has an availability of 1 which means that a node never fails. Every edge e has an availability of a_e. With these definitions we can write the constraint for a path p in the following way:

$$\prod_{e\in p} a_e \geq a_g$$

Through taking the logarithm of this inequality and multiplying with -1 we get

$$\prod_{e\in p} a_e \geq a_g \Longleftrightarrow \sum_{e\in p} -\log a_e \leq -\log a_g \quad .$$

If we define c_e through $c_e := -\log a_e$ and replace the bound a_g by $c_g := -\log a_g$ we get the linear constraint

$$\sum_{e\in p} c_e \leq c_g \quad .$$

We have shown that an availability requested for a path can be modeled with a linear constraint.

In an analog way we can adapt the cost function for the unprotected routing problem such that we maximize the availability of the searched path. Availability maximization is often necessary where a targeted availability is not achievable, to obtain at least best-effort availability values regardless of cost. The cost function can also model the maximization of the availability for the whole connection in the 1+1 protection problem. Again we assume that nodes do not fail. Let a_w be the availability of the working path p_w and a_p of the protection path p_p. These availabilities are calculated via

$$a_w = \prod_{e\in p_w} a_e \quad \text{and} \quad a_p = \prod_{e\in p_p} a_e$$

We want to maximize the availability of the whole connection which is $1-(1-a_w)(1-a_p)$. This maximization problem can be transformed in a minimization problem:

$$\begin{aligned} & \max\{1-(1-a_w)(1-a_p)\} \\ \Longleftrightarrow \quad & \min\{-1+(1-a_w)(1-a_p)\} \\ \Longleftrightarrow \quad & \min\{(1-a_w)(1-a_p)\} \end{aligned}$$

We derived a monotonically increasing cost function which should be minimized. Therefore we can chose the objective to maximize the availability of the whole connection even if we chose the 1+1 protection.

The paper is organized in the following way: The next section describes three developed algorithms; one for the unprotected routing problem, a heuristic and one exact algorithm for the 1+1 protection scheme. To accelerate the described schemes we introduce two different techniques in Section III. Through simulations with different realistic networks and grid topologies we compare our proposed algorithm with the software CPLEX in Section IV. Section V concludes with a summary of our results.

II. The Algorithms

The three algorithms we introduce in this section incorporate linear constraints directly into the search process. Furthermore they minimize among all paths and wavelengths simultaneously. All these three schemes are using basic ideas of the "Best-First Branch-and-Bound" algorithm [3], and adapt them to WDM networks. This BFBAB algorithm is very similar to the algorithm A*Prune, introduced by [4]. The basic idea of this two algorithms is to execute Dijkstra's algorithm and to check for every calculated sub-path if the constraints are fulfilled.

A. A Modification Of The BFBAB Algorithm

In this paragraph we introduce the modified BFBAB algorithm for the unprotected routing problem. This algorithm is exact, i.e. it finds the shortest path that fulfills the constraints.

We describe how the modified BFBAB algorithm works in detail. This exact algorithm searches in a structured way all possible, feasible paths which are connecting S and T. The algorithm uses three different objects. One of them is a variable Best_Cost that stores the value of the cost function for the shortest feasible path known so far. The second object is the State-Space-Node. This object stores a node v of the network, the path from S to this node v, the set of the free wavelengths for this path, and the values for the constraints and the cost function. The third object is a heap that stores State-Space-Nodes.

The algorithm starts from the source node S and constructs a corresponding State-Space-Node, containing the node S, the empty path, all the wavelengths which are in the system, and 0 for all the constraints and the cost function. Then this object is stored in the heap. The variable Best_Cost is set to infinity. As long as the heap is not empty and the cost of the top element of the heap is smaller than the value of the variable Best_Cost the algorithm executes the following steps:

1) Take the first element of the heap and delete this object from the heap. Let v be the node of this object A.
2) For all nodes k which can be directly accessed from v do the following:
   a) Construct a new State-Space-Node B with the node k. The path p of this new object is defined by the path from object A plus the edge e which leads from v to k. The free wavelengths for this object are the intersection between the free wavelengths of the object A and the free wavelengths of the added edge e.
   b) Check if the path p violates any constraints. If it does, delete B and go to step 2.
   c) Check if the node k is the target node T. If it is, store this object B as a feasible solution. If additionally the value for the cost function of B is smaller than Best_Cost, set the variable Best_Cost to the value for the cost function of B. Proceed with step 2.
   d) Store the object B in the heap and sort the heap w.r.t. the value of the cost functions of the objects.

