

Full Design of Robust Optical Networks

Arne Glenstrup* Christian Fenger† Thomas Stidsen‡

Abstract

We present here a full multifibre optical WDM network design problem formulated as an integer linear problem. The design problem consists in laying out ducts, fibres, routes, and wavelengths, given a set of nodes, the duct and fibre prices, and the traffic demands. We compare different methods for solving the design problem. These are integer linear programming, simulated annealing, and simulated allocation. We find that integer linear programming is useful for benchmarking other algorithms on small networks that consist of less than 7 nodes. For larger networks that cannot be handled by integer linear program solvers, we find that simulated allocation is more promising than simulated annealing. Further we include path protection in the problem formulation and solution.

1 Introduction

As new optical WDM networks are being deployed and competition is increasing, a proper design of the networks is becoming increasingly important. In the mesh network topology design presented here, the task is to determine the position of links between a given set of nodes, satisfying a traffic matrix and minimising an objective function. Genetic algorithms have been used for this problem [1], but typical network sizes have been 20 nodes or less; in this paper we treat networks of size up to 73 nodes, but with preimposed restrictions in allowed links between node pairs. The objective function used here is the cost of a link, computed from the deployment cost of ducts and fibres, and the routing and wavelength assignment problem is solved as a side effect. We assume no wavelength converters, whereby the wavelength continuity constraint applies.

We begin by formulating the problem in a integer linear formulation to define it mathematically. This formulation will also be used when the design task is solved by integer linear programming using the Branch and Bound algorithm in the CPLEX-package. However, for larger problems this method becomes too slow, whereby heuristic methods are used. We use simulated annealing and simulated allocation [4]. We compare all the methods for different

*panic@tele.dtu.dk, Department of Telecommunication, Technical University of Denmark

†cf@com.dtu.dk, Research Center COM, Technical University of Denmark

‡tks@imm.dtu.dk, Department of Mathematical Modelling, Technical University of Denmark

networks sizes, for different numbers of wavelengths in a fibre, with and without protection. All three methods work on a set of possible paths between the nodes; this set of paths has been chosen by a smart path-selection algorithm. This preprocessing is essential so to minimise the solution space for the heuristics.

2 Problem Formulation

We consider a network consisting of a number of nodes and E links, cf. Figure 1. Each link is

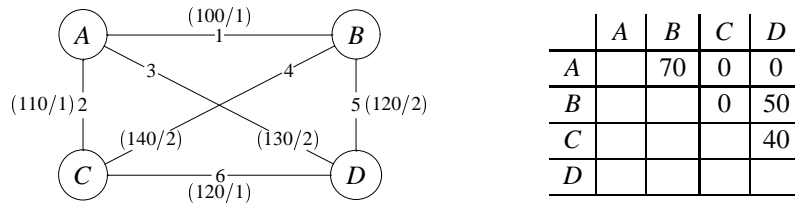


Figure 1: WDM network example with 6 links and a traffic demand matrix containing 3 demands. Each links is numbered and labeled with (duct/fibre) cost

assigned a *duct cost* and a *fibre cost*; nodes are assumed to have infinite switch capacity, but are not assigned any cost. Along each link a number of fibres can be deployed, but only if a duct is constructed. Thus the cost of deploying $x > 0$ fibres is an affine function $cx + f$, where f is the cost of the duct and c is the cost of a fibre.

We assume given D traffic demands, which are assumed symmetric. The traffic demand matrix lists the demand between node pairs in number of *lightpaths*, where a lightpath is a combination of set of links connecting the nodes, and a wavelength to transmit on.

Each fibre can carry C different wavelengths, and as we assume no wavelength conversion in nodes, we impose the wavelength continuity constraint: A path between two nodes traversing several links must use the same wavelength on each of these links. Naturally, if two different paths use the same wavelength on a link, they must use different fibres on this link.

The *nominal design problem* is simply to satisfy the demands by deploying ducts and fibres at minimal cost, whereas the *path protection design problem* additionally requires that if any single link fails, all demands supplied by a lightpath using that link can be supplied by other, spare lightpaths that are otherwise unused.

