

# A Hybrid Constraint Model for the Routing and Wavelength Assignment Problem

Helmut Simonis\*

Cork Constraint Computation Centre  
Department of Computer Science, University College Cork, Ireland  
h.simonis@4c.ucc.ie

**Abstract.** In this paper we present a hybrid model for the demand acceptance variant of the routing and wavelength assignment problem in directed networks, an important benchmark problem in optical network design. Our solution uses a decomposition into a MIP model for the routing and optimization aspect, combined with a finite domain constraint model for the wavelength assignment. If a solution to the constraint problem is found, it provides an optimal solution to the overall problem. If the constraint problem is infeasible, we use an extended explanation technique to find a good relaxation of the problem which leads to a near optimal solution. Extensive experiments show that proven optimality is achieved for more than 99.8% of all cases tested, while run-times are orders of magnitude smaller than the best known MIP solution.

## 1 Introduction

The routing and wavelength assignment problem (RWA) [2, 12, 20] in optical networks considers a network where demands can be transported on different optical wavelengths through the network. Each accepted demand is allocated a path from its source to its sink, as well as a specific wavelength. Demands routed over the same link must be allocated different wavelengths, while demands whose paths are link disjoint may use the same wavelengths.

The RWA problem is a well studied, important problem in optical network design, for which many problem variants have been considered. Depending on the technology used, the network may be assumed to be *directed* or *undirected*. The *static design problem* considers the problem of allocating all given demands on a network topology, using the minimal number of frequencies. The *demand acceptance problem* considers a fixed, given number of frequencies on all links in the network. The objective is to accept the maximal number of demands in the network. In this paper we discuss the demand acceptance problem in a directed network.

More formally, we are considering a directed network  $G = (N, E)$  of nodes  $N$  and edges  $E$ . A demand  $d \in D$  is between source  $s(d)$  and sink  $t(d)$ . We use the notation  $\text{In}(n)$  and  $\text{Out}(n)$  to denote all edges entering resp. leaving node  $n$ . The set  $\Lambda$  of

---

\* This work was supported by Science Foundation Ireland (Grant Number 05/IN/I886). Support from Cisco Systems and the Silicon Valley Community Foundation is gratefully acknowledged.

available wavelengths is fixed and identical throughout the network. We can formulate a basic model of the problem with two sets of 0/1 integer variables. Variables  $y_d^\lambda$  denote whether demand  $d$  is accepted using wavelength  $\lambda$ , variables  $x_{de}^\lambda$  state whether edge  $e$  is used to transport demand  $d$  on wavelength  $\lambda$ .

$$\max \sum_{d \in D} \sum_{\lambda \in \Lambda} y_d^\lambda \quad (1)$$

s.t.

$$y_d^\lambda \in \{0, 1\}, x_{de}^\lambda \in \{0, 1\} \quad (2)$$

$$\forall d \in D : \sum_{\lambda \in \Lambda} y_d^\lambda \leq 1 \quad (3)$$

$$\forall e \in E, \forall \lambda \in \Lambda : \sum_{d \in D} x_{de}^\lambda \leq 1 \quad (4)$$

$$\forall d \in D, \forall \lambda \in \Lambda : \sum_{e \in \text{In}(s(d))} x_{de}^\lambda = 0, \sum_{e \in \text{Out}(s(d))} x_{de}^\lambda = y_d^\lambda \quad (5)$$

$$\forall d \in D, \forall \lambda \in \Lambda : \sum_{e \in \text{Out}(t(d))} x_{de}^\lambda = 0, \sum_{e \in \text{In}(t(d))} x_{de}^\lambda = y_d^\lambda \quad (6)$$

$$\forall d \in D, \forall \lambda \in \Lambda, \forall n \in N \setminus \{s(d), t(d)\} : \sum_{e \in \text{In}(n)} x_{de}^\lambda = \sum_{e \in \text{Out}(n)} x_{de}^\lambda \quad (7)$$

Constraint (2) enforces integrality of the solution, constraint (3) states that a demand can use at most one wavelength. The *clash* constraint (4) states that on each edge, only one demand may use any given wavelength. Constraints (5) and (6) link the  $x$  and  $y$  variables at the source (resp. sink) of each demand. Finally, constraint (7) enforces flow balance on all other nodes of the network.

The main contributions of this paper are

- a novel, two-step problem decomposition for the RWA problem into a MIP (Mixed Integer Programming) and finite domain constraint model,
- a new, very accurate upper bound to the RWA problem based on a resource-based relaxation of an existing, source aggregation MIP solution,
- the use of explanation techniques to understand infeasibility of the constraint model, suggesting good candidates for problem relaxation,
- extension of the model to handle parallel fibers without increasing problem size,
- experimental results showing that very high quality solutions are obtained by this method in seconds, outperforming the best MIP model by orders of magnitudes.

