On a Traveling Salesman based Bilevel Programming Problem

Pablo Adasme[1], Rafael Andrade[2], Janny Leung[3] and Abdel Lisser[4]
[1]*Departamento de Ingeniería Eléctrica, Universidad de Santiago de Chile, Avenida Ecuador 3519, Santiago, Chile*
[2]*Departamento de Estatística e Matemática Aplicada, Universidade Federal do Ceará, Campus do Pici, BL 910, CEP 60.455-760, Fortaleza, Ceará, Brazil*
[3]*Department of Systems Engineering & Engineering Management, Chinese University of Hong Kong, Shatin, Hong Kong, China*
[4]*Laboratoire de Recherche en Informatique, Université de Paris-Sud 11, Bât. 650 Ada Lovelace, Paris, France*

Keywords: Bilevel Programming, Traveling Salesman Problem, Mixed Integer Linear Programming Formulations, Iterative Sub-tour Elimination Constraint Procedure.

Abstract: In this paper, we consider a linear bilevel programming problem where both the leader and the follower maximize their profits subject to budget constraints. Additionally, we impose a Hamiltonian cycle topology constraint in the leader problem. In particular, models of this type can be motivated by telecommunication companies when dealing with traffic network flows from one server to another one within a ring topology framework. We transform the bilevel programming problem into an equivalent single level optimization problem that we further linearize in order to derive mixed integer linear programming (MILP) formulations. This is achieved by replacing the follower problem with the equivalent Karush Kuhn Tucker conditions and with a linearization approach to deal with the complementarity constraints. The topology constraint is handled by the means of two compact formulations and an exponential one from the classic traveling salesman problem. Thus, we compute optimal solutions and upper bounds with linear programs. One of the compact models allows to solve instances with up to 250 nodes to optimality. Finally, we propose an iterative procedure that allows to compute optimal solutions in remarkably less computational effort when compared to the compact models.

1 INTRODUCTION

Bilevel programming is a two level hierarchical optimization framework where the upper level problem is referred to as the leader whilst the lower level problem is referred to as the follower problem. In a bilevel programming problem (BPP), we aim to find an optimal point such that the leader and the follower maximize (or minimize) their respective objective functions subject to linking constraints. Applications concerning BPPs arise in agriculture, economic systems, finance, engineering, banking, transportation, network design, management and planning to name a few. For a more general description of BPP applications and algorithmic approaches to solve these problems see for instance (Dempe, 2003; Floudas and Pardalos, 2001; Migdalas et al., 1997; Thirwani and Arora, 1998; Vicente et al., 1994; Wang et al., 1994).

In this paper, we consider a linear bilevel programming problem (LBPP) where both the leader and the follower maximize their profits subject to budget constraints. Additionally, we impose a Hamiltonian cycle topology constraint in the leader problem. In particular, models of this type can be motivated by telecommunication companies when dealing with traffic network flows from one server to another one within a ring topology framework. As an example, consider the problem of flow traffic management in a backbone wireless token ring network where users connect to any node and pass their messages trough the ring in order to reach another user which is also connected to a node in the ring (Lee et al., 2001; Song and Yang, 1997). In these types of networks, the traffic flows can be significantly large which might require more than one network operator to deal with the flow problem.

We transform the LBPP into an equivalent single level optimization problem that we further linearize in order to derive mixed integer linear programming (MILP) formulations. This is achieved by replacing the follower problem with the equivalent Karush Kuhn Tucker (KKT) conditions and with

Adasme P., Andrade R., Leung J. and Lisser A.
On a Traveling Salesman based Bilevel Programming Problem.
DOI: 10.5220/0006190503290336
In *Proceedings of the 6th International Conference on Operations Research and Enterprise Systems (ICORES 2017)*, pages 329-336
ISBN: 978-989-758-218-9

the linearization approach proposed in (Audet et al., 1997) to deal with the complementarity slackness conditions. The topology constraint is handled by the means of two compact polynomial formulations and an exponential one from the classic traveling salesman problem (TSP) (Gavish and Graves, 1978; Letchford et al., 2013; Miller et al., 1960). Thus, we compute optimal solutions and upper bounds with the MILP and linear programming (LP) relaxations, respectively.

