Solutions for the SONET Ring Assignment with Capacity Constraints

Roberto Aringhieri[*] *Mauro Dell'Amico*[*]

[*] DISMI - University of Modena and Reggio Emilia
Viale Allegri 15, 42100 Reggio Emilia - Italy
Email: {aringhieri,dellamico}@unimo.it

1 Introduction

Synchronous Optical Network (SONET) in North America and Synchronous Digital Hierarchy (SDH) in Europe and Japan is the current transmission and multiplexing standard for high speed signals within the carrier infrastructure.

The typical topology of a SONET network is a collection of *local rings* or simply *ring* directly connecting a subset of customer sites or nodes. Each node sends, receives and relays messages through a device called add-drop-multiplexer (ADM). In bidirectional rings the traffic between two nodes can be sent clockwise or counterclockwise, and the volume of traffic is limited by the ring capacity, which is equal in both directions. The capacity of the bidirectional ring have to accommodate the sum of bandwidth requests between all pairs of nodes connected by the ring. Finally, all rings are connected together by a special ring called *federal ring* through an expensive equipment, the digital cross connect (DCC), which joins each local ring to the federal ring.

Designing the network, a set of technological constraints have to be taken into account: maximum length of a ring, maximum number of nodes in a ring, maximum traffic on a ring, etc. We deal with the designing problem of a SONET network in which: i) each customer has to be assigned to exactly one local ring; ii) the maximum capacity of each ring is bounded by a common value, say B; iii) the number of local rings, that is the cost of DCCs installed, has to be minimized. This problem is called SONET Ring Assignment Problem (SRAP) with capacity constraints. Fixing the number k of rings required we derive a parametric version of the problem called k-SRAP. Goldschmidt, Laugier and Olinick [4] have been shown that SRAP is $\mathcal{NP}$-complete reducing the 3-Regular Graph Bisection Problem to a special case of SRAP. They have proposed three greedy algorithms to solve SRAP.

In section 2 we present integer formulation for k-SRAP. In section 3 we discuss the greedy algorithms proposed in [4]. We present our approaches based on Tabu Search (TS) [3] methodology in sections 4. Preliminary computational results and current work are reported in section 5.

2 Integer Formulation

We introduce the following notation: let n be the number of nodes, B the common capacity bound, d_{uv} the symmetric traffic demand between the pair of nodes u and v ($u, v = 1, \ldots, n$, $u \neq v$) and $W_u = \sum_{v \neq u} d_{uv}$ the total traffic on node u ($u = 1, \ldots, n$). We use two sets of decision variables: $p_{uvr} = 1$ ($u, v = 1, \ldots, n$, $u \neq v$, $r = 1, \ldots, n$) if nodes u and v belong both to ring r, $p_{uvr} = 0$ otherwise; $x_{ur} = 1$ ($u = 1, \ldots, n$, $r = 1, \ldots, n$) if node i is assigned to ring r, $x_{ur} = 0$ otherwise.

We present a model for k-SRAP which is derived from that presented in [4].

$$k\text{-SRAP}: \max \quad \sum_{r=1}^{k}\sum_{u=1}^{n-1}\sum_{v=u+1}^{n} p_{uvr}d_{uv}$$

$$\text{s.t.} \quad \sum_{u=1}^{n} x_{ur}W_u - \sum_{u=1}^{n-1}\sum_{v=u+1}^{n} p_{uvr}d_{uv} \leq B, \qquad r = 1,\ldots,k \tag{1}$$

$$\frac{1}{2}\sum_{u=1}^{n} W_u - \sum_{r=1}^{k}\sum_{u=1}^{n-1}\sum_{v=u+1}^{n} p_{uvr}d_{uv} \leq B \tag{2}$$

$$\sum_{r=1}^{k} x_{ur} = 1, \qquad u = 1,\ldots,n \tag{3}$$

$$p_{uvr} \leq x_{ur}, \qquad r = 1,\ldots,k, \quad \{u,v = 1,\ldots,n | u < v\} \tag{4}$$

$$p_{uvr} \leq x_{vr}, \qquad r = 1,\ldots,k, \quad \{u,v = 1,\ldots,n | u < v\} \tag{5}$$

$$x_{ur}, p_{uvr} \in \{0,1\}, \qquad r = 1,\ldots,k, \quad u,v = 1,\ldots,n$$

Our version includes constraint (2) which guarantees the feasibility of the federal ring: actually the constraints (1) and (2) assure that the total traffic through the rings and the federal ring are satisfied, respectively. The constraint (3) assures that each node is assigned to only one ring. The objective function is to maximize the total traffic within the rings which is equivalent to minimize the total traffic through the federal ring.