In the next section we will describe existing solutions to the problem, with special emphasis on complete MIP models. We then describe our decomposition strategy, presenting a resource-based MIP relaxation and the finite domain constraint model. In section 4, we describe how we can detect and explain infeasibility of the constraint model, and how we can relax the problem to obtain good, but possibly sub-optimal solutions. We then extend our approach to allow parallel links in the network without increasing the size of the model. This is followed in section 6 by an extensive experimental evaluation of the proposed technique, before we consider possible further research in section 7.

## 2 Related Work

The RWA problem has been studied using many different solution approaches, see [4] for an overview. We can distinguish two main approaches. Greedy heuristics use local search techniques to accept demands incrementally, providing fast solutions for large problem cases, but without a formal guarantee of solution quality. Alternatively, complete methods, mainly based on ILP (Integer Linear Programming) techniques, can provide optimal solutions, but are restricted in the problem size handled.

The MIP formulation (1) does not scale well with increasing number of demands and network size. A major factor is the potential symmetry between all frequencies as well as additional symmetries due to multiple demands between the same source and sink. In [5], different ILP reformulations of the problem are considered, the best alternative uses a source aggregation, described below. In this model, one does not consider individual demands, but aggregates all demands starting in the same source node. We introduce integer variables  $y_{sd}$  to denote how many demands from a source  $s$  to a sink  $d$  are accepted. The upper limit for these variables is given by  $P_{sd}$ , the total number of requested demands between  $s$  and  $d$ . We then define variables  $x_{se}^\lambda$  to state whether wavelength  $\lambda$  on edge  $e$  is used to transport a demand originating in  $s$ , without identifying which demand is carried. We use the notation  $D_s$  for the set of destinations of requested demands originating in  $s$ .

$$\max \sum_{s \in N} \sum_{d \in D_s} y_{sd} \quad (8)$$

s.t.

$$y_{sd} \in \{0, 1, \dots, P_{sd}\}, x_{se}^\lambda \in \{0, 1\} \quad (9)$$

$$\forall e \in E, \forall \lambda \in \Lambda: \sum_{s \in N} x_{se}^\lambda \leq 1 \quad (10)$$

$$\forall s \in N, \forall \lambda \in \Lambda: \sum_{e \in \text{In}(s)} x_{se}^\lambda = 0 \quad (11)$$

$$\forall s \in N, \forall d \in D_s, \forall \lambda \in \Lambda: \sum_{e \in \text{In}(d)} x_{se}^\lambda \geq \sum_{e \in \text{Out}(d)} x_{se}^\lambda \quad (12)$$

$$\forall s \in N, \forall d \in D_s: \sum_{\lambda \in \Lambda} \sum_{e \in \text{In}(d)} x_{se}^\lambda = \sum_{\lambda \in \Lambda} \sum_{e \in \text{Out}(d)} x_{se}^\lambda + y_{sd} \quad (13)$$

$$\forall s \in N, \forall n \neq s, n \notin D_s, \forall \lambda \in \Lambda: \sum_{e \in \text{In}(n)} x_{se}^\lambda = \sum_{e \in \text{Out}(n)} x_{se}^\lambda \quad (14)$$

Constraints (9) define the integrality conditions. Constraint (10) specifies the *clash* constraint between demands from different sources. Constraint (11) states that demands originating in  $s$  can not be routed through  $s$ , while constraints (12) and (13) consider the destinations of demands originating in  $s$  and state that the correct number of demands must be dropped in each node, linking the  $x$  and  $y$  variables. Finally, constraint (14) enforces flow balance at all other nodes of the network.

The solution of formulation (8) does not provide an immediate, unique solution for the demand acceptance problem. The path for each accepted demand must be extracted from the solution by a small procedure, which can also be used to eliminate loops in the paths at the same time.

A very nice property of the model is that the LP relaxation, replacing the integrality constraint (9) with continuous domain restrictions, provides a very tight upper bound for the ILP solution.

The aggregated model (8) performs much better than model (1), especially if the number of demands is increasing. But experiments in [5] show that obtaining optimal solutions even for small networks may require several hours, if not days, of computation time. As network size increases, the number of sources to consider increases as well, leading to a dramatic performance loss.

The use of column generation has been considered [6] for a *path-based* [16] formulation of the problem. This increases the problem size that can be handled, but still requires solution times of several hours.

So far constraint programming has not been used to solve the RWA problem; a general overview of constraint applications in the network domain is given in [16]. Smith in [17] discusses a design problem for optical networks, but this is restricted to a ring topology, and minimizes the need for ADM multiplexers.

On the other hand, the RWA problem considered here is not too far removed from the demand acceptance problem in MPLS traffic engineering (MPLS-TE) in IP networks, which has been approached with multiple hybrid constraint solution techniques as described in [8, 9, 16]. The main difference is that demands in the MPLS-TE problem have integer sizes and overall link capacity limits are enforced instead of clash constraints. This motivated our solution approach to the RWA problem using a decomposition using a resource based relaxation, which we will describe in the next chapter.