Our contribution in this paper is not theoretical, but mainly focussed on computational numerical results on a novel problem in the domain of bilevel programing. As far as we know, LBPPs including Hamiltonian cycle topology constraints have not been considered in the literature so far. We compare numerically the exponential model with the two compact formulations for randomly generated instances. For this purpose, first we solve the exponential model by generating all cycle elimination constraints and then, by using an iterative algorithmic procedure which consists of adding violated cycle elimination constraints within each iteration until no cycle is found in the current solution. Finally, we further propose a relaxed version of the iterative algorithm and compute tight upper bounds as well. Our main numerical result is to show that solving the exponential model with the iterative procedure is by far more convenient than using the compact formulations which are more theoretical based approaches. In fact, we solve to optimality instances with up to 250 nodes so far, in less than 70 seconds in average compared to the higher CPU times required by the compact formulations. It has been proved that BPPs are strongly NP-hard even for the simplest case in which all the involved functions are affine. See for instance (Migdalas et al., 1997; Scholtes, 2004). The reader is referred to the books (Dempe, 2002; Migdalas et al., 1997) for a more general understanding on bilevel programming.

The remaining of the paper is organized as follows. In section 2, first we present and explain the LBPP problem. Then, we discuss three equivalent MILP models while using the exponential and the two compact formulations. Subsequently, in section 3, we present the alternative iterative procedures which allow to obtain optimal solutions and tight upper bounds using the exponential model. Afterwards, in section 4 we compare all the proposed models and the iterative procedures with the optimal solution of the problem. Finally, in section 5 we give the main conclusions of the paper and provide some insight for future research.

2 LINEAR BILEVEL AND MILP MODELS

In this section, we present and explain the LBPP. In particular, we use an exponential number of sub-tour elimination constraints (SECs) to characterize the Hamiltonian cycle condition in the leader problem. Subsequently, we use two additional compact modeling approaches from the traveling salesman problem (Gavish and Graves, 1978; Letchford et al., 2013; Miller et al., 1960) and obtain equivalent MILP models. Let $G(V,E)$ denote a complete graph with set of nodes V and set of directed arcs E. Our LBPP can be formulated as follows

$$\mathrm{BP}_1 : \max_{\{f,g,x\}} \left\{ \sum_{ij\in E} C_{ij} f_{ij} + \sum_{ij\in E} D_{ij} g_{ij} \right\} \quad (1)$$

$$\text{s.t.} \sum_{j:ij\in E} A_{ij} f_{ij} + \sum_{j:ij\in E} B_{ij} g_{ij} \leq c, \forall i \in V \quad (2)$$

$$f_{ij} + g_{ij} \leq M x_{ij}, \forall ij \in E \quad (3)$$

$$\sum_{j:ij\in E} x_{ij} = 1, \forall i \in V \quad (4)$$

$$\sum_{j:ji\in E} x_{ji} = 1, \forall i \in V \quad (5)$$

$$\sum_{ij\in E(S)} x_{ij} \leq |S| - 1, \forall S \subset V \quad (6)$$

$$x_{ij} \in \{0,1\}, f_{ij} \geq 0, \forall ij \in E \quad (7)$$

$$g \in \arg\max_{\{g\}} \left\{ \sum_{ij\in E} H_{ij} g_{ij} + \sum_{ij\in E} P_{ij} f_{ij} \right\} \quad (8)$$

$$\sum_{j:ij\in E} Q_{ij} f_{ij} + \sum_{j:ij\in E} R_{ij} g_{ij} \leq r, \quad \forall i \in V \quad (9)$$

$$g_{ij} \geq 0, \forall ij \in E \quad (10)$$

In BP_1, the input symmetric matrices $\{C,D,A,B,H,P,Q,R\} \in \mathbb{M}_{|V|\times|V|}(\mathbb{R}_+)$ and the scalars $\{c,r\} \in \mathbb{R}_+$, respectively. Constraints (1)-(7) correspond to the leader problem whereas constraints (8)-(10) represent the follower problem. Without loss of generality, we assume that the set V represents servers to visit whereas the set E represents traffic links by which the network flow service should be carried out from one server to another. Thus, the binary decision variable $x_{ij} = 1$ if and only if the leader company decides to use the link (i,j) and $x_{ij} = 0$ otherwise, $\forall ij \in E$. Similarly, the decision variables $f_{ij}, g_{ij} \geq 0$ represent the amount of network flow to transport from i to j for the leader and follower problems, respectively. The objective functions maximize the profits for both the leader and the follower and are given in (1) and (8), respectively. Whereas the constraints (2) and (9) represent the costs structure

