

A NOVEL INTEGER PROGRAMMING FORMULATION FOR THE K-SONET RING ASSIGNMENT PROBLEM

E.M. MACAMBIRA, C.N. MENESES, P.M. PARDALOS, AND M.G.C. RESENDE

ABSTRACT. We consider the problem of interconnecting a set of customer sites using SONET rings of equal capacity, which can be defined as follows: Given an undirected graph $G = (V, E)$ with nonnegative edge weight d_{uv} , $(u, v) \in E$, and two integers k and B , find a partition of the nodes of G into k subsets so that the total weight of the edges connecting the nodes in different subsets of the partition is minimized and the total weight of the edges incident to any subset of the partition is at most B . This problem, called the k -SONET Ring Assignment Problem (k -SRAP), arises in the design of optical telecommunication networks when a ring-based topology is adopted. We show that this network topology problem corresponds to a graph partitioning problem with capacity constraints and it is NP-hard. In this paper we propose a novel and compact 0-1 integer linear programming formulation for this problem. We report computational results comparing our formulation with another formulation found in the literature. The results show that our formulation outperforms the previous one.

1. INTRODUCTION

A telecommunications network is a complex web of links and nodes, and one cannot speak of a single network but rather of a hierarchy of networks. For instance, in [10] the authors present three levels of network: the tributary network, the local network and the backbone network.

The planning of a network raises various combinatorial optimization problems related to network design. Generally, these problems are solved independently for each network level. In this paper, we study a network topology problem that arises in the planning of an optical backbone network. This problem is called k -SONET Ring Assignment Problem, or k -SRAP for short.

The current standard for optical networks is denoted as SONET (Synchronous Optical NETWORK) and uses a ring-based topology. Each customer is connected to one or more rings and the entire network is made up of a collection of such rings.

The design of a based-ring topology network can have several variants. The simplest of them occurs when the network has a single ring. Another variant is the one where there are several rings. This class can be further subdivided into two subclasses: ring intersections are allowed and ring intersections are not allowed. Finally, the third class of based-ring topology network has a hierarchical structure. That is, the rings have no intersections and there is a ring that connect all of the other rings. This top-level ring is called the *federal ring*. The (optical) backbone network discussed in this paper is of this type.

Date: October 28, 2005.

Key words and phrases. Telecommunication networks, SONET rings, integer programming, graph partitioning.

AT&T Labs Research Technical Report TD-6HLLNR..

The k -SRAP is a special case of the SONET Ring Assignment Problem, or SRAP for short, and has not been studied much. Goldschmidt et al. [3] show that the SRAP can be solved by several resolutions of the k -SRAP. Due to the relevance of the SRAP in the design of fiber-optic telecommunications using SONET technology, we are motivated to study the solution of k -SRAP via integer programming. We propose a novel and compact 0-1 integer linear programming formulation for the k -SRAP.

We report computational experiments comparing our formulation with the one presented in [3]. Empirical evidence indicates that the former formulation is substantially better than the latter. It is worthwhile to note that the tested instances are the same ones tested by Goldschmidt et al. [3].

The paper is organized as follows. In Section 2 we describe the k -SRAP as a graph optimization problem, while the original formulation from [3] for this problem is discussed in Section 3. In Section 4, the novel integer linear programming formulation is presented. In Section 5, we describe the computational experiments and analyze the results. Finally, in Section 6 we draw some conclusions and discuss future directions.

2. PROBLEM DESCRIPTION

In this section we formally describe the k -SONET Ring Assignment Problem. The k -SRAP can be seen as a node-partitioning problem for a given graph G where one wants to find a partition of the nodes of G satisfying certain properties. Before describing the problem we briefly review some relevant aspects in the design of backbone network that use SONET technology and describe the SONET Ring Assignment Problem.