### 3 Solution Approach

In this section we describe our solution approach which is based on a simple decomposition strategy.

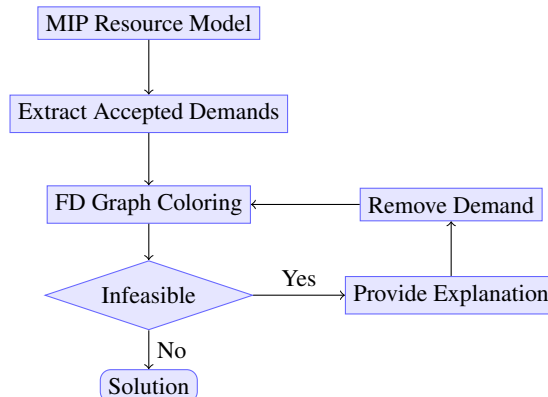
#### 3.1 Problem Decomposition

A solution to the RWA problem must consider the following three activities:

1. Select demands to be accepted
2. Choose paths for each accepted demand
3. Assign wavelength

We choose a decomposition technique which handles the first two steps with a MIP program which is a relaxation of model (8), and a second step which consists of a finite domain model for a resulting graph coloring problem, where each accepted demand is a node and there are disequality constraints between demands which are routed over the same link. The overall solution approach is shown in figure 1. In the MIP model, we replace the clash constraints with simple capacity constraints which limit the number

**Fig. 1.** Solution Approach



of demands that can be routed through each link. The optimal solution to this problem is an upper bound to the RWA problem. We then generate a graph coloring problem by imposing disequality constraints between all accepted demands which are routed over the same link. The number of available colors is limited by the available wavelengths. Using finite domain constraint programming, we search for a feasible solution to the problem. If we are successful, we have an optimal solution to the overall problem. If the constraint problem is infeasible, we drop some demands until a feasible solution is obtained. This solution may be sub-optimal, but usually is very close to the previously obtained upper bound. In order to choose which demand to drop, we have to understand why the constraint problem is infeasible. We use two techniques to find an explanation, one, structurally by detecting large cliques in the constraint graph, the other based on the QuickXplain [7] method. Once an explanation is obtained we heuristically choose one of its demands for relaxation in the overall problem. This creates a new graph coloring problem, which we recursively solve with our procedure.

Note that the suggested approach is not guaranteed to find an optimal solution, as a sub-optimal, infeasible solution to the second step only affects the second phase of the algorithm. To obtain a complete procedure, we would have to extend the feedback loop from the explanation generation back to the first phase MIP model. We will consider such an extension in future work.

### 3.2 Resource Allocation MIP Model

In order to simplify model (8), we relax the wavelength constraints and replace them with capacity constraints over all links. Each link has capacity  $C = |\Lambda|$ . We still use the integer variables  $y_{sd}$ , which state how many demands from  $s$  to  $d$  are accepted, but we replace the  $x_{se}^\lambda$  variables with integer  $z_{se}$  variables. These variables count how many demands originating in  $s$  are routed over link  $e$ . We use  $T_s = \sum_{d \in D_s} P_{sd}$ , the total number of requested demands starting in  $s$ , as the upper bound on the  $z_{se}$  variables.

$$\max \sum_{s \in N} \sum_{d \in D_s} y_{sd} \quad (15)$$

s.t.

$$y_{sd} \in \{0, 1 \dots P_{sd}\}, z_{se} \in \{0, 1 \dots T_s\} \quad (16)$$

$$\forall e \in E : \sum_{s \in N} z_{se} \leq C \quad (17)$$

$$\forall s \in N : \sum_{e \in \text{In}(s)} z_{se} = 0 \quad (18)$$

$$\forall s \in N, \forall d \in D_s : \sum_{e \in \text{In}(d)} z_{se} = \sum_{e \in \text{Out}(d)} z_{se} + y_{sd} \quad (19)$$

$$\forall s \in N, \forall n \neq s, n \notin D_s : \sum_{e \in \text{In}(n)} z_{se} = \sum_{e \in \text{Out}(n)} z_{se} \quad (20)$$

Constraint (16) describes the integrality constraints, note that all variables have integer, not 0/1 domains. Constraint (17) is the link capacity constraint, which relaxes the clash constraints (10) for individual wavelengths. Constraint (18) limits the use of the source node, while constraint (19) describes the balance around the destination nodes, linking the  $y$  and  $z$  variables. Finally, constraint (20) imposes flow balance for all other nodes.

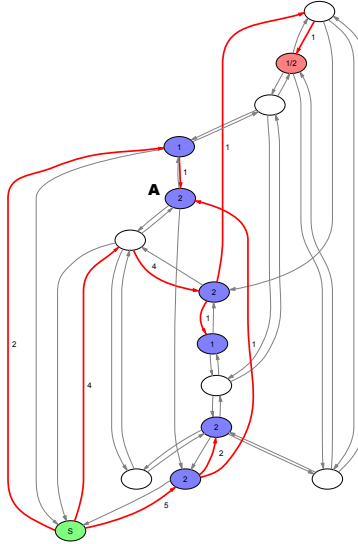
The optimal solution to model (15) provides an upper bound to the RWA problem. Perhaps surprisingly, this bound was not discussed in [5], although an equivalent relaxation based on a *path formulation* [16] was presented.