3 Integer Linear Formulations

The purpose of this section is twofold. First it defines the mesh network topology design problems mathematically, that is as integer linear formulations, and second it serves as input for solving the design problems as integer linear programs. A number of articles have been written about these integer linear programs, e.g. [2] and [3].

3.1 Nominal design

Below we define the nominal mesh network topology design problem, that is to determine the position of ducts and fibres given the nodes, cost of ducts and fibres, the traffic demand between the nodes, and a set of possible paths between the nodes.

indices:

$d = 1, \dots, D$	demands (between pairs of nodes)
$j = 1, \dots, m(d)$	paths for flows realizing demand d
$c = 1, \dots, C$	colors (wavelengths) available on the fibers
$e = 1, \dots, E$	links

constants:

h_d	volume of demand d to be realized, expressed as the number of light-paths
a_{edj}	link-path incidence coefficient: $a_{edj} = 1$ if link e belongs to path j realizing demand d , $a_{edj} = 0$ otherwise
c_e	cost per fibre of link e
f_e	cost for a duct of link e

variables (all non-negative integers):

φ_{djc}	flow (number of light-paths) realizing demand d in color c on path j
n_{ce}	number of times the color c is used on link e (auxiliary ¹)
y_e	capacity of link e expressed in the number of fibers (auxiliary)
σ_e	binary variable equal one if duct is present at link e , otherwise zero (binary, auxiliary)

objective:

minimize $C(\vec{y}, \vec{\sigma}) = \sum_e c_e y_e + f_e \sigma_e$

constraints:

$\sum_c \varphi_{djc} = h_d$	$d = 1, \dots, D$	satisfy demands
$\sum_d \sum_j a_{edj} \varphi_{djc} = n_{ce}$	$c = 1, \dots, C, e = 1, \dots, E$	compute n_{ce}
$y_e \geq n_{ce}$	$c = 1, \dots, C, e = 1, \dots, E$	ensure sufficient capacity
$y_e > 0 \Rightarrow \sigma_e > 0$	$e = 1, \dots, E$	require ducts for fibres

Links				Demands				Paths								
e	between	f_e	c_e	d	between	h_d	$m(d)$	d	j	a_{edj}						
1	A-B	100	1	1	A-B	70	2	1	1	1	0	0	0	0	0	0
2	A-C	110	1	2	B-D	50	3	2	2	0	1	0	1	0	0	0
3	A-D	130	2	3	C-D	40	2	2	1	0	0	0	0	1	0	0
4	B-C	140	2					2	2	1	0	1	0	0	0	0
5	B-D	120	2					3	3	0	0	0	1	0	1	0
6	C-D	120	1					3	1	0	0	0	0	0	0	1
									2	0	1	1	0	0	0	0
										$e = 1$	2	3	4	5	6	

Figure 2: WDM network example represented by integer linear parameters

For each demand d between a node pair, we compute $m(d)$ possible paths that connect the nodes. The constants for the example in Figure 1 are shown in Figure 2. The last constraint, for requiring a duct if any fibres are present, can be stated as $\sigma_e \cdot M \geq y_e$, where M is chosen such that $M > y_e$ for all e , i.e. $M > \sum_d h_d$.

3.2 Protection design

Below we define the protection mesh network topology design problem, where the protection method is path protection of single-link failures. We have chosen to use this protection method because the protection paths do not necessarily become longer than the primary paths, which besides the advantages of short paths also leads to less spare capacity usage than for example link protection of single-link failures. The drawback of using path protection is that all links in the protection path and primary path must be different, which constrains the choices of paths.