Note that n-SRAP does not provide optimal solution for SRAP since the solution which minimizes the traffic through the federal ring not necessarily minimizes the number of rings. Although it is useful for checking the feasibility of the problem instances: in fact, if n-SRAP is unfeasible, SRAP is unfeasible too.

3 Previous work

Goldschmidt, Laugier and Olinick [4] have proposed three different greedy heuristics for solving SRAP: the *edge-based*, the *cut-based* and the *node-based* heuristics. In the edge-based heuristic, edges are ordered by non decreasing weight. At each iteration, two different rings connected by the maximum weight edge are merged if the resulting ring is feasible ($\sum_{u\in r, v\neq u} d_{uv} \leq B$). In the cut-based heuristic the two rings sharing the maximum traffic are merged if the resulting ring is feasible. Both the edge-based and the cut-based heuristics start their computation assigning each node to different ring. Finally, the node-based heuristic receives as input a tentative number of ring k, then assigns to each of the k rings a node randomly chosen. At each iteration the heuristic selects the ring with the largest unused capacity, say r, then, the unassigned node with largest traffic toward ring r is chosen and added to the ring. The node-based heuristic is repeated ten times and every time a feasible solution is computed the value of k is decreased by one.

In [4] the edge-based and cut-based heuristics are applied to a set of benchmark instances. The minimum feasible k is computed and given as input to the node-based heuristic. Note that both the edge-based and cut-based heuristics may have different behavior if different tie break rules are used. Therefore the authors propose to repeat these procedures ten times and to choose randomly how to break ties. When the heuristic solution has a value greater than the simple lower bound

$$k_{lb} = \frac{\sum_u \sum_{v\neq u} d_{uv}}{2B}, \tag{6}$$

then k-SRAP is used to try to prove the optimality of the heuristic solution.

4 Tabu Search Procedure

We introduce the following notation: let $T_r = \sum_{u=1}^{n-1} \sum_{v=u+1}^{n} p_{uvr} d_{uv}$ be the total traffic on ring r and let $BN = \max_r T_r$ be the network bottleneck. A ring r is feasible if $T_r \leq B$.

We have first defined the *node improvement* neighborhood ($N1$): $N1$ generates all possible solutions generated moving a nodes from its ring to a different one or to a new ring, such that the resulting ring (existing or new) is feasible. We have also defined useful data structures for fast solution updating.

We have used two tabu lists: the *node-list* which contains moved nodes and the *node-from-list* which contains pairs of moved nodes and their original rings. Their length is not fixed and varies during the search as described in [1]. We maintain the length of node-list smaller than the length of node-from-list: the idea is to fix a moved node for some iterations and then to forbid its return to the original ring for some others iterations.

Then, we have studied several objective functions for having a better drive of the search. The main difficulty is to take into account both the number of ring k and the bottleneck BN. We have defined the following objective functions:

$$
\begin{aligned}
z_1 &= k + \alpha \max\{0, BN - B\}, \\
z_2 &= z_1 + \begin{cases} k\,T_c & \text{if new ring } c \text{ has been created} \\ 0 & \text{otherwise} \end{cases}, \\
z_3 &= kB + \alpha BN.
\end{aligned}
$$

where α is a parameter. Function z_1 leads the search through moves which minimize k while penalizing those unfeasible. Function z_2 modify z_1 penalizing also moves which creates a new ring. Finally, function z_3 leads the search through moves which minimize k and, for those with the same k, minimizes BN.

We have observed that the expected solution improvement depends on the feasibility status of current solution. For example, if the current solution is feasible and a move creates a new feasible solution, then we are interested in minimizing k. On the contrary, if the current is unfeasible and a move creates a new unfeasible solution, then we would like to minimize BN. Therefore we have studied a multiobjective function which drives the search taking into account these different cases. Denoting with tilde $\widetilde{(\cdot)}$ the values associated with unfeasible solution, we have defined:

$$
z_4 = \begin{cases} k\,B + BN & \text{if the move starts from a feasible solution and generates a feasible one} \\ (k+1)\widetilde{BN} & \text{if the move starts from a feasible solution and generates a unfeasible one} \\ k\,B & \text{if the move starts from a unfeasible solution and generates a feasible one} \\ n\,\widetilde{BN} & \text{if the move starts from a unfeasible solution and generates a unfeasible one} \end{cases}
$$

Since $kB+BN \leq (k+1)B \leq (k+1)\widetilde{BN}$ and $kB \leq n\widetilde{BN}$, z_4 assures to drive the search from unfeasible solutions to feasible ones. Note that z_4 is a sort of extension of z_3.