2.1. The SONET ring assignment problem. Consider the following graph optimization problem. We are given an undirected graph $G = (V, E)$ and a positive integer B . Associated with each edge (u, v) in E is a non-negative integer d_{uv} . A feasible solution of the problem is a partition of the nodes in G into sets V_1, V_2, \dots, V_ℓ such that:

- i: $\sum_{(u,v) \in G[V_j] \cup \delta(V_j)} d_{uv} \leq B$, for all $j \in \{1, \dots, \ell\}$ and,
- ii: $\sum_{j=1}^{\ell} \sum_{(u,v) \in \delta(V_j)} d_{uv} \leq B$,

where $G[V_j]$ is the graph induced by V_j in G and $\delta(V_j)$ is the set of edges with exactly one extremity in V_j . The goal is to find a feasible solution that minimizes the size of the partition, i.e., the value of ℓ .

Notice that a trivial necessary condition for the feasibility of an instance of this problem requires that $\sum_{v|(u,v) \in E} d_{uv} \leq B$ for each node u in V . Moreover, for any feasible instance, there can be no optimal solution in which two subsets of the partition are composed of singletons. This is because the two isolated nodes can be grouped together since, otherwise, condition (ii) would not hold for any solution. Thus, a simple upper bound for the optimal value is given by $\lceil \frac{B}{2} \rceil$.

This graph problem is also known as the SONET Ring Assignment Problem. The SRAP was investigated recently by Goldschmidt et al. [3]. In their paper the authors motivated the study of the problem by showing its relevance to the design of fiber-optic telecommunication networks using the SONET (Synchronous Optical NETWORK) or SDH (Synchronous Digital Hierarchy) technology. In this context, the nodes of graph G are associated with client sites while the edges are associated with existing traffic demand between pairs of clients. The edge weights d_{uv} measure the actual demand for communication among clients. Given a feasible solution to the graph problem stated above, the nodes in each subset of the partition form a *local ring* or simply a *ring*. Due to physical limitations imposed on the equipment used in building the network, the total demand in each local ring must

not exceed a constant B^1 . Moreover, the total demand between clients in different rings is handled by an additional ring called the *federal ring*.

The traffic routing between clients in a ring requires a special type of device, installed at each client, that is able to aggregate and disaggregate the traffic. This device is called an *Add-Drop Multiplexer* (ADM). The ADMs are specified according to their capacities as follows:

- ADM-1: capacity concentrator STM-1 (155Mbs),
- ADM-4: capacity concentrator STM-4 (622Mbs),
- ADM-16: capacity concentrator STM-16 (2.5Gbs).

The connection of a local ring to the federal ring is made possible through a device known as a *Digital Cross-Connect System* (DCS). Since the capacity of the federal ring is also bounded by B , the sum of the weights of the edges in the multicut corresponding any feasible solution cannot exceed that amount. Finally, because the DCSs are by far the most expensive equipment needed to implement a network, a basic problem in the design of low-cost SONET optical networks asks for a solution that minimizes the number of local rings. One can easily check that the graph problem discussed at the beginning of this section correctly models the latter problem.

We refer to [3] for a more detailed discussion of the architecture of SONET networks and a review of the literature on the SRAP. In their paper, the authors prove that SRAP is NP-hard.

2.2. The k -SONET ring assignment problem. The k -SRAP can be described formally as a graph optimization problem: Given an undirected graph $G = (V, E)$, with positive weights d_{uv} associated with the edges in E , and two integer values k and B , our goal is to find a partition of the nodes in G into at most $k \leq |V|$ sets V_1, V_2, \dots, V_k so that the sum of the weights of the edges connecting different sets is minimized and such that the total weight of the edges incident to any given set in the partition is less than or equal to B .

The k -SRAP has been recently investigated by Goldschmidt et al. [3] and it was shown to be NP-hard. The authors in [3] also introduced an integer programming formulation for k -SRAP. In the next section we discuss this formulation.

In terms of the network design application, k -SRAP has the same input data as SRAP with the additional parameter k . As in SRAP, we are still assigning each node to exactly one SONET local ring and connecting the rings with a federal ring. Another aspect in common is the local ring capacity. The total demand on a local ring, for both problems, is limited to the bandwidth B .

However, there are two important differences between the problems. First, k -SRAP does not constrain the capacity of the federal ring to be B . Instead, the objective is to minimize the amount of traffic that would put on the federal ring. The second difference is that we can use at most k rings in a k -SRAP solution whereas we could use up to $|V|$ rings in a SRAP solution. We refer to [3] for a more detailed discussion of how SRAP is solved by solving k -SRAP.