Similar to the situation for model (8), the solution does not directly tells us which demands are accepted and which paths they should take. A simple, but non-deterministic program is required to extract these elements, which are required in the second step of our procedure. At the same time, this procedure removes possible loops in the paths, which would cause problems in the graph coloring model. A simple example is given in figure 2. It shows the result obtained for one source (in green, marked **S**) and all possible sinks for demands starting in **S**. The red links show the non-zero values of the  $z_{se}$  variables for this source, the nodes in blue and red show the non-zero values of the  $y_{sd}$  variables. The node marked **A** can be reached on two different paths. Which path is used for which demand is not defined in the MIP model, but will be selected by the solution extraction routine. In the red node, only one of the two requested demands is satisfied. The MIP solution again does not state which of the demands must be rejected.

### 3.3 Graph Coloring Model

After running the MIP model (15), and extracting the accepted demands and their paths, we can now try to allocate the available wavelengths with a simple finite domain constraint model. For each of the  $a$  accepted demands, a finite domain variable  $f_d$  ranging over all available wavelengths  $\Lambda$  is generated. For each link, we consider all demands

**Fig. 2.** Non-deterministic Solution Extraction



which are routed over it. The corresponding domain variables must be pairwise different, or, alternatively, an *alldifferent* constraint must hold between all of them. Considering all links together, we can express the constraints by either

- a set of binary *disequality* constraints between any two demands which are routed over a common link
- a set of *alldifferent* constraints [18], one for each link in the network
- a single *some-different* global constraint [14], expressing the complete disequality constraint network

More formally, using  $D^e$  to denote all accepted demands routed over link  $e$ , we generate the *alldifferent* constraints:

$$\forall e \in E : \text{alldifferent}(\{f_d | d \in D^e\}) \quad (21)$$

Alternatively, we can generate an equivalent set of disequality constraints:

$$\forall d_1, d_2 \in D, d_1 \neq d_2, \exists e \in E \text{ s.t. } d_1 \in D^e, d_2 \in D^e : f_{d_1} \neq f_{d_2} \quad (22)$$

We initially use *bounds consistent* *alldifferent* constraints to provide a good compromise between computational effort required and resulting propagation.

In our search routine, we are interested only in finding a feasible solution quickly, not in proving infeasibility, as this is handled separately. We try three different search routines, each limited by a timeout (5 seconds) to restrict the effort expended:

- We first use an incomplete search routine, in our case *credit-based search* [1, 3], which explores the top of the search tree completely, but allows only limited backtracking in lower parts of the tree. The variable selection method used is *first-fail*, and we use a randomized value selection method. We consecutively try credit amounts of  $a$ ,  $a^2$  and  $a^3$  to improve chances of finding a solution quickly.
- In a second step, we try a complete search routine, still using bounds-consistent alldifferent constraints, keeping the same variable and value selection strategies. This might succeed when a larger amount of backtracking steps is required to find a solution.
- In a third step, we again use a complete search, but impose *domain consistent* versions of the alldifferent constraints, so that more propagation may detect failures which cause deep backtracking in earlier steps.

We could add specialized symmetry breaking techniques to handle the existing *value symmetries* between the available frequencies, as well as potential *variable symmetries* between demands sharing the same source, sink and path. These constraints will interfere with our method to detect infeasibility, and therefore were not implemented.

In the absence of additional symmetry breaking constraints, we can use a combination of shaving, impact analysis and symmetry breaking to preassign one of the large cliques to values. For each of the alldifferent constraints, we try to fix all their variables to values 1 to  $C$ , and measure the product of the domain sizes of the remaining variables in the problem. We fix the clique whose assignment will lead to the largest problem reduction before starting the overall search. We also observe that in a few cases this preassignment leads to an immediate failure, which we will exploit in the next section to explain infeasibility.

## 4 Handling Infeasibility

In the vast majority of examples tested, the finite domain solver quickly finds a feasible solution, which results in an optimal solution of the overall problem. In relatively few cases, the constraint problem is infeasible, which means we have to drop additional demands to obtain a complete RWA solution. We use two specialized methods to detect infeasibility and provide an explanation, which we can use to suggest likely demands to be rejected.

### 4.1 Clique Detection

A simple technique considers the structure of the constraint graph. For many links, the number of demands routed over them is equal to the overall capacity  $C$ , the number of available wavelengths. The variables corresponding to these demands form a clique in the disequality constraint graph. We can easily check each such clique if any additional variable (demand) must be different from each of its variables. This increases the size of the clique beyond the number of possible values and makes the problem infeasible. To obtain a feasible solution, we have to remove at least one of the demands in the infeasible clique from the problem.