Looking at not good solutions, we have observed that they are often composed by some large rings and by a ring with few nodes. The introduction of *empty minimum cardinality ring* neighborhood ($N2$) is aimed to deal with these kind of solutions. Applying $N2$ means that the nodes belonging to the minimum cardinality ring are compulsory moved to the other rings, minimizing z. The switching of $N1$ to $N2$ has done after i_{ni} not improving iterations (condition 1). To avoid the escape from promising regions, the feasibility of current solution is required before switching the neighborhoods (condition 2).

5 Preliminary Results

Computational experiments have been made using the same benchmark of instances used in [4]. The reported results concern only the 118 instances proved to be feasible by solving n-SRAP by Cplex and XPress. We have implemented the greedy randomized procedures of [4] and run each procedure with 100000 restarting. With these algorithms 111 over 118 instances are solved.

		Z1		Z2		Z3		Z4	
condition 1	condition 2	avg. ms	#	avg. ms	#	avg. ms	#	avg. ms	#
required	required	84.4	115	88.2	108	233.8	8	154.3	118
required	not required	68.6	98	56.5	91	–	6	95.2	98
not required	required	103.6	118	67.7	109	–	1	125.3	118
not required	not required	129.5	97	132.4	94	496.8	88	367.2	88

Table 1: Number of instances optimally solved and average computation time ($\alpha = 1$, $i_{ni} = 10$)

In table 1 we summarize the number of instances optimally solved by TS with the objective functions proposed in section 4. Each different TS was tested varying the two conditions for switching the neighborhoods $N1$ and $N2$. In table 2 are reported TS results for the instances not solved in [4]: the

					Z1			Z2			Z4		
instance	n	m	k_{lb}	k^*	k	iter	ms	k	iter	ms	k	iter	ms
RH.15.3	15	53	2	3	3	28	0.0				3	28	0.0
RH.50.3	50	89	4	4	4	3516	2420.0	5	117	230.0	4	541	490.0
RH.50.4	50	89	3	4	4	350	400.0	4	1054	840.0	4	423	410.0
RH.50.7	50	90	4	5	5	451	450.0	5	847	700.0	5	449	440.0
RL.25.1	25	51	3	4	4	268	50.0				4	336	70.0
RL.30.3	30	58	3	4	4	37	20.0				4	324	100.0
RL.30.6	30	56	3	4	4	78	40.0				4	86	40.0

Table 2: TS results for the instances not solved in [4] ($\alpha = 1$, $i_{ni} = 10$)

results show the effectiveness of z_1 and z_4 to drive the search while z_2 fails to find feasible solution for three instances; z_3 fails to solve all instances proposed. Finally, in table 3 the effectiveness of using the

Neighborhoods	Z1		Z2		Z3		Z4	
used	avg. ms	#	avg. ms	#	avg. ms	#	avg. ms	#
$N1$ and $N2$ with condition 2	103.6	118	67.7	109	–	1	125.3	118
$N1$ only	131.0	97	131.8	94	500.0	88	369.5	88

Table 3: Effectiveness of neighborhood $N2$ ($\alpha = 1$, $i_{ni} = 10$)

neighborhoods $N1$ and $N2$ are proved.

We have been generating harder instances for SRAP in order to experiment the Tabu Search proposed with a variety of benchmarks. We are expecting that Tabu Search using z_1 or z_2 with only condition 2 is not enough to optimally solve harder instances. Therefore we have been developing more sophisticated techniques as the Path Relinking [2] [6], the long term memory (XTS) [1] and the Scatter Search [2] [5].

References

[1] M. Dell'Amico and M. Trubian. Solution of large weighted equicut problems. *European J. Oper. Res.*, 106(2-3):500–521, 1998.

[2] F. Glover. Scatter search and path relinking. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 297–316. McGraw Hill, 1999.

[3] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, 1997.

[4] O. Goldschmidt, A. Laugier, and E. V. Olinick. Sonet/sdh ring assignment with capacity contraints. Technical report, IEOR, Berkeley University, 2000.

[5] M. Laguna. Scatter search. In P. M. Pardalos and M. G. C. Resende, editors, *Handbook of Applied Optimization*. Oxford Academic Press, 2001.

[6] M. Laguna, R. Martí, and V. Campos. Intensification and diversification with elite tabu search solutions for the linear ordering problem. *Computers Oper. Res.*, 26:1217–1230, 1999.