associated to each one of them. Constraints (7) and (10) are domain constraints for the decision variables. The constraints (3)-(6) characterize the Hamiltonian cycle condition imposed in the leader problem. In particular, the constraint (3) implies that whenever the variable $x_{ij} = 0$, then the variables f_{ij} and g_{ij} must be equal to zero $\forall ij \in E$. For this purpose, we use a large positive BigM value denoted by M. Constraints (4)-(5) enforce the condition that each node should be connected to two nodes in the circuit. Finally, constraint (6) represents SECs $\forall S \subset V$. Notice that we do not include SECs for $S \equiv V$ in order to allow obtaining feasible solutions, i.e., Hamiltonian cycles.

An equivalent MILP model can be straightforwardly obtained by replacing the follower problem with the equivalent KKT conditions and by using the linearization approach proposed in (Audet et al., 1997) to deal with the complementarity slackness conditions (Audet et al., 1997; Dempe, 2003). This leads to the following equivalent MILP model

$$\text{MIP}_1 : \max_{\{f,g,x,\lambda,\mu,\theta,\nu\}} \left\{ \sum_{ij\in E} C_{ij} f_{ij} + \sum_{ij\in E} D_{ij} g_{ij} \right\}$$

$$\text{s.t.} \sum_{j:ij\in E} A_{ij} f_{ij} + \sum_{j:ij\in E} B_{ij} g_{ij} \leq c, \forall i \in V$$

$$f_{ij} + g_{ij} \leq M x_{ij}, \forall ij \in E$$

$$\sum_{j:ij\in E} x_{ij} = 1, \forall i \in V$$

$$\sum_{j:ji\in E} x_{ji} = 1, \forall i \in V$$

$$\sum_{ij\in E(S)} x_{ij} \leq |S| - 1, \forall S \subset V \quad (11)$$

$$x_{ij} \in \{0,1\}, f_{ij} \geq 0, \forall ij \in E$$

$$H_{ij} - \lambda_i R_{ij} + \mu_{ij} = 0, \forall ij \in E \quad (12)$$

$$\sum_{j:ij\in E} Q_{ij} f_{ij} + \sum_{j:ij\in E} R_{ij} g_{ij} \leq r, \forall i \in V \quad (13)$$

$$r - \sum_{j:ij\in E} Q_{ij} f_{ij} - \sum_{j:ij\in E} R_{ij} g_{ij} + \nu_i L \leq L,$$

$$\forall i \in V \quad (14)$$

$$\lambda_i \leq \nu_i L, \forall i \in V \quad (15)$$

$$\mu_{ij} + \theta_{ij} L \leq L, \forall ij \in E \quad (16)$$

$$g_{ij} \leq \theta_{ij} L, \forall ij \in E \quad (17)$$

$$g_{ij}, \mu_{ij} \geq 0 \forall ij \in E, \lambda_i \geq 0 \forall i \in V \quad (18)$$

$$\nu_i \in \{0,1\} \forall i \in V, \theta_{ij} \in \{0,1\} \forall ij \in E \quad (19)$$

where the constraints (12) are due to the derivatives obtained with the Lagrangian function of the follower problem and with respect to the variables $g_{ij}, \forall ij \in E$. The non-negative variables λ_i, $\forall i \in V$ and μ_{ij}, $\forall ij \in E$ are dual variables for the constraints (9) and (10), respectively. The constraints (13)-(15) enforce the condition that either $\left(r - \sum_{j:ij\in E} Q_{ij} f_{ij} - \sum_{j:ij\in E} R_{ij} g_{ij}\right)$ or λ_i should be equal to zero $\forall i \in V$. This is handled with the binary variable ν_i $\forall i \in V$ and with the large positive value L. Similarly, the constraints (16)-(17) enforce the condition that either μ_{ij} or g_{ij} should be equal to zero for all $ij \in E$. This is handled with the variables $\theta_{ij} \in \{0,1\} \forall ij \in E$. Finally, (18)-(19) are domain constraints for the decision variables.