Because the modified BFBAB algorithm solves the NP-complete unprotected routing problem in an exact way, its running time is not polynomial with having complexity of $O(d^N)$, see [3]. Here d is the maximum node degree and N is the number of nodes in the network. Although the complexity is exponential, the actual running times in optical networks are fast, as we will see in the simulation paragraph.

B. An Exact Algorithm For The 1+1 Protection Problem

The idea of this exact algorithm, also called EI1+1BFBAB, is to adapt the modified BFBAB algorithm, such that it searches instead of a single path a cycle which includes node S and T. Hence, it searches a path which starts from node S, passes through node T, and also returns back from T to S.

This exact algorithm works in the same way as the modified BFBAB algorithm until a sub-path arrives at the node T. In this case the values of the constraints for this sub-path are stored in other variables and the origin variables are set to 0. If both searched paths can use arbitrary wavelengths, the set of the free wavelengths is stored in another set and the origin set is reset to all the wavelengths that are in the system. If both searched paths have to use the same wavelength, we proceed as described in the modified BFBAB algorithm. When we construct the second sub-path which connects T with S we must ensure the required disjointness concerning the first already computed sub-path. This exact algorithm proofs either the existence of no solution or it calculates a cycle. The derived cycle, which starts at node S, is cut at the node T into two disjoint paths p_1 and p_2. Without loss of generality path p_1 starts at node S and ends at node T and path p_2 starts at node T and ends at node S. This pair of paths p_1 and p_2 is the optimal solution of the 1+1 protection problem.

The variable Best_Cost is used in the same way as in the modified BFBAB algorithm.

The EI1+1BFBAB algorithm always finds the optimal solution of the NP-complete 1+1 protection problem. Its complexity is again not polynomial, being $O(d^{2N})$. Again, here d represents the maximum node degree and N the number of nodes in the network.

C. A Heuristic For The 1+1 Protection Problem

The basic idea of the following heuristic H1+1BFBAB is to calculate with the help of the modified BFBAB algorithm the shortest paths iteratively and to check if two disjoint paths exists among the computed paths known so far. The algorithm is described by the following steps:

1) Let A be the empty set. Compute with the modified BFBAB algorithm one feasible path p. If no such path is available, stop this algorithm with the result that there is no solution for this problem. If p exists, add p into A.
2) Calculate a new path p_i with the modified BFBAB algorithm using the knowledge of the previous run of the BFBAB algorithm, i.e. the previous run can be continued to calculate p_i. If there is no such additional path p_i, stop this algorithm with the result that the 1+1 protection problem has no solution.
3) Check if the new path p_i is edge disjoint (or node disjoint) to one of the paths stored in A, and optionally further check if this two disjoint paths can use the same wavelength. This tests are performed, depending on the value of the cost function for every path. Because the n paths in the set A are ordered w.r.t. the cost, i.e. $p_1 \leq p_2 \leq \ldots \leq p_n$ w.r.t. the costs, we first check p_i with p_1, then p_i with p_2, and so on until we test p_i with p_n. The first path pair p_i and p_m that fulfills the constraints is the solution the heuristic calculates. If no such path pair exists, p_i is added to the set A and the set A is ordered w.r.t. the cost. Proceed with step 2.

After calculating a path p_i in step 1 or in step 2 we do not conclude the algorithm but pause it. So we can use the information the modified BFBAB algorithm has already gathered to calculate an additional feasible path. Because the BFBAB algorithm has already stored many sub-paths to many different nodes, it can calculate another feasible path by carrying on the previous calculation. To avoid the stopping of the BFBAB algorithm after the shortest path is calculated, the algorithm must not use the variable Best_Cost.

By ordering the set A or testing, depending on the cost of the paths in step 3, we can ensure that the calculated path pair is the shortest among all the known paths.

Although this described algorithm always finds a solution if one exists, it does not always calculate the optimal path pair. As we will see in Section IV however, the heuristic still often computes the optimal solution. If the optimal solution is not found by the heuristic, the approximation is still extremely good.

The complexity of this algorithm is not polynomial, because — as mentioned above — it always finds a solution if one exists. Although it is really fast for optical networks in practice.