3. PREVIOUS WORK

In this section we present an integer linear programming formulation proposed by Goldschmidt et al. [3] for k -SRAP. Experiments with this formulation are reported in the forthcoming sections. Hereafter, the parameters n and m are used to denote the number of nodes and edges in G , respectively, while the set $\{1, \dots, n\}$ is represented by N .

¹Any ring satisfying this property is said to be a *feasible ring*.

This formulation consists of four groups of binary variables, namely x , y , p , and z . In the first one, there are n^2 variables x_{ui} for $u \in V$ and $i \in N$. The value of x_{ui} is one if and only if node u is assigned to ring i . In the second group, there are n variables each representing a possible local ring to be opened² in a feasible solution. Thus, the variable y_i is one if and only if the i -th ring is opened.

For each edge $(u, v) \in E$, where $u < v$, and every $i \in N$, the variable p_{uvi} has value one if and only if both nodes u and v are assigned to ring i . This third group of variables has mn elements. Finally, in the fourth group, we have $2mn$ z variables. Each z variable is indexed by three elements, the first two taken from $V \times V \setminus \{(u, u) \mid u \in V\}$ and the last one from N . The variable z_{uvi} is then defined to be one if and only if u is in ring i while v is not. A possible formulation for k -SRAP with these variables is given below.

$$\begin{aligned}
(1) \quad (GM) \quad & \min \sum_{u=1}^{n-1} \sum_{v=u+1}^n \sum_{i=1}^n d_{uv} z_{uvi}, \\
(2) \quad & \text{s.t.} \quad \sum_{(u,v) \in E \mid u < v} d_{uv} p_{uvi} + \sum_{(u,v) \in E} d_{uv} z_{uvi} \leq B, & \forall i \in N, \\
(3) \quad & \sum_{i \in N} y_i \leq k, \\
(4) \quad & \sum_{i=1}^n x_{ui} = 1, & \forall u \in V, \\
(5) \quad & x_{ui} - y_i \leq 0, & \forall u \in V, \forall i \in N, \\
(6) \quad & x_{ui} + x_{vi} - p_{uvi} \leq 1, & \forall (u, v) \in E, \forall i \in N, \\
(7) \quad & x_{ui} - x_{vi} - z_{uvi} \leq 0, & \forall u \in V, \forall (u, v) \in E, \forall i \in N, \\
(8) \quad & x_{ui} \in \{0, 1\}, & \forall u \in V, \forall i \in N, \\
(9) \quad & y_i \in \{0, 1\}, & \forall i \in N, \\
(10) \quad & p_{uvi} \in \{0, 1\}, & \forall (u, v) \in E, \forall i \in N, \\
(11) \quad & z_{uvi} \in \{0, 1\}, & \forall u \in V, \forall (u, v) \in E, \forall i \in N.
\end{aligned}$$

The objective (1) is to minimize the amount of traffic that would put on the federal ring. Now, let us briefly examine the constraints in formulation (GM). Constraints (2) state the maximum local ring capacity. Constraints (3) force the number of SONET rings to be at most k . Constraints (4) say that each node must belong to only one ring. Constraints (5) state that a local ring is open if at least one node is assigned to that ring. For each tuple (u, v, i) , constraints (6) say that variable p_{uvi} equals one if both nodes u and v belong to ring i , whereas constraints (7) state that variable z_{uvi} has value one if node u belongs to ring i and node v does *not* belong to ring i . Constraints (8) through (11) force the integrality of the variables x , y , p , and z , respectively.

The above formulation requires $n^2 + 3mn + 2n + 1$ constraints and $n^2 + 3mn + n$ binary variables.

4. NOVEL 0-1 ILP FORMULATION

In this section we present a novel integer linear programming formulation for k -SRAP. The purpose is to reduce the number of variables using $\Theta(n^2)$ constraints. Our objective is produce a model which is easier to solve.

²A ring is said to be *open* if there is at least one node assigned to it.