As it can be observed, the number of SECs in MIP_1 is exponential. To overcome this difficulty, we further consider the following MILP model which uses a well known characterization of the feasible space of the traveling salesman problem (Letchford et al., 2013; Miller et al., 1960)

$$\text{MIP}_2 : \max_{\{f,g,x,u,\lambda,\mu,\theta,\nu\}} \left\{ \sum_{ij\in E} C_{ij} f_{ij} + \sum_{ij\in E} D_{ij} g_{ij} \right\}$$

$$\text{s.t.} \sum_{j:ij\in E} A_{ij} f_{ij} + \sum_{j:ij\in E} B_{ij} g_{ij} \leq c, \forall i \in V$$

$$f_{ij} + g_{ij} \leq M x_{ij}, \forall ij \in E$$

$$\sum_{j:ij\in E} x_{ij} = 1, \forall i \in V$$

$$\sum_{j:ji\in E} x_{ji} = 1, \forall i \in V$$

$$u_1 = 1 \quad (20)$$

$$2 \leq u_i \leq |V|, \forall i \in V, (i \neq 1) \quad (21)$$

$$u_i - u_j + 1 \leq (|V| - 1)(1 - x_{ij}),$$

$$\forall ij \in E, (i \neq 1), (j \neq 1) \quad (22)$$

$$u_j \in \mathbb{Z}_+, \forall j \in V \quad (23)$$

$$x_{ij} \in \{0,1\}, f_{ij} \geq 0, \forall ij \in E$$

$$H_{ij} - \lambda_i R_{ij} + \mu_{ij} = 0, \forall ij \in E$$

$$\sum_{j:ij\in E} Q_{ij} f_{ij} + \sum_{j:ij\in E} R_{ij} g_{ij} \leq r, \forall i \in V$$

$$r - \sum_{j:ij\in E} Q_{ij} f_{ij} - \sum_{j:ij\in E} R_{ij} g_{ij} + \nu_i L \leq L, \forall i \in V$$

$$\lambda_i \leq \nu_i L, \forall i \in V$$

$$\mu_{ij} + \theta_{ij} L \leq L, \forall ij \in E$$

$$g_{ij} \leq \theta_{ij} L, \forall ij \in E$$

$$g_{ij}, \mu_{ij} \geq 0 \forall ij \in E, \lambda_i \geq 0 \forall i \in V$$

$$\nu_i \in \{0,1\} \forall i \in V, \theta_{ij} \in \{0,1\} \forall ij \in E$$

where the constraints (22) ensure that, if the salesman travels from i to j, then the nodes i and j are arranged sequentially. These constraints together with (20)-(21) and (23) ensure that each node is in a unique position. A third formulation can be obtained by using the classic single commodity flow formulation for the TSP (Gavish and Graves, 1978; Letchford et al., 2013). For this purpose, we assume that the salesman carries $|V| - 1$ units of a commodity when he leaves node 1, and delivers 1 unit of this commodity to each node. We can define additional continuous variables

$w_{i,j} \geq 0, \forall ij \in E$ representing the amount of the commodity (if any) routed directly from node i to node j. The new MILP formulation is

$$\text{MIP}_3 : \max_{\{f,g,x,w,\lambda,\mu,\theta,\nu\}} \left\{ \sum_{ij\in E} C_{ij} f_{ij} + \sum_{ij\in E} D_{ij} g_{ij} \right\}$$

$$\text{s.t.} \sum_{j:ij\in E} A_{ij} f_{ij} + \sum_{j:ij\in E} B_{ij} g_{ij} \leq c, \forall i \in V$$

$$f_{ij} + g_{ij} \leq M x_{ij}, \forall ij \in E$$

$$\sum_{j:ij\in E} x_{ij} = 1, \forall i \in V$$

$$\sum_{j:ji\in E} x_{ji} = 1, \forall i \in V$$

$$\sum_{j:ji\in E} w_{ji} - \sum_{j>1:ij\in E} w_{ij} = 1,$$

$$\forall i \in \{2,\ldots,|V|\} \quad (24)$$

$$0 \leq w_{ij} \leq (|V|-1) x_{ij}, \forall ij \in E \quad (25)$$

$$x_{ij}, f_{ij} \geq 0 \forall ij \in E$$

$$H_{ij} - \lambda_i R_{ij} + \mu_{ij} = 0, \forall ij \in E$$

$$\sum_{j:ij\in E} Q_{ij} f_{ij} + \sum_{j:ij\in E} R_{ij} g_{ij} \leq r, \forall i \in V$$