III. Acceleration techniques

To accelerate the three described algorithms we used the technique look-ahead, which is described e.g. in [5]. To apply this concept first we calculate the shortest paths — in the sense of the constraint i — for every constraint i from the target node T to all other nodes in the network. This can be done by Dijkstra's algorithm, for example. Let $c_{i,v}$ be the value for the shortest path from T to node v for the constraint i. Furthermore let the values for the constraint i be c_i for a sub-path p and the boundary for the constraint i be $c_{NB,i}$. To test if a sub-path p, which starts from node S and has the end node v can be part of the optimal path, normally we test if $c_i \leq c_{NB,i}$. With the look-ahead concept we add to the c_i the value $c_{i,v}$. This value $c_{i,v}$ was calculated in the first step. With the look-ahead concept we test the inequality $c_i + c_{i,v} \leq c_{NB,i}$ instead of $c_i \leq c_{NB,i}$. By this technique it is known much earlier that a sub-path p cannot be part of a feasible path.

The look-ahead concept can be used for the three introduced algorithms without having any disadvantages.

Another concept to speed up the modified BFBAB algorithm is the technique of the dominated paths, which e.g. is also described in [5]. We compare two sub-paths p_A and p_B which start at node S and end at the same node v. If path p_A is better than p_B in all its properties, we know that p_B can never be part of an optimal path. That means we do not need to take p_B into account any further. All its properties show that both the cost function and all the constraints of path p_A are less than or equal to those of path p_B. If this is not the case we do the comparison vice versa, i.e. test if p_B is better than p_A.

The dominated path concept can only be used for the unprotected routing algorithm, but not for the 1+1 protection algorithms.

Applying these acceleration techniques enhances the calculating times extremely, as we see in the following paragraph.

IV. Simulation Results

We implemented the three described algorithms in a given simulator using the programming language C/C++. For all the simulations a computer with an Opteron 252 with 2.6 GHz CPU-frequency and 4 GB RAM was used. We chose the following three constraints:

$$\text{length} \leq 2000\,\text{km} \quad (1)$$

$$\text{hops} \leq 7 \quad (2)$$

$$\frac{320\,\text{km}}{3} \cdot \text{hops} + \text{length} \leq 2026.67\,\text{km} \quad (3)$$

In the European research project NOBEL five realistic optical networks have been modeled: These five networks are one core (BT Core) and one regional network (BT Metro) of the British Telecom, one from the Telecom Italia (Tilab), one European network (PAN Europe) and one of the Deutsche Telekom (DT Netz). We assume for the simulations that every link in all five networks can carry 80 wavelengths.

TABLE I

USED TIME (IN SECONDS) CPLEX NEEDED FOR SOLVING THE UNPROTECTED ROUTING PROBLEM

Network	Number of Nodes and Node Degree δ	ILP	
		$\varnothing$ Time	$80 \cdot \varnothing$ Time
BT Core	79 / 3.5	0.0092	0.74
TILAB	38 / 4.2	0.0067	0.54
BT Metro	38 / 3.3	0.0061	0.49
DT Netz	17 / 3.0	0.0047	0.37
PAN Europe	16 / 2.9	0.0048	0.38

For the following section we define δ as the average node degree of a network and denote the maximum routing time in a network as T_{max}.

A. Unprotected Routing Problem

To compare the developed algorithms we solve the unprotected routing problem with the standard multi-purpose ILP solver CPLEX and with the modified BFBAB algorithm on the five given networks and on grids. CPLEX should find the shortest path which fulfills the constraints for every wavelength in the system separately. In the next step it searches the best path among all the computed ones. To estimate how long it takes to solve the unprotected problem by CPLEX, we multiply the time CPLEX needs to solve the problem for one wavelength with the number of wavelength in the system. Both, the average time and the time multiplied by 80, which represents the number of wavelengths in the system, is displayed in Table I. Here the length of the path was minimized. We see that even for the network with 79 nodes CPLEX needed only 0.74 seconds on average for solving the unprotected routing problem.

Instead of solving the problem for every wavelength separately, CPLEX can minimize among all wavelengths and all paths simultaneously. For both cases similar times were encountered for solving the whole unprotected routing problem.

With the modified BFBAB algorithm we run simulations with five different minimization objectives. The first objectives minimize the number of hops or the physical length of the path. In the other cases we chose the weight for the objective function in different ways. A scenario in which the operator has the option to lease other links was assumed. In this case we set the weights for the operators own links to 1 and for leased links to 2. In the next case we choose the weight depending on the current load of the link. If l wavelengths are already used on a link, the weight was set to 2^l. In the last scenario we set the weights for the objective function on a random number between $[1, 10]$. The corresponding times are shown in Table II.