As the graph coloring problem in this case is already given as a collection of cliques, many of which are already at limit size, we don't have to search for large cliques in the disequality constraint graph. This more generic method has been used in other constraint solvers for graph coloring problems.

## 4.2 Explanation

When we create the constraint variables and set up the alldifferent constraints, there will be no initial constraint propagation, if the size of the largest clique is equal to or smaller than the number of available frequencies. But we noticed that sometimes the preassignment of a single clique to initial values will lead to a failure, showing the problem to be inconsistent. At this point, we can use traditional explanation techniques like QuickXplain [7] to find a conflict set in the constraint graph after the preassignment. Together with the alldifferent constraint corresponding to the preassignment this explanation provides an (not necessarily minimal) explanation of the overall infeasibility. We run the explanation procedure at the level of individual alldifferent constraints, not their binary decomposition, in order to profit from the bounds consistent reasoning of the constraints. It might be possible to reduce the explanation further by removing some of the variables from the alldifferent constraints, we don't attempt this. Instead, we provide an explanation for each of the failed preassignments, and consider that set of possible explanations.

An explanation for the infeasibility in our system is a set of alldifferent constraints over variables which correspond to accepted demands. This means that that set of demands routed on the paths which were assigned in the first phase can not be allocated to the given wavelengths while satisfying the clash constraints. If (some of) the demands were allocated on different paths, such a solution might exist. We do not consider this possibility in our current approach, but resolve the infeasibility by rejecting one of the demands occurring in the explanation.

In order to suggest which demand should be removed to make the problem feasible, we count how often a demand occurs in an explanation and in how many explanations it is present. We order the demands by decreasing number of occurrences and try to remove the demands with the largest count first. This creates a new constraint problem, which we try to solve recursively with the same technique. The method will always terminate, as we remove one demand at each step, but fortunately will require only one or two steps in most cases before a feasible solution is found.

## 5 Extension to Multigraphs

An interesting extension of the problem considers the possibility of using more than one fiber between nodes in the network, e.g. replacing the directed graph with a multigraph. This is often done to increase capacity on connections which carry a lot of traffic, and can typically be achieved at little extra cost as cables between locations already carry many fibers in a single cable strand.

At first sight, we can use our existing model without change, modelling each parallel fiber as a separate edge in the multigraph, for which we generate variables in our MIP

model and constraints in the finite domain graph coloring model. The disadvantage is that we increase the number of variables and introduce additional symmetries in our model, as the choice between the parallel fibers is unrestricted.

We can reduce the size of our models by changing some of the constraints slightly. In the MIP model (15), we only introduce one variable for every set  $\hat{e} \in \hat{E}$  of  $k_{\hat{e}}$  parallel fibers. We adjust the capacity constraints (17) to

$$\forall \hat{e} \in \hat{E} : \sum_{s \in N} z_{se} \leq k_{\hat{e}} * C \quad (23)$$

The rest of the model is not affected, and the number of constraints and variables is the same as in the case without multiple fibers.

The second phase of our decomposition is no longer a standard graph coloring problem, as on a set of parallel fibers we can use the same wavelength up to  $k_{\hat{e}}$  times. Our constraint model is only slightly affected. We have to replace the alldifferent constraint with a global cardinality constraint (gcc)[11, 13], which allows values to be used repeatedly. The variables and their domains don't need to be modified, and for every connection which uses multiple fibers we have to use gcc instead of alldifferent constraints. Note that we can no longer model the problem with disequality constraints alone, nor can we use a single some-different constraint. The explanation part of the program is not affected, the QuickXplain procedure works for any constraint network. In the experiments below we have not included scenarios with multiple fibers, as we did not find realistic network scenarios using them in the literature.

## 6 Experimental Results

Most of the published results on the RWA problem use randomly generated demands on a few given network structures. We also use this approach and generate given numbers of demands between randomly chosen source and sink nodes. Multiple demands between the same nodes are allowed, but source and sink must be different.

### 6.1 Fixed Network Structure

In the literature we found four actual optical network topologies used in experiments. Their size is quite small, ranging from 14 to 27 nodes.

**nsf** 14 nodes, 42 edges

**eon** 20 nodes, 78 edges

**mci** 19 nodes, 64 edges

**brezil** 27 nodes, 140 edges

We explored all combinations of number of demands (100-800 demands in increments of 50, 15 cases) and available wavelengths (5-50 in increments of 5, 10 cases) for the four networks, and created 100 random problems for each combination. This created 60000 problems ranging over a variety of typical scenarios. In many cases, all demands can be accepted, as there are enough frequencies available. At the other

extreme, when there are many demands for few frequencies, most demands between far-distant nodes will be dropped, and only demands which can be satisfied with a short route will be used. This is an artifact of the objective function which does not reward the acceptance of long-distance demands. In between the two extreme cases, difficult optimization problems are found where most, but not all demands can be accepted and near-optimal solutions are important for customer satisfaction and revenue generation.