To facilitate the presentation, consider a feasible solution for the k -SRAP, where the node set V is partitioned into the sets V_1, V_2, \dots, V_k for $1 \leq k \leq n$. For every $j \in \{1, \dots, k\}$, the internal and external demands of ring j are, respectively, D_j and W_j , where

$$(12) \quad D_j = \sum_{u \in V_j} \sum_{\substack{v \in V_j \\ u < v}} d_{uv},$$

$$(13) \quad W_j = \sum_{u \in V_j} \sum_{v \notin V_j} d_{uv}.$$

Now assume that D represents the total demand and d_j is the total demand for the ring j , that is,

$$D = \sum_{(u,v) \in E} d_{uv},$$

$$d_j = D_j + W_j.$$

Using the definitions above for D_j, W_j, D , and d_j , for $j \in \{1, \dots, k\}$, the demand that passes through a federal ring can be expressed as

$$(14) \quad \bar{D} = D - \sum_{j=1}^k D_j,$$

and hence

$$(15) \quad 2 \times \bar{D} = \sum_{j=1}^k W_j.$$

Finally, using the capacity of a federal ring and equations (14) and (15) we obtain

$$(16) \quad \begin{aligned} \bar{D} \leq B &\implies D + \bar{D} \leq D + B \implies \\ &\stackrel{(14)}{\implies} \bar{D} + \bar{D} + \sum_{j=1}^k D_j \leq D + B \implies \\ &\stackrel{(15)}{\implies} \sum_{j=1}^k W_j + \sum_{j=1}^k D_j \leq D + B. \end{aligned}$$

Note that inequality (16) gives another way to express the constraint for the capacity of a federal ring, and consequently an alternative objective function can be expressed by

$$(17) \quad \min \sum_{u=1}^{n-1} \sum_{v=u+1}^n \sum_{i=1}^n d_{uv} f_{uvi} - D.$$

By carefully observing inequality (16) we can define new binary variables for the edges in the instance graph $G = (V, E)$. For each edge $(u, v) \in E$, with $u < v$, and each node $i \in N$, a variable f_{uvi} is set to zero if neither node u nor node v is assigned to ring i . In other words, for a fixed ring i , the edges with $f_{uvi} = 1$ are those which are internal to ring i or link nodes of ring i to nodes at some other ring.

As in SRAP, the model (GM) present a high level of symmetry. It is well-known that the presence of symmetry in the model spoils the performance of enumeration algorithms available for integer linear programming (for more details, see [1]). Other studies about

symmetry in integer linear programming models for combinatorial optimization problems can be found in [4] and [9].

A possible way to reduce the symmetry of the solutions searched in the branch-and-bound tree is using linear inequalities. A simple way to reduce the symmetry in model (*GM*) is to add the constraints (see e.g., [5]):

$$(18) \quad y_i \leq y_{i-1}, \quad \forall i \in N.$$

The foregoing inequality states that a ring i is opened only if ring $i - 1$ is opened.

Obviously, these inequalities are not enough to completely eliminate symmetry. Nevertheless, as we shall see in Section 5, they are very helpful in practice. Other inequalities that we have tried, like those that force the number of nodes to decrease as ring indexes increase, did not work as well in practice as the inequalities in (18). A possible explanation for the success of those inequalities is that they do not increase the density of the LP constraint matrix, which is known to harm the efficiency of the solvers that compute the linear relaxations.

Given the variables x , y , and f defined above, we get the following novel integer linear programming formulation for the k -SRAP:

$$(19) \quad (CM) \quad \min \sum_{u=1}^{n-1} \sum_{v=u+1}^n \sum_{i=1}^n d_{uv} f_{uvi} - D,$$

$$(20) \quad s.t. \quad \sum_{(u,v) \in E} d_{uv} f_{uvi} \leq B, \quad \forall i \in N,$$

$$(21) \quad \sum_{i \in N} y_i \leq k,$$

$$(22) \quad \sum_{i=1}^n x_{ui} = 1, \quad \forall u \in V,$$

$$(23) \quad x_{ui} - y_i \leq 0, \quad \forall u \in V, i \in N,$$

$$(24) \quad f_{uvi} - x_{ui} \geq 0, \quad \forall (u, v) \in E, \forall i \in N,$$