We introduce $S = E + 1$ failure situations, which in this case correspond to all the situations where one link is broken, and the normal situation.

indices:

$d = 1, \dots, D$	demands (between pairs of nodes)
$j = 1, \dots, m(d)$	paths for flows realizing demand d
$c = 1, \dots, C$	colors (wavelengths) available on the fibers
$e = 1, \dots, E$	links
$s = 0, \dots, S$	failure situations, $s = 0$ corresponds to the nominal state.
$k = 1, \dots, l(d, j)$	backup paths for protecting nominal flow realizing demand d on path j
$g = 1, \dots, C$	backup colors

¹By auxiliary variables we mean variables which can be directly calculated based on the decision variables, i.e. φ_{djc}

constants:

h_{ds}	volume of demand d to be realized in situation s , expressed as the number of light-paths
a_{edj}	link-path incidence coefficient: $a_{edj} = 1$ if link e belongs to path j realizing demand d , $a_{edj} = 0$ otherwise
α_{es}	binary link failure coefficient: $e_s = 0$ if link e is failed in situation s , $e_s = 1$ if it works
$\delta_{djs} = \prod_{e:a_{edj}=1} \alpha_{es}$	path failure coefficient: $\delta_{djs} = 0$ if path j of demand d is failed in situation s , $\delta_{djs} = 1$ otherwise
b_{edjk}	link-path incidence coefficient: $b_{edjk} = 1$ if link e belongs to backup path k protecting path j of demand d , $b_{edjk} = 0$ otherwise
c_e	cost per fibre of link e
f_e	for a duct of link e

variables (all non-negative integers):

Φ_{djc}	flow (number of light-paths) realizing demand d in color c on path j
Ψ_{djkcg}	backup flow on path k in color g protecting nominal flow Φ_{djc} on path j
ϵ_{djkcg}	protection flow allocation variable: $\epsilon_{djkcg} = 1$ if colour c on path j supplying demand d is backed up by colour g on path k (binary)
n_{ces}	number of times the color c is used on link e in situation s (auxiliary)
y_e	capacity of link e expressed in the number of fibers (auxiliary)
σ_e	variable equal one if duct is present at link e , otherwise zero (binary)

objective:

minimize $C(\vec{y}, \vec{\sigma}) = \sum_e c_e y_e + f_e \sigma_e$

constraints:

$$\begin{array}{lll}
\sum_j \delta_{djs} \sum_c \phi_{djc} = h_{ds} & d = 1, \dots, D, s = 0, 1, \dots, S & \text{satisfy demands} \\
\sum_k \sum_g \varepsilon_{djkcg} = 1 & d = 1, \dots, D, j = 1, \dots, m(d) & \text{one path per backup path} \\
& c = 1, \dots, C & \\
\Psi_{djkcg} > 0 \Rightarrow \varepsilon_{djkcg} > 0 & d = 1, \dots, D, j = 1, \dots, m(d) & \text{compute } \varepsilon_{djkcg} \\
& k = 1, \dots, l(d, j), c, g = 1, \dots, C & \\
\sum_k \sum_g \Psi_{djkcg} = \phi_{djc} & d = 1, \dots, D, j = 1, \dots, m(d) & \text{satisfy backup demands} \\
& c = 1, \dots, C & \\
\sum_d \sum_j (\delta_{djs} a_{edj} \phi_{djc} + (1 - \delta_{djs}) \sum_k \sum_g b_{edjk} \Psi_{djkcg}) = n_{ces} & & \text{compute } n_{ces} \\
& e = 1, \dots, E, c = 1, \dots, C & \\
& s = 1, \dots, S & \\
y_e \geq n_{ce} & c = 1, \dots, C, e = 1, \dots, E & \text{ensure sufficient capacity} \\
& s = 1, \dots, S & \\
y_e > 0 \Rightarrow \sigma_e > 0 & e = 1, \dots, E & \text{require ducts for fibres}
\end{array}$$

Again, the constraint $y_e > 0 \Rightarrow \sigma_e > 0$ can be rewritten as $\sigma_e \cdot M \geq y_e$, where M is chosen such that $M > y_e$ for all e , i.e. $M > 2 \sum_d h_d$, and $\Psi_{djkcg} > 0 \Rightarrow \varepsilon_{djkcg} > 0$ can be rewritten as $\Psi_{djkcg} \leq \varepsilon_{djkcg} h_d$.