Table 1 shows the distribution of outcomes over all 60000 test cases considered. Only 88 (0.14%) are infeasible, the vast majority is solved to optimality. In more than 98% of the cases the first search routine using only  $a$  units of credit finds a feasible solution. There are very few instances where complete search is required. As graph coloring problems, these instances do not seem to be very hard, given the structural information we can exploit.

**Table 1.** Overall Distribution of Solutions

Type	Technique	Count
Infeasible	clique	50
	preassign	38
Feasible	credit total	59962
	of that, credit $a$ units	58861
	of that, credit $a^2$ units	940
	of that, credit $a^3$ units	161
	complete search, BC alldifferent	25
	complete search, GAC alldifferent	12

Table 2 shows the result for selected, interesting parameter combinations. The entries summarize the results over 100 runs with the same parameters, but different random seeds. The column *Opt.* tells how many solutions were proven optimal. The columns *Avg LP*, *Avg MIP* and *Avg FD* show the average cost obtained by the LP relaxation of the MIP model (15), the MIP model itself and the final number of accepted demands obtained by the finite domain solver. The LP relaxation already is a very good approximation of the total cost, for the examples shown the LP and MIP cost coincide, and even on the full set of tests the LP cost is very tight, the MIP-LP gap never exceeds 0.94. The next column, *Max Gap*, shows the largest gap between MIP and FD solution, i.e. the number of demands removed due to infeasibility of the graph coloring model. This value never exceeds 2 in the examples shown, it never exceeds 4 in any of the tests run. We then show average and maximal total run times on a Windows XP laptop with a 2.4GHz processor and 2GB of memory. Results were obtained using ECLiPSe 6.0 [19] with the eplex library [15] for the Coin-OR [10] CLP/CBC MIP solver.

In table 3 we compare our results (Hybrid Model) to the Full MIP model (8) presented in [5]. One can see that the difference in solution quality is minimal, but the times required for the full model (again, using ECLiPSe 6.0 with eplex and the Coin-OR CLP/CBC solver) are much higher, especially for larger network size and/or large number of demands. Note that for the brazil network with 700 demands and 25 frequen-

**Table 2.** Selected Examples (100 Runs Each)

Network	Dem.	$\lambda$	Opt.	Avg LP	Avg MIP	Avg FD	Max Gap	Avg Time	Max Time
brezil	500	15	98	483.86	483.86	483.84	1.00	0.92	1.34
brezil	600	20	100	590.96	590.96	590.96	0.00	1.00	1.34
brezil	700	20	100	672.53	672.53	672.53	0.00	1.19	1.78
brezil	800	25	99	781.39	781.39	781.37	2.00	1.44	11.47
eon	500	20	100	471.56	471.56	471.56	0.00	0.65	0.77
eon	600	25	100	574.80	574.80	574.80	0.00	0.82	1.13
eon	700	30	100	677.35	677.35	677.35	0.00	1.05	1.81
eon	800	35	100	779.17	779.17	779.17	0.00	1.28	1.94
mci	500	25	100	486.38	486.38	486.38	0.00	0.80	2.28
mci	600	30	100	585.18	585.18	585.18	0.00	1.27	29.81
mci	700	35	100	684.00	684.00	684.00	0.00	1.30	3.53
mci	800	40	100	782.86	782.86	782.86	0.00	1.68	5.21
nsf	500	35	100	495.20	495.20	495.20	0.00	0.50	0.60
nsf	600	40	100	588.63	588.63	588.63	0.00	0.66	0.98
nsf	700	45	100	678.44	678.44	678.44	0.00	0.86	1.35
nsf	800	45	100	727.15	727.15	727.15	0.00	0.95	1.56

cies (results are shown in italics), only 99 of the MIP models were solved. One problem instance did not terminate within 5 days of execution.

## 6.2 Increasing Number of Demands

Table 4 shows results for another series of tests, where we increase the number of demands for the eon network, increasing at the same time the number of available frequencies so that over 90%, but below 100% of the demands can be accepted. We split the reported run-times into the average and maximum time needed for phase 1 (MIP), and phase 2 (FD). One can see that the time for phase 1 is not affected, as the MIP model (15) does depend neither on the number of demands, nor on the number of frequencies available. The average time for phase 2 slowly increase with problem size, while the number of optimal solutions stays very high (99-100%). The time for finding and exploiting the explanation in an infeasible scenario increases significantly, but not prohibitively.

## 6.3 Random Networks

We also wanted to check the stability of the proposed method for larger network sizes. For this we generated random network structures with 30 to 100 nodes and an average edge density of 0.25. We tested these for randomly generated problems of 500 demands and 30 frequencies, with 100 instances for each problem case.