We see in Table II that the average routing times of the modified BFBAB algorithms are nonsensitive to the objective function. In the case of the leased links and the weight depending on the load of the links, the maximum routing times T_{max} deviate from the average by a factor up to 35. In all other cases the factor is 8.

Another interesting fact are the higher routing times of the Tilab network compared with the times of the BT Metro network, which has the same number of nodes. This is caused by the higher average node degree of the Tilab network. It has a node degree of 4.2. On the contrary, the BT Metro network has an average node degree of 3.3. Caused by this node degree the routing of the Tilab network takes roughly the same time as routing on the bigger BT Core network (node degree = 3.6) on average. With these data we can deduce that the modified BFBAB algorithm is sensitive to the average node degree of a network.

By comparing the CPLEX and the BFBAB times we see that CPLEX needs for the problem-solving concerning one wavelength almost the same time than BFBAB needs for solving the whole unprotected routing problem. Hence, the modified BFBAB algorithm is faster by a factor which corresponds to the number of wavelengths in the system. This statement is also true for a different number of wavelengths, because the modified BFBAB algorithm minimizes among all wavelengths simultaneously and the running time is independent of the number of wavelengths in the system.

Further simulations were executed for both algorithms on grid topologies. The lengths of the links were chosen randomly from a uniform distribution between $[1, 300]$. We took the same constraints as before, but scaled the boundaries accordingly, so that feasible paths can be found in the bigger networks. Again, we minimized the physical lengths of the path. Similar times were achieved for some other objective functions. The resulting times are displayed in Fig. 1. The logarithmic scaled abscissa displays the nodes in the grids and the logarithmic scaled ordinate displays the corresponding routing times for the unprotected problem.

We see in Fig. 1 that the BFBAB algorithms routes one demand under 1 second for up to 900 nodes. Furthermore the figure tells that CPLEX scales better than the modified BFBAB algorithm. But for grid sizes up to 2500 nodes the modified BFBAB algorithm has a far better performance than CPLEX. Only in even more bigger networks CPLEX will outperform the modified BFBAB algorithm. Because optical networks have far less nodes than 2500, normally less than 100, the modified BFBAB algorithm is a better choice than CPLEX.

B. 1+1 Protection Problem

Four different variations exist for the 1+1 protection problem. These are generated by the combination of the requested disjointness (node or edge) and the requirement that both paths shall use the same wavelength or can independently use arbitrary ones. To analyze the routing time for the four different variations we display the average and maximum routing time for the BT Metro network in Table III. The paths were calculated by the introduced heuristic and the exact algorithm for the 1+1 problem. The objective function was to minimize the physical length of the sum of both paths. As we

TABLE II

TIME (IN SECONDS) FOR SOLVING THE UNPROTECTED ROUTING PROBLEM WITH LINEAR CONSTRAINTS AND WITH DIFFERENT OBJECTIVE FUNCTIONS ON REALISTIC OPTICAL NETWORKS BY THE MODIFIED BFBAB ALGORITHM

Network	Number of Nodes / δ	Hops		Length		Leased Links		Exponential		Random	
		∅ Time	T_{max}	∅ Time	T_{max}	∅ Time	T_{max}	∅ Time	T_{max}	∅ Time	T_{max}
BT Core	79 / 3.5	0.0086	0.07	0.0086	0.03	0.0085	0.30	0.0080	0.20	0.0085	0.02
TILAB	38 / 4.2	0.0081	0.03	0.0091	0.03	0.0081	0.03	0.0077	0.05	0.0084	0.03
BT Metro	38 / 3.3	0.0065	0.02	0.0068	0.02	0.0065	0.02	0.0061	0.02	0.0068	0.02
DT Netz	17 / 3.0	0.0045	0.02	0.0047	0.02	0.0046	0.02	0.0047	0.02	0.0048	0.02
PAN Europe	16 / 2.9	0.0035	0.02	0.0037	0.02	0.0036	0.02	0.0044	0.24	0.0035	0.02

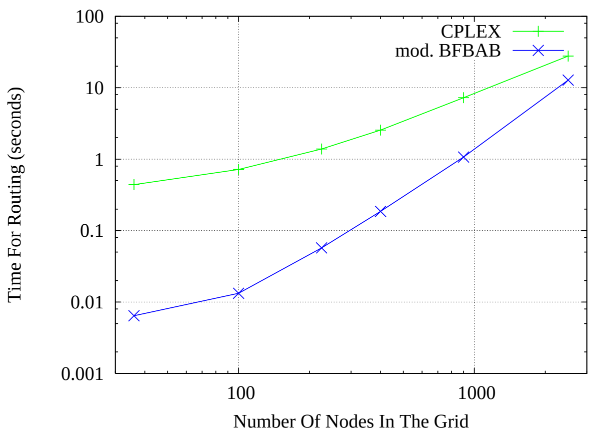

Fig. 1. Time for solving the unprotected routing problem on grids by the modified BFBAB algorithm and the software CPLEX

see, all times for the different variations are quite similar. The heuristic is about three times faster than the exact algorithm.