3.3 Problem size

By looking at the variables and equations above, we find that the complexity of the problems are given by

Problem	Variables	Equations
Nominal design	$O(CE + CDJ)$	$O(CE + D)$
Protection design	$O(CE^2 + C^2DJ^2)$	$O(C^2DJ^2 + CE^2)$

where J is the maximum number of paths for any demand. In practice, J can be considered constant (usually 8 or less).

4 Heuristic Design Methods

The complexity of the problem causes the integer linear program to grow quickly to sizes which the standard solvers (CPLEX) cannot solve to optimality within reasonable time. If we want to optimize practical networks, we have to rely on heuristics of some sort.

As in the integer linear programs, a solution consists of simple choices of which paths and wavelengths to use for each demand. Each pair of nodes demands a number of connections, which is established through lightpaths. Given a (possibly partial) solution with one or more lightpaths assigned, corresponding to the solution variables ϕ_{djc} or Ψ_{djkcg} and ε_{djkcg} , the corresponding layout of the network can be calculated as the auxiliary variables \vec{y} and $\vec{\sigma}$ are calculated for the nominal design problem and the protection design problem.

In this section we will describe the two types of metaheuristics:

- Simulated annealing, a well known metaheuristic which is widely used and hence will only be briefly described.
- Simulated allocation, a lesser known metaheuristic which has been applied to similar optimization problems. This algorithm will hence be more thoroughly described.

4.1 Simulated annealing

The standard layout of a simple simulated annealing algorithm for minimising solution cost is given in Figure 3. We choose start and stop temperatures T_0 , T_{stop} , and time limit t_{stop} , and then let constant $\gamma = t_{stop}^{-1} \log(T_0/T_{stop})$ such that $T_{stop} = T_0 \cdot e^{-\gamma t_{stop}}$.

```

choose a random  $x$  in solution space
 $cost_{min} := cost(x)$ 
 $x_{min} := x$ 
 $T := T_0$  /* temperature */
 $t := 0$  /* real time timer variable */
repeat
  for  $i := 1$  to Markov length do
    choose an  $x'$  in the neighborhood of  $x$ 
     $\delta := cost(x') - cost(x)$ 
    choose a random  $r \in [0, 1[$ 
    if  $\delta < 0$  or  $r < e^{-\delta/T}$  then
       $x := x'$ 
      if  $cost(x) < cost_{min}$  then
         $cost_{min} := cost(x)$ 
         $x_{min} := x$ 
       $T := T_0 \cdot e^{-\gamma t}$ 
    until  $t \geq t_{stop}$ 

```

Figure 3: Simulated annealing core algorithm

Choosing the initial x in the solution space is a simple random (with linear distribution) allocation of the necessary lightpaths. For each temperature, T , value the parameter, *Markov length*, determines how many solutions will be searched. In all the Simulated Annealing runs we have used the value 60. The procedure for selecting x' from the neighborhood of x is somewhat more complicated. Two types of neighborhoods have been tested:

- Removal of one single lightpath and random re-assignment.
- Removal of all lightpaths through one link and random re-assignment.

The first procedure turns out to give unsatisfactory results, because of the high duct costs which have a strong influence on the overall cost of a solution. For this reason, the second type performs better.

4.2 Simulated allocation

Simulated allocation is a new metaheuristic, invented by Pióro [4] for use in optimization of telecommunication networks. The basic simulated allocation algorithm is shown in Figure 4.

```
let  $x$  be an unallocated network
 $cost_{min} := \infty$ 
 $t := 0$           /* real time timer variable */
repeat
  choose a random  $r \in [0, 1[$ 
  if  $r < q(x)$  then
    allocate and add one or more lightpaths to  $x$ 
  else
    disconnect a lightpath or link in  $x$ 
  if  $x$  is fully allocated and  $cost(x) < cost_{min}$  then
     $cost_{min} := cost(x)$ 
     $x_{min} := x$ 
  if  $x$  is fully allocated or  $cost(x) > cost_{min}$  then
    bulk disconnect many links in  $x$ 
until  $t \geq t_{stop}$ 
```

Figure 4: Simulated allocation core algorithm

Contrary to simulated annealing, simulated allocation starts with an empty solution. Instead a solution is gradually constructed. Inside the main loop, a probabilistic choice is made whether to allocate a lightpath or disconnect one or more lightpaths. Allocation simply means adding to the current partial solution x one lightpath, and disconnection is the opposite, i.e. removing one or more lightpaths from the solution. In order to reach a complete solution, the probability for choosing allocation $q(x)$ should be sufficiently high. Whenever a complete solution is reached, it is compared with the current best solution. If the solution is complete or the partial solution is worse than the current best (i.e. it can never improve the current best), a large number of lightpaths are disconnected so that a different part of the solution space can be reached. The algorithm is terminated after a predetermined number of seconds, t_{stop} .