Table 5 shows the results, again splitting the execution times into the MIP and FD components. One can see that with increasing network size the MIP solution times start to dominate. Unfortunately, the LP solver times also increase proportionately, which

**Table 3.** Compared to MIP Model for Complete Problem

Network	Dem.	$\lambda$	Opt.	Hybrid Model			Full MIP		
				Avg FD	Avg Time	Max Time	Avg Opt	Avg Time	Max Time
brezil	500	15	98	483.84	0.92	1.34	483.86	1218.40	14103.84
brezil	600	20	100	590.96	1.00	1.34	590.96	6076.81	87767.95
brezil	700	25	98	695.48	1.01	1.80	695.48	13623.15	78463.89
brezil	800	25	99	781.37	1.44	11.47	781.39	7567.68	15456.50
eon	500	20	100	471.56	0.65	0.77	471.56	352.21	585.45
eon	600	25	100	574.80	0.82	1.13	574.80	1411.67	2877.88
eon	700	30	100	677.35	1.05	1.81	677.35	1727.52	3568.13
eon	800	35	100	779.17	1.28	1.94	779.17	2485.64	4116.11
mci	500	25	100	486.38	0.80	2.28	486.38	1023.16	1664.31
mci	600	30	100	585.18	1.27	29.81	585.18	1621.30	2895.88
mci	700	35	100	684.00	1.30	3.53	684.00	1987.23	3428.41
mci	800	40	100	782.86	1.68	5.21	782.86	2316.88	4402.44
nsf	500	35	100	495.20	0.50	0.60	495.20	82.85	173.19
nsf	600	40	100	588.63	0.66	0.98	588.63	155.90	373.63
nsf	700	45	100	678.44	0.86	1.35	678.44	205.82	586.61
nsf	800	45	100	727.15	0.95	1.56	727.15	173.53	410.97

means that just obtaining an upper bound with the model shown becomes expensive when we consider more than 100 nodes. In order to extend the problem size further, we may have to hybridize the first phase of the algorithm as well, along the lines of [9], if we want to find proven optimal solutions.

## 7 Future Work

There are multiple directions in which the current work could be extended. We have only considered the demand acceptance variant of the RWA problem. This has the advantage that the LP relaxation of the model is an extremely good upper bound. The ILP model discussed in [4] for the static design problem has a much weaker LP relaxation. In that case, our resource-constrained relaxation might help to obtain more realistic lower bounds on the number of required frequencies.

We also so far have only considered the directed network variant of the problem. It seems straightforward to extend the model to handle the undirected case discussed in [4] as well.

At the moment, we do not attempt to generate minimal explanations, or indeed a minimal set of explanations, as we are only interested in them to suggest a further relaxation of the demand acceptance problem. By creating a more compact representation, we might be able to feed them as no-good constraints into the first phase of the algorithm, generating a complete, hybrid algorithm for the RWA problem.

**Table 4.** Increasing Number of Demands

Network	Dem.	$\lambda$	Opt.	Avg LP	Avg MIP	Avg FD	Max Gap	Avg MIP Time	Max MIP Time	Avg FD Time	Max FD Time
eon	800	30	100	741.78	741.78	741.78	0.00	0.15	0.17	0.83	1.61
eon	900	40	100	880.59	880.59	880.59	0.00	0.14	0.16	1.18	2.17
eon	1000	40	100	950.36	950.36	950.36	0.00	0.15	0.17	1.37	3.42
eon	1100	50	100	1082.61	1082.61	1082.61	0.00	0.14	0.16	1.71	2.83
eon	1200	50	100	1156.38	1156.38	1156.38	0.00	0.15	0.17	2.07	5.92
eon	1300	50	100	1219.82	1219.82	1219.82	0.00	0.16	0.17	2.22	5.24
eon	1400	60	100	1361.47	1361.47	1361.47	0.00	0.15	0.16	2.92	4.94
eon	1500	60	99	1428.78	1428.78	1428.77	1.00	0.15	0.17	4.22	106.97
eon	1600	70	100	1565.90	1565.90	1565.90	0.00	0.15	0.16	3.89	8.48
eon	1700	70	100	1637.47	1637.47	1637.47	0.00	0.16	0.17	4.58	13.59
eon	1800	80	100	1769.86	1769.86	1769.86	0.00	0.15	0.16	5.19	8.81
eon	1900	80	99	1844.46	1844.46	1844.45	1.00	0.15	0.17	7.23	163.41
eon	2000	90	100	1972.66	1972.66	1972.66	0.00	0.15	0.17	6.34	9.61

**Table 5.** Random Networks (Edge Density 0.25, 100 Runs Each)

Network	Dem.	$\lambda$	Opt.	Avg LP	Avg MIP	Avg FD	Avg MIP Time	Max MIP Time	Avg FD Time	Max FD Time
r30	500	30	100	391.82	391.82	391.82	0.45	0.55	0.12	0.16
r40	500	30	100	424.58	424.58	424.58	1.07	1.23	0.14	0.17
r50	500	30	100	437.69	437.69	437.69	2.13	2.38	0.09	0.13
r60	500	30	100	447.21	447.21	447.21	3.92	4.34	0.08	0.16
r70	500	30	100	453.41	453.41	453.41	6.78	7.50	0.10	0.17
r80	500	30	100	457.65	457.65	457.65	10.75	11.95	0.10	0.17
r90	500	30	100	464.69	464.69	464.69	16.08	17.45	0.08	0.22
r100	500	30	100	466.67	466.67	466.67	22.74	25.22	0.09	0.25