$$r - \sum_{j:ij\in E} Q_{ij} f_{ij} - \sum_{j:ij\in E} R_{ij} g_{ij} + \nu_i L \leq L, \forall i \in V$$

$$\lambda_i \leq \nu_i L, \forall i \in V$$

$$\mu_{ij} + \theta_{ij} L \leq L, \forall ij \in E$$

$$g_{ij} \leq \theta_{ij} L, \forall ij \in E$$

$$g_{ij}, \mu_{ij} \geq 0 \forall ij \in E, \lambda_i \geq 0 \forall i \in V$$

$$\nu_i \in \{0,1\} \forall i \in V, \theta_{ij} \in \{0,1\} \forall ij \in E$$

The constraints (24) ensure that one unit of the commodity is delivered to each node while the bounds in (25) ensure that the commodity can flow only along arcs in the solution. Hereafter, we denote by LP_1, LP_2 and LP_3 the LP relaxations of MIP_1, MIP_2 and MIP_3, respectively.

In the next section, we present an alternative iterative algorithmic procedure that allows to obtain optimal solutions and tight upper bounds for MIP_1.

3 ITERATIVE PROCEDURE FOR GENERATING SECS

The procedure to generate SECs can be easily adapted using Algorithms 4.1 and 4.2 from (Adasme et al., 2015) to MIP_1. The main idea can be described as follows. If we remove constraints (11) from MIP_1 and solve the resulting integer linear programming problem, then the underlying optimal solution induces a graph $\tilde{G}$ that may contain a cycle with at least two nodes. In this case, it can be detected by a depth-first search procedure (Cormen et al., 2009). In this paper, we adapt the procedure 4.1 from (Adasme et al., 2015) to find cycles in directed graphs with at least 2 and up to $|V|-1$ nodes. In particular, if the cardinality of a subset of nodes found with Algorithm 4.1 inducing a cycle equals $|V|$, we do not generate the SEC, otherwise Hamiltonian cycles would be infeasible for the problem.

Algorithm 3.1: Iterative procedure to compute upper bounds for MIP_1.

Data: A problem instance of MIP_1.
Result: An upper bound with solution $(f, g, x_R, \lambda, \mu, \theta, \nu)$ for MIP_1 with objective function value z_b.
Step 0: Set $k = 1$;
Let MIP_{1_k} be the problem obtained from MIP_1 by removing the constraints (11) at iteration k;
Solve the MILP relaxation of problem MIP_{1_k} and let $(f^k, g^k, x_R^k, \lambda^k, \mu^k, \theta^k, \nu^k)$ be its optimal solution of value z_k at iteration k;
Let $z_0 = \inf$;
Step 1: **while** $|z_{k-1} - z_k| > \varepsilon$ **do**
  Construct the graph $\tilde{G} = (V, \tilde{E})$ with the rounded solution $(\tilde{f}^k, \tilde{g}^k, \tilde{x}_R^k, \tilde{\lambda}^k, \tilde{\mu}^k, \tilde{\theta}^k, \tilde{\nu}^k)$ obtained from $(f^k, g^k, x_R^k, \lambda^k, \mu^k, \theta^k, \nu^k)$;
  $\mathcal{C} = searchCycles(\tilde{G}, V)$;
  foreach *cycle* $\in \mathcal{C}$ **do**
    Add the corresponding constraint (11) to MIP_{1_k};
  Set $k = k + 1$;
  Solve the MILP relaxation of problem MIP_{1_k} and let $(f^k, g^k, x_R^k, \lambda^k, \mu^k, \theta^k, \nu^k)$ be its optimal solution of value z_k at iteration k;
return *the solution* $(f^k, g^k, x_R^k, \lambda^k, \mu^k, \theta^k, \nu^k, z_k)$;