The problem dependent procedures are:

Allocation of one single lightpath to the current solution. It chooses the best single lightpath from a group of these.

Disconnection of all lightpaths passing through a certain link, or of only one lightpath.

Bulk disconnection of all lightpaths through a certain set of links. The set of links are chosen at random with equal probability.

Simulated allocation is not as widely applicable as simulated annealing or other metaheuristics; it requires the possibility of evaluating partial solutions, i.e. solutions which are not feasible. In some ways it resembles an iterated construction algorithm, which is restarted from different

partial solutions. The point is though, that these restarts are not random, but are based on parts of previously discovered solutions.

5 Results and Discussion

5.1 Experimental design

The networks used in experiments are two real-world examples of networks in Denmark, and a small, 5-node network:

<i>network</i>	<i>nodes</i>	<i>links</i>	<i>average duct cost</i>	<i>average fibre cost</i>	<i>demands</i>	<i>demand volume</i>
A	5	10	152.0000	1.535000	10	71
B	23	29	213.5862	1.999310	21	844
C	73	165	31692.84	316.9284	279	1824

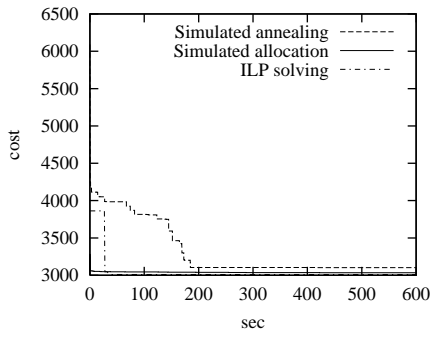
For each network, up to 3 mutually disjoint shortest paths and up to 5 shortest paths are generated for each pair of demand nodes. Each (task–network–wavelengths–method) configuration was run five times for 10 minutes. The results are shown in Table 1. Figures 5 and 6 show ex-

<i>Task</i>	<i>Net</i>	<i>w</i>	<i>Simulated allocation</i>		<i>Simulated annealing</i>		<i>ILP solving</i>	
			<i>average</i>	<i>std.dev.</i>	<i>average</i>	<i>std.dev.</i>	<i>result</i>	<i>status</i>
nominal	A	4	444.00	0.00	444.60	1.79	444.00	optimal
		16	413.00	0.00	415.20	0.89	413.00	optimal
		256	404.00	0.00	404.00	0.00	—	timeout
	B	4	4384.40	1.49	4453.50	19.12	4377.68	optimal
		16	3035.02	3.26	3103.25	13.33	3010.46	timeout
		256	2605.78	1.01	2674.70	6.96	—	timeout
	C	4	2053019.09	19371.23	2089457.59	144781.07	2847760.00	timeout
		16	1898782.67	4598.40	1979889.76	97755.62	—	timeout
		256	1842618.71	4984.74	1954155.21	55768.88	—	timeout
protection	A	4	579.00	0.00	589.00	1.41	—	timeout
		16	531.00	0.00	537.00	1.41	—	timeout
		256	515.00	0.00	516.80	1.67	—	memout
	B	4	9614.92	7.15	9667.46	11.70	—	timeout
		16	6074.30	3.74	6083.91	4.15	—	memout
		256	4935.40	0.96	4948.83	3.11	—	memout
	C	4	3604097.35	32069.91	3705365.99	28938.54	—	timeout
		16	3341704.02	14458.96	3409685.70	48311.27	—	memout
		256	3246105.29	33566.56	3315111.54	65634.92	—	memout