$$(25) \quad f_{uvi} - x_{vi} \geq 0, \quad \forall (u, v) \in E, \forall i \in N,$$

$$(26) \quad f_{uvi} - x_{ui} - x_{vi} \leq 0, \quad \forall (u, v) \in E, \forall i \in N,$$

$$(27) \quad y_i - y_{i-1} \leq 0, \quad \forall i \in N,$$

$$(28) \quad x_{ui} \in \{0, 1\}, \quad \forall u \in V, \forall i \in N,$$

$$(29) \quad y_i \in \{0, 1\}, \quad \forall i \in N,$$

$$(30) \quad f_{uvi} \in \{0, 1\}, \quad \forall (u, v) \in E, \forall i \in N.$$

Let us briefly comment on model (*CM*). The objective is the same as that for the (*GM*) model, as discussed earlier. Constraints (20) refer to the capacity limitation of local rings. Constraints (24)–(26) ensure the correct relationship between f and x variables. Finally, constraints (18) are added to the model since the reduction of symmetry is beneficial for computational purposes. This formulation has $n^2 + 3mn + 3n + 1$ constraints and $n^2 + mn + n$ binary variables.

To conclude this section, we summarize in Table 1 the total number of variables and constraints in the models presented so far. Clearly, both models have a number of variables that is polynomial in the size of the input graph.

TABLE 1. Model sizes.

Model	# variables	# constraints
(GM)	$n^2 + 3mn + n$	$n^2 + 3mn + 2n + 1$
(CM)	$n^2 + mn + n$	$n^2 + 3mn + 3n + 1$

5. COMPUTATIONAL RESULTS

In this section, we report computational results using the formulations (GM) and (CM) discussed in Sections 3 and 4. Our experiments were performed on a 450MHz AMD K6 PC with 256 MB of RAM under Windows XP.

In order to compare the practical performance of the novel and the original formulation, we use instances proposed by Goldschmidt et al. [3] for the SRAP. To solve the models we used XPress 12.05 [2]. As stopping criteria for the Xpress, we set to 300,000 and 3,600 the number of nodes in the branch-and-bound tree and the CPU time (in seconds), respectively. For both formulations, we did *not* activate the separation of the standard cuts neither the pre-processing techniques of XPress.

5.1. Instances. The experiments use two groups of instances, namely, *geometric* and *randomly generated*. Within each group, instances are divided into those having *low* and *high* ring capacities, respectively, 155Mbs and 622Mbs.

In the following tables, the characteristics of an instance can be inferred from its name. Instance names always start with two letters. The first letter is either “g” or “r” meaning that the instance belongs to the geometric or the random subdivision, respectively. The second letter is either “l” or “h”, depending on the ring capacity is $B = 155\text{Mbs}$ or $B = 622\text{Mbs}$, respectively. For more detail on how those instances were generated as well as their properties, we refer the interested reader to [3] where they were introduced.

We test instances with $n = |V| = 15$, $m = |E| < n(n - 1)/2$ and the values k are taken from $\{2, 3, \dots, 7\}$, i.e., $k \leq \lfloor n/2 \rfloor$. In total, there are 119 instances.

5.2. Analysis of the experiments. Tables 2 through 6 summarize the computational results. The columns in these tables have the following meaning: *name* is the instance name, z^* indicates the optimal integer solution value, z gives the best lower bound found, *#Nodes* is the number of nodes explored in the branch-and-bound tree, $\Delta\%$ is the percentage gap between the optimal integer solution value (z^*) and the best lower bound found (z):

$$(31) \quad \Delta\% = \frac{z^* - z}{z^*} \times 100.$$

Finally, a column for CPU times is presented. The column indicates the time in seconds necessary to solve the model or to halt the execution after one hour of computation.

From Tables 2 through 6 we conclude that our formulation (CM) was able to prove the optimality for all 119 tested instances, while formulation (GM) proved optimality of only 38 instances.