Algorithm 4.1 is used iteratively by Algorithm 4.2 in (Adasme et al., 2015) that we adapt to solve MIP_1. The procedures in Algorithms 4.1 and 4.2 can be straightforwardly explained in more detail as follows. First, we remove constraints (11) from MIP_1 and solve the resulting integer optimization problem. Consider the underlying optimal solution graph $\tilde{G} = (V, \tilde{E})$ where V is the set of nodes and $\tilde{E}$ is the set of arcs such that $\tilde{E} \subseteq E$. If $\tilde{G}$ contains a cycle with two or up to $|V|-1$ nodes, then Algorithm 4.1 detects it. A subset of nodes inducing a cycle defines a new constraint (11) which cuts off this cycle from the solution space. Problem MIP_1 is re-optimized taking into account the new added constraints. This iterative process goes on until the underlying current optimal solution of MIP_1 has no more cycles. Since the number of cycles is finite, so is the number of constraints (11) that can be added to MIP_1. Notice that the number of SECs of type (11) that can be added to

MIP_1 is at most $O(2^{|V|})$. Consequently, Algorithm 4.2 adapted to solve MIP_1, converges to the optimal solution of the problem in at most $O(2^{|V|})$ outer iterations. The proof can be directly deduced from Theorem 2 in (Adasme et al., 2015).

The aforementioned procedure can also be used to compute upper bounds for MIP_1. This procedure is depicted in Algorithm 3.1 and is described as follows. First, we remove constraints (11) from MIP_1 and solve the resulting mixed integer linear programming relaxation of MIP_1 obtained while relaxing the variables $0 \leq x_{ij} \leq 1 \ \forall ij \in E$ at step 0. Next, we search cycles in the current rounded solution $0 \leq x_{ij} \leq 1 \ \forall ij \in E$. If $\tilde{G}$ contains a cycle with two or more nodes, then Algorithm 4.1 referred to as "*searchCycles*$(\tilde{G}, V)$" in (Adasme et al., 2015) detects it. A subset of nodes inducing a cycle defines a new constraint (11). The mixed integer programming relaxation of MIP_1 is re-optimized taking into account the new added constraints. This iterative process goes on until the difference between the current optimal objective function value z_k and the previous one z_{k-1} is less than a small positive value ε.

4 PRELIMINARY NUMERICAL RESULTS

In this section, we present preliminary numerical results. A Matlab (R2012a) program is developed using CPLEX 12.6 to solve MIP_1, MIP_2, MIP_3 and their corresponding LP relaxations. The numerical experiments have been carried out on an Intel(R) 64 bits core (TM) with 2.6 GHz and 8 Gigabytes of RAM. CPLEX solver is used with default options. Each entry in matrices $\{C, D, H\}$ and in matrices $\{A, B, R, Q\}$ is randomly and uniformly distributed in the intervals $[0; 10]$ and $[0; 5]$, respectively. The scalar values $c = r = 100$. The BigM values M and L are set to $M = 100$ and $L = 10^{10}$, respectively. We set the parameter $\varepsilon = 10^{-8}$ in Algorithm 3.1. In Table 1, first we solve MIP_1 with up to 15 nodes while generating all cycle elimination constraints. Subsequently, in Table 3, we solve MIP_1 with the iterative procedures presented in section 3. We limit CPLEX to 2 hours of CPU time in order to solve the linear models. The legend in Table 1 is as follows. Column 1 shows the instance number. Column 2 presents the number of nodes. Columns 3-7 and 8-12 present the optimal solution of MIP_1 and MIP_2, the number of branch and bound nodes used by CPLEX, the CPU time in seconds to solve the MILPs and their corresponding LP relaxations together with their CPU time in seconds, respectively. Finally, in columns 13-14, we present gaps we compute as $\left[\frac{LP-Opt}{Opt}\right] * 100$ for MIP_1 and MIP_2, respectively. The legend in Table 2 for MIP_3 is analogous to Table 1. We mention that each row in Tables 1, 2 and 3 corresponds to the same instance.