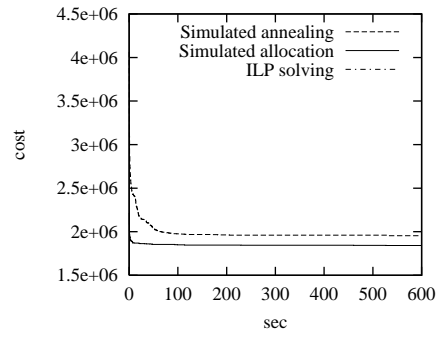
Table 1: Experimental results

amples of progress for the current best solution during optimisation (pointwise averages over the multiple runs).

The results indicate that simulated allocation is in general slightly better than simulated annealing. Furthermore, simulated allocation obtains almost optimal results in the cases where

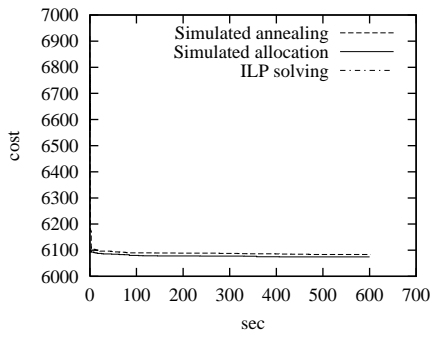


(a) Network B, 16 wavelengths

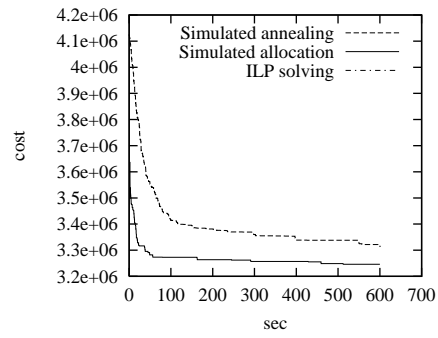


(b) Network C, 256 wavelengths

Figure 5: Nominal case



(a) Network B, 16 wavelengths



(b) Network C, 256 wavelengths

Figure 6: Protection case

the ILP solver reaches a result within the time limit. We have also conducted experiments for 15-minute runs, and they indicate that only small improvements are gained when the heuristics are run for more than 5 minutes on these network problems. The inclusion of protection raised the cost of the net by 30% to 120%.

6 Summary and further work

In this paper we have presented methods to solve the mesh network topology design problem for deployment of ducts and fibres and the routing and wavelength assignment problem both in the case of nominal design and for protection. We formulated the problems as integer linear formulations and solved them using the heuristics, simulated annealing and simulated allocation, and integer linear programming. For larger networks we found that integer linear programming was useless due to its exponential rise in demand of time and memory. Simulated annealing and simulated allocation were always able to find solutions, however, simulated allocation was always able to find the better solution.

In our future work we plan to include the cost of nodes, where switching and termination takes place. This type of cost takes up a major part of the budget when deploying networks and is therefore essential to include. Inclusion of node cost does complicate the problem significantly, since extra choices with respect to node capacity and functionality must be taken.

References

- [1] Sinclair, M.C., "Optical Mesh Topology Design using Node-Pair Encoding Genetic Programming", *Proc. Genetic and Evolutionary Computation Conference (GECCO-99)*, Orlando, Florida, USA, pp.1192-1197, July 1999.
- [2] Baroni, S., Bayvel, P., Gibbens, R.J. & Korotky, S.K., "Analysis and design of resilient multifiber wavelength-routed optical transport networks", *Journal of Lightwave Technology*, Vol.17 Issue.5, pp. 743-58, 1999.
- [3] Wauters, N & Demeester, P., "Design of the Optical Path Layer in Multiwavelength Cross-Connected Networks", *IEEE Journal on Selected Areas in Communications*, Vol.14 Issue.5, pp. 881-892, 1996.
- [4] Pióro, M., "Solving Multicommodity Integral Flow Problems by Simulated Allocation", *Telecommunication Systems*, Baltzer Science Publishers, Vol.7, Nos.1-3, pp.17-28, 1997.