The poor performance of the branch-and-bound algorithm when using the formulation (GM) is the result of the symmetry in the feasible solutions for the k SRAP. This symmetry is avoided in the feasible solutions for the formulation (CM). This is due to the constraints (18) that satisfactorially lead to the excellent performance of the branch-and-bound algorithm.

We also note that the CPU times of the formulation (CM) is substantially smaller than those for the formulation (GM). One can see from the results that (CM) solved all instances in less than an hour, whereas the (GM) model solved just 47 out of 119 instances tested. Among the 47 instances solved by both models, the model (CM) solved 39 instances in less than a minute, whereas the (GM) model did not solve any of them that fast.

These results show that our model is superior to (GM). Moreover, it is worth mentioning that our preliminary results showed that the superiority of (CM) over (GM) becomes even more evident when no tight upper bound on the optimal value is provided *a priori* to the models.

TABLE 2. Results for $k = 3$.

Name	z^*	Formulation (GM)				Formulation (CM)			
		z	#Nodes	$\Delta\%$	CPU time	z	#Nodes	$\Delta\%$	CPU time
gl-15-1	9750	9750	50441	0	669	9750	224	0	1
gl-15-4	10350	10350	43774	0	628	10350	278	0	2
gl-15-7	12750	12750	96038	0	1989	12750	209	0	1
gl-15-9	7650	7650	25740	0	315	7650	317	0	1
gh-15-1	33900	33900	22671	0	503	33900	262	0	2
gh-15-2	39600	30087.93	64400	24.02	3600	39600	1345	0	40
gh-15-8	58650	36750	37100	37.34	3600	58650	2347	0	70
gh-15-9	40500	37767.43	109000	6.75	3600	40500	576	0	4
rl-15-1	11250	11250	146353	0	1245	11250	981	0	4
rl-15-4	13200	13200	143306	0	2550	13200	391	0	2
rl-15-6	9300	9300	73602	0	1114	9300	617	0	3
rl-15-8	11250	9962.50	209900	11.44	3600	11250	722	0	4
rl-15-9	9600	9600	98441	0	1040	9600	790	0	3
rh-15-1	46800	38486.75	97000	17.76	3600	46800	1210	0	10
rh-15-5	42000	30450	83200	27.5	3600	42000	2072	0	13
rh-15-6	42750	30748.13	86400	28.07	3600	42750	4051	0	23
rh-15-9	39900	39900	94686	0	2267	39900	1367	0	8

TABLE 3. Results for $k = 4$.

Name	z^*	Formulation (GM)				Formulation (CM)			
		z	#Nodes	$\Delta\%$	CPU time	z	#Nodes	$\Delta\%$	CPU time
gl-15-1	9750	9750	59188	0	1062	9750	637	0	6
gl-15-4	10350	10350	45318	0	716	10350	642	0	6
gl-15-7	12750	12750	125214	0	2393	12750	1015	0	11
gl-15-9	7650	7650	27870	0	407	7650	684	0	4
gh-15-1	33900	33900	22059	0	429	33900	819	0	6
gh-15-2	39600	28650	61300	27.65	3600	39600	3855	0	99
gh-15-8	58650	33150	38500	43.48	3600	58650	9801	0	248
gh-15-9	40500	35100	75200	13.33	3600	40500	1586	0	13
rl-15-1	11250	11250	173268	0	1630	11250	2553	0	10
rl-15-4	13200	11550	110800	12.5	3600	13200	992	0	7
rl-15-6	9300	9300	73000	0	1239	9300	1478	0	8
rl-15-8	11250	9450	160900	16	3600	11250	3040	0	16
rl-15-9	9600	9600	96290	0	1053	9600	1971	0	8
rh-15-1	46800	33546.20	70100	28.32	3600	46800	4574	0	38
rh-15-5	42000	30366.84	66200	27.69	3600	42000	6223	0	47
rh-15-6	42750	30577.02	70200	28.47	3600	42750	11758	0	74
rh-15-9	39900	39900	94179	0	2386	39900	4260	0	27

TABLE 4. Results for $k = 5$.