Because the times for the routing are nearly the same in the four variations, we limit our further simulations to search for edge disjoint paths, only, which both use the same wavelength. In Table IV the average and maximum times for the heuristic and the exact algorithms are displayed, also the time it takes for solving the unprotected problem by CPLEX. The success ratio (SR) is defined by

$$\mathrm{SR} := \frac{\text{Number of optimal found solutions}}{\text{Number of demands in which a solution exists}} \, .$$

It approximates the probability of the heuristic finding the optimal solution. Let $HeuSum$ and $ExactSum$ be the sum of the objective function of all the successfully routed pairs of paths of the heuristic and the exact algorithm. Also let Heu and $Exact$ be the sum of the objective function of the current successfully routed pair of paths of the heuristic and the exact algorithm. We define the total relative error E by

$$\mathrm{E} := \frac{HeuSum - ExactSum}{ExactSum}$$

and the current relative error e by

$$\mathrm{e} := \frac{Heu - Exact}{Exact} \quad .$$

In addition to the success ratio SR the total relative error E and the value e_{max}, which is the maximum of all current relative errors e seen in the simulations, is shown in Table IV.

This table shows that the heuristic calculates the optimal pair of paths for over 70% and sometimes even for over 90%. That also means the total relative error is quite small, only 2%. Even if the optimal solution is not found, the maximum current error has a value up to 80% but mostly up to 50% only. The heuristic is approximately four times faster than the exact algorithm. For the Tilab network the heuristic is even about 100 times faster than the EI1+1BFBAB algorithm. This is because the exact algorithm needs that much time for this network. Again, this is caused by the relative high average node degree of 4.2. The heuristic and the exact algorithm are much faster than the software CPLEX on these five networks except on the Tilab network. On this network CPLEX has a better performance than the exact algorithm, but the heuristic is still about 20 times faster than CPLEX.

For further performance-studies we solved the 1+1 protection problem on grids. The grids were generated in the same way than before. We also used the same constraints with appropriate boundaries. Times for the routing of the software CPLEX, the heuristic and the exact algorithm are displayed in Fig 2.

The EI1+1BFBAB algorithm has a better average performance than CPLEX up to 81 nodes. But this exact algorithm scales worse than CPLEX. As a result CPLEX is faster on grids with more than 81 nodes. In contrast to the bad scaling of the EI1+1BFBAB algorithm, the heuristic scales for grids up to 100 nodes well, although CPLEX scales even better. The heuristic is for grids up to 100 nodes faster than CPLEX.

TABLE III

TIME (IN SECONDS) FOR SOLVING THE 1+1 PROTECTING ROUTING PROBLEM FOR NODE OR EDGE DISJOINT PATHS ON THE BT METRO NETWORK BY THE H1+1-BFBAB AND EI1+1-BFBAB ALGORITHM

Disjoint	Wavelengths for both paths	H1+1-BFBAB		EI1+1-BFBAB	
		∅ Time	T_{max}	∅ Time	T_{max}
Node	same	0.0083	0.03	0.026	0.20
Node	arbitrary	0.0082	0.02	0.026	0.19
Edge	same	0.0079	0.02	0.033	0.25
Edge	arbitrary	0.0083	0.30	0.035	0.26

TABLE IV

TIME (IN SECONDS) FOR SOLVING THE 1+1 PROTECTING ROUTING PROBLEM ON REALISTIC OPTICAL NETWORKS BY THE H1+1-BFBAB, THE EI1+1-BFBAB ALGORITHM AND CPLEX