## 8 Conclusion

In this paper we have described a hybrid combination of ILP and constraint programming to solve the demand acceptance variant of the routing and wavelength assignment (RWA) problem. We have shown that decomposing the problem into a resource-constrained, optimized routing problem and a graph coloring problem works very well, producing either proven optimal or near optimal solutions for all cases tested. This method out-performs a full MIP model by orders of magnitude, making the proposed method an efficient solution for realistic network sizes.

## Acknowledgment

We want to thank Paul Davern and Hadrien Cambazard for helpful comments on a draft of the paper.

## References

1. Krzysztof R. Apt and Mark Wallace. *Constraint Logic Programming using ECLiPSe*. Cambridge University Press, New York, NY, USA, 2007.
2. Dhritiman Banerjee and Biswanath Mukherjee. A practical approach for routing and wavelength assignment in large wavelength-routed optical networks. *IEEE Journal on Selected Areas in Communications*, 14(5):903–908, June 1996.
3. N. Beldiceanu, E. Bourreau, P. Chan, and D. Rivreau. Partial search strategy in CHIP. In *2nd International Conference on Metaheuristics MIC97*, Sophia Antipolis, France, 1997.
4. Brigitte Jaumard, Christophe Meyer, and Babacar Thiongane. ILP formulations for the routing and wavelength assignment problem: Symmetric systems. In M. Resende and P. Pardalos, editors, *Handbook of Optimization in Telecommunications*, pages 637–677. Springer, 2006.
5. Brigitte Jaumard, Christophe Meyer, and Babacar Thiongane. Comparison of ILP formulations for the RWA problem. *Optical Switching and Networking*, 4(3-4):157–172, 2007.
6. Brigitte Jaumard, Christophe Meyer, and Babacar Thiongane. On column generation formulations for the RWA problem. *Discrete Applied Mathematics*, 157:1291–1308, March 2009.
7. Ulrich Junker. Quickxplain: Conflict detection for arbitrary constraint propagation algorithms. In *IJCAI'01 Workshop on Modelling and Solving problems with constraints (CONS-1)*, Seattle, WA, USA, August 2001.
8. Jonathan Lever. A local search/constraint propagation hybrid for a network routing problem. *International Journal on Artificial Intelligence Tools*, 14(1-2):43–60, 2005.
9. V. Liatsos, S. Novello, and H. El Sakkout. A probe backtrack search algorithm for network routing. In *Proceedings of the Third International Workshop on Cooperative Solvers in Constraint Programming, CoSolv'03*, Kinsale, Ireland, September 2003.
10. Robin Lougee-Heimer. The common optimization interface for operations research. *IBM Journal of Research and Development*, 47:57–66, January 2003.
11. Claude-Guy Quimper. *Efficient Propagators for Global Constraints*. PhD thesis, University of Waterloo, 2006.
12. Rajiv Ramaswami and Kumar N. Sivarajan. Routing and wavelength assignment in all-optical networks. *IEEE/ACM Trans. Netw.*, 3(5):489–500, 1995.
13. Jean-Charles Régin. Generalized arc consistency for global cardinality constraint. In *AAAI/IAAI, Vol. 1*, pages 209–215, 1996.
14. Yossi Richter, Ari Freund, and Yehuda Naveh. Generalizing alldifferent: The somedifferent constraint. In *Principles and Practice of Constraint Programming - CP 2006*, pages 468–483, Nantes, France, 2006.
15. Kish Shen and Joachim Schimpf. Eplex: Harnessing mathematical programming solvers for constraint logic programming. In Peter van Beek, editor, *CP*, volume 3709 of *Lecture Notes in Computer Science*, pages 622–636. Springer, 2005.
16. Helmut Simonis. Constraint applications in networks. In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*. Elsevier, 2006.
17. Barbara M. Smith. Symmetry and search in a network design problem. In Roman Barták and Michela Milano, editors, *CPAIOR*, volume 3524 of *Lecture Notes in Computer Science*, pages 336–350. Springer, 2005.
18. Willem Jan van Hoeve. The alldifferent constraint: A survey. *CoRR*, cs.PL/0105015, 2001.
19. M. Wallace, S. Novello, and J. Schimpf. ECLiPSe : A platform for constraint logic programming. *ICL Systems Journal*, 12(1), May 1997.
20. Hui Zang, Jason P. Jue, and Biswanath Mukherjee. A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks. *Optical Networks Magazine*, pages 47–60, January 2000.