Name	z^*	Formulation (<i>GM</i>)				Formulation (<i>CM</i>)			
		z	#Nodes	$\Delta\%$	CPU time	z	#Nodes	$\Delta\%$	CPU time
gl-15-1	9750	9750	46798	0	666	9750	1337	0	15
gl-15-4	10350	10350	46316	0	704	10350	1645	0	15
gl-15-7	12750	12750	120042	0	2344	12750	2459	0	28
gl-15-9	7650	7650	31257	0	407	7650	1324	0	8
gh-15-1	33900	33900	26327	0	580	33900	1434	0	11
gh-15-2	39600	28852.62	61600	27.14	3600	39600	8867	0	189
gh-15-8	58650	37200	40300	36.57	3600	58650	25509	0	541
gh-15-9	40500	33788.67	71200	16.57	3600	40500	3858	0	32
rl-15-1	11250	11250	152289	0	1559	11250	7105	0	31
rl-15-4	13200	11446.51	97000	13.28	3600	13200	2304	0	17
rl-15-6	9300	9300	70420	0	1412	9300	3711	0	20
rl-15-8	11250	9450	129800	16	3600	11250	6301	0	34
rl-15-9	9600	9600	96918	0	1273	9600	4707	0	23
rh-15-1	46800	35304.98	71600	24.56	3600	46800	9581	0	88
rh-15-5	42000	29343.51	62200	30.13	3600	42000	13970	0	110
rh-15-6	42750	30455.62	69800	28.76	3600	42750	25772	0	175
rh-15-9	39900	39900	95418	0	2317	39900	8108	0	55

TABLE 5. Results for $k = 6$.

Name	z^*	Formulation (<i>GM</i>)				Formulation (<i>CM</i>)			
		z	#Nodes	$\Delta\%$	CPU time	z	#Nodes	$\Delta\%$	CPU time
gl-15-1	9750	9750	45345	0	638	9750	2945	0	30
gl-15-4	10350	10350	44037	0	691	10350	3400	0	26
gl-15-7	12750	12750	127133	0	2669	12750	5689	0	74
gl-15-9	7650	7650	27733	0	355	7650	2384	0	16
gh-15-1	33900	33900	24363	0	565	33900	2068	0	17
gh-15-2	39600	29250	66700	26.14	3600	39600	18818	0	351
gh-15-8	58650	37200	44000	36.57	3600	58650	55354	0	1163
gh-15-9	40500	34143.65	71500	15.69	3600	40500	8662	0	70
rl-15-1	11250	11250	178344	0	1841	11250	12313	0	56
rl-15-4	13200	11316.67	94400	14.27	3600	13200	4365	0	34
rl-15-6	9300	9300	71467	0	1362	9300	6689	0	38
rl-15-8	11250	9284.85	133000	17.47	3600	11250	13368	0	75
rl-15-9	9600	9600	99208	0	1291	9600	9087	0	47
rh-15-1	46800	35321.13	70100	24.53	3600	46800	17944	0	186
rh-15-5	42000	29680.33	66100	29.33	3600	42000	25422	0	209
rh-15-6	42750	30138.18	71000	29.5	3600	42750	52924	0	385
rh-15-9	39900	39900	90517	0	2267	39900	14198	0	98

6. CONCLUDING REMARKS

In this paper we have presented a novel and compact 0-1 integer linear programming formulation for the k SONET Ring Assignment Problem. Computational experiments comparing our formulation and one proposed by Goldschmidt et al. [3] are reported. These results showed a clear superiority of our formulation and it turns out that small instances are solved significantly faster with the novel model than with the original formulation from [3].

Further computational experiments on more and larger benchmark instances, for example 25, 30 and 50 nodes and $B=2.5$ Gbs, are necessary to confirm the efficiency of the model

TABLE 6. Results for $k = 7$.