Netz	H1+1-BFBAB					EI1+1-BFBAB		CPLEX
	∅ Time	T_{max}	SR	E in %	e_{max} in %	∅ Time	T_{max}	80 · ∅ Time
BT Core	0.0094	0.15	0.83	2.0	79	0.045	0.63	1.91
TILAB	0.0114	0.05	0.71	2.0	48	1.268	38.55	0.99
BT Metro	0.0079	0.02	0.83	1.2	37	0.033	0.25	0.79
DT Netz	0.0054	0.02	0.75	1.6	36	0.020	0.08	0.55
PAN Europe	0.0038	0.02	0.93	0.8	28	0.005	0.02	0.53

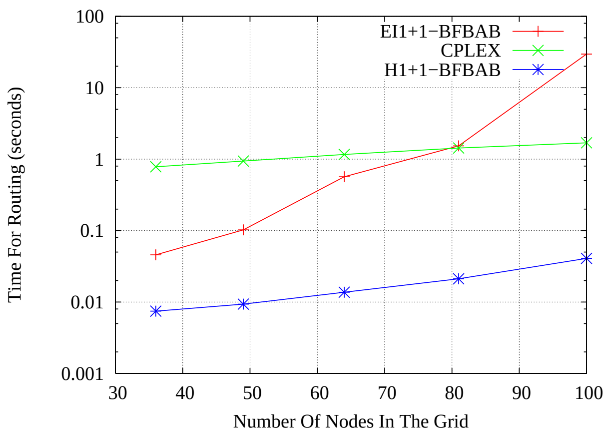

Fig. 2. Time for solving the 1+1 protection routing problem on grids by the H1+1BFBAB, EI1+1BFBAB algorithm and the software CPLEX

Further simulations demonstrated that for grids with 225 nodes CPLEX and the heuristic have roughly the same average routing times.

The maximum routing times differ from the average ones highly. This discrepancy gets bigger, if the size of the network grows. We define the factor l as the quotient of the maximal time divided by the average routing time. For a grid with 100 nodes the heuristic has a factor of 7.1, which is tolerable. In contrast to that, CPLEX has a factor of $l = 2$ only. But the exact algorithm has a factor up to 66.6, which is high. This means that most of the demands can be routed in a short time, although some demands exist which need a lot more time with this exact algorithm.

V. CONCLUSION

We developed three different algorithms for routing problems. One algorithm is for the unprotected routing problem with linear constraints, which outperforms CPLEX, even in networks with over 2000 nodes. In smaller networks we showed that the developed algorithm is faster than CPLEX – about a factor ω, where ω corresponds to the number of wavelengths in the system.

The other two algorithms solve the 1+1 protection problem, taking linear constraints into account. In small networks the EI1+1BFBAB algorithm is faster than CPLEX. In networks with 100 nodes and more the introduced heuristic H1+1BFBAB is faster than CPLEX. Furthermore, to a high percentage it computes the optimal solution. If the optimal paths are not found by the heuristic, the approximation is quite good. Moreover the H1+1-BFBA algorithm does always find a solution, if at least one exists.

The modified BFBAB and the H1+1BFBAB qualify for routing in large optical networks. The EI1+1BFBAB algorithm is adequate as long as the network size is small, and is suitable for typical optical networks.

While all three algorithms have been investigated using linear cost functions and constraints, they are applicable to problems with monotonically increasing cost functions and constraints.

ACKNOWLEDGMENT

The authors would like to thank Anita Rambach for reviewing this paper. The authors acknowledge the EU project NOBEL.

REFERENCES

[1] M. Ziegelmann, “Constrained shortest paths and related problems,” Ph.D. dissertation, Universität Saarbrücken, Naturwissenschaftlich-Technische Fakultät I, 2001. [Online]. Available: http://scidok.sulb.uni-saarland.de/volltexte/2004/251/pdf/MarkZiegelmann_ProfDrKurtMehlhorn.pdf

[2] Y. Guo, F. Kuipers, and P. V. Mieghem, “Link-disjoint paths for reliable QoS routing,” *International Journal of Communication Systems*, vol. 16, no. 9, pp. 779–798, Nov. 2003.

[3] W.-L. Yang, “Solving the MCOP problem optimally,” in *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN 2002)*. Washington, DC, USA: IEEE Computer Society, Nov. 2002, pp. 100–112.

[4] G. Liu and K. G. Ramakrishnan, “A*Prune: An algorithm for finding K shortest paths subject to multiple constraints,” in *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2001)*, vol. 2. Anchorage, Alaska, USA: IEEE Computer Society, Apr. 22–26 2001, pp. 743–749.

[5] P. V. Mieghem and F. A. Kuipers, “Concepts of exact QoS routing algorithms,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 5, pp. 851–864, Oct. 2004.