Name	z^*	Formulation (<i>GM</i>)				Formulation (<i>CM</i>)			
		z	#Nodes	$\Delta\%$	CPU time	z	#Nodes	$\Delta\%$	CPU time
gl-15-1	9750	9750	46207	0	641	9750	6385	0	42
gl-15-4	10350	10350	44994	0	672	10350	7337	0	45
gl-15-7	12750	12750	136853	0	2714	12750	9717	0	128
gl-15-9	7650	7650	31295	0	417	7650	3166	0	26
gh-15-1	33900	33900	24200	0	541	33900	3121	0	27
gh-15-2	39600	28763.62	64700	27.36	3600	39600	33332	0	592
gh-15-8	58650	36018.77	40000	38.59	3600	58650	105025	0	2040
gh-15-9	40500	35588.31	75200	12.13	3600	40500	15480	0	127
rl-15-1	11250	11250	177438	0	1759	11250	24237	0	114
rl-15-4	13200	11940	108900	9.55	3600	13200	7902	0	66
rl-15-6	9300	9300	61612	0	1265	9300	12204	0	68
rl-15-8	11250	9103.77	129800	19.08	3600	11250	23836	0	140
rl-15-9	9600	9600	104853	0	1336	9600	16419	0	87
rh-15-1	46800	36150	71900	22.76	3600	46800	31401	0	367
rh-15-5	42000	29250	64400	30.36	3600	42000	43499	0	411
rh-15-6	42750	30600	69000	28.42	3600	42750	91308	0	706
rh-15-9	39900	39900	89300	0	2466	39900	22331	0	175

(*CM*). Besides, we also want to combine branch-and-price algorithm with (meta)-heuristic approaches for the resolution of the k -SRAP. This will be done in the classical sense, that is, by heuristically determining a good starting solution for an exact algorithm [6, 7, 8].

REFERENCES

- [1] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Salvendy, and P. H. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.
- [2] Dash Optimization Inc. *Xpress-Optimizer Reference Manual*, 2003.
- [3] O. Goldschmidt, A. Laugier, and E. V. Olinick. SONET/SDH ring assignment with capacity constraints. *Discrete Applied Mathematics*, 129:99–128, 2003.
- [4] S. Holm and M. M. Sorensen. The optimal graph partitioning problem: solution method based on reducing symmetric nature and combinatorial cuts. *O.R. Spektrum*, 15:1–8, 1993.
- [5] E. M. Macambira, N. Maculan, and C. C. de Souza. Reducing symmetry of the SONET ring assignment problem using hierarchical inequalities. Technical Report ES-636/04, Programa de Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, 2004.
- [6] E.M. Macambira and C.C. de Souza. The edge-weighted clique problem: valid inequalities, facets and polyhedral computations. *European Journal of Operational Research*, 123(2):346–371, 2000.
- [7] E.M. Macambira, N. Maculan, and C.C. de Souza. Integer programming models for the SONET ring assignment problem. Technical report, Instituto de Computação, Universidade Estadual de Campinas, June 2005.
- [8] C. N. Meneses and C. C. de Souza. Exact solutions of rectangular partitions via integer programming. *International Journal of Computational Geometry and Applications*, 10(5):477–522, 2000.
- [9] H. D. Sherali and J. C. Smith. Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407, 2001.
- [10] P. Soriano, C. Wynants, R. Séguin, M. Labbé, M. Gendreau, and B. Fortz. Design and dimensioning of survivable SDH/SONET networks. In B. Sansò e P. Soriano (eds.), editor, *Telecommunications Network Planning*, pages 147–167. Kluwer Academics Publishers, 1999.

(E.M. Macambira) DEPARTMENT OF STATISTICS, FEDERAL UNIVERSITY OF PARAÍBA, CIDADE UNIVERSITÁRIA, 21941-972 JOÃO PESSOA, PB, BRAZIL

E-mail address, E.M. Macambira: elder@de.ufpb.br

(C.N. Meneses) DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, 303 WEIL HALL, GAINESVILLE, FL 32611

E-mail address, C.N. Meneses: claudio@ufl.edu

(P.M. Pardalos) DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, 303 WEIL HALL, GAINESVILLE, FL 32611

E-mail address, P.M. Pardalos: pardalos@ufl.edu

(M.G.C. Resende) INTERNET AND NETWORK SYSTEMS RESEARCH CENTER, AT&T LABS RESEARCH, 180 PARK AVENUE, ROOM C241, FLORHAM PARK, NJ 07932 USA.

E-mail address, M.G.C. Resende: mgcr@research.att.com