From Tables 1, 2 and 3, we observe that the optimal objective function values for MIP_1, MIP_2 and MIP_3 are exactly the same. In particular, in Tables 1 and 2, we see that the CPU times are in average significantly lower for MIP_3 than for MIP_2. In particular, when the instance dimensions increase. Regarding the number of branch and bound nodes, we observe that CPLEX requires significantly less nodes for solving MIP_3 than for MIP_2. For MIP_1, the number of nodes equals zero for the instances 1-7. Notice that CPLEX cannot find a feasible solution within 2 hours for the instance #18 using MIP_2. This is somehow reflected by the number of branch and bound nodes which is significantly higher for MIP_2 than for MIP_3. Concerning the LP bounds, the LP relaxations are slightly tighter for MIP_1 than for MIP_2 and MIP_3. Whereas the bounds for LP_2 and LP_3 remain nearly the same. On the opposite, the CPU times for LP_3 are in average larger than for LP_2, in particular for the large size instances. We also see that CPLEX can solve all the instances to optimality using MIP_3 that shows a significantly better performance than the rest of the MILP formulations. Finally, we mention that we cannot solve, with the exponential model, instances with more than 15 nodes due to the large number of sub-tour elimination constraints involved.

In Table 3, the legend is as follows. In column 1, we show the instance number. In columns 2-5, we show the optimal solution obtained with the adapted version of Algorithm 4.2 (Adasme et al., 2015), its CPU time in seconds, the number of cycles found with this algorithm and the number of iterations, respectively. In columns 6-10, we present the upper bounds obtained with Algorithm 3.1, its CPU time in seconds, the number of cycles found with it, the number of iterations, and the optimal solution found with MIP_1 while using all the cycle elimination constraints found with Algorithm 3.1, respectively. For the latter, we do not report the CPU time required by CPLEX. However, we mention that for the largest size instances (e.g. 21-22), these CPU times took less than 20 seconds. The rest of the instances were solved in less than 2 seconds. Finally, in columns 11-12, we provide gaps that we compute by $\left[\frac{Opt_{It}^{R} - Opt_{It}}{Opt_{It}}\right] * 100$ and $\left[\frac{Opt_{It}^{F} - Opt_{It}}{Opt_{It}}\right] * 100$, respectively.

From Table 3, we observe that Algorithm 4.2 can find the optimal solutions for all the instances. In particular, we observe that the large size instances #15-22 are solved in significantly less CPU time compared to

Table 1: Numerical results obtained with MIP_1 and MIP_2.

#	$\|V\|$	MIP_1					MIP_2					Gaps	
		Opt	*B&Bn*	Time (s)	LP	Time (s)	Opt	*B&Bn*	Time (s)	LP	Time (s)	Gap_1 %	Gap_2 %
1	4	945.90	0	0.35	1547.52	0.35	945.90	0	0.36	1547.52	0.41	63.60	63.60
2	6	1223.95	0	0.41	1969.08	0.39	1223.95	0	0.41	1969.08	0.42	60.88	60.88
3	8	1574.80	0	0.37	2990.03	0.42	1574.80	0	0.37	2996.69	0.38	89.87	90.29
4	10	1992.63	0	0.53	5252.53	0.53	1992.63	29	0.34	5368.06	0.40	163.60	169.40
5	12	2781.27	0	1.83	5472.70	1.28	2781.27	6	0.40	5745.17	0.39	96.77	106.57
6	14	3795.17	0	7.13	6476.50	5.45	3795.17	120	0.42	6505.47	0.39	70.65	71.41
7	15	4233.11	0	14.74	6596.09	11.05	4233.11	0	0.39	6723.12	0.34	55.82	58.82
8	20	-	-	-	-	-	5920.61	1886	1.39	10090.09	0.39	-	70.42
9	25	-	-	-	-	-	7979.07	30	0.53	13672.93	0.38	-	71.36
10	30	-	-	-	-	-	11272.16	8	0.50	16961.67	0.41	-	50.47
11	40	-	-	-	-	-	14325.01	0	0.62	25545.21	0.52	-	78.33
12	50	-	-	-	-	-	17772.48	0	0.80	30645.26	0.55	-	72.43
13	60	-	-	-	-	-	23021.58	528	2.63	39462.28	0.86	-	71.41
14	70	-	-	-	-	-	28178.71	54	2.17	47919.83	1.19	-	70.06
15	80	-	-	-	-	-	32776.32	335	3.87	55799.72	1.55	-	70.24
16	90	-	-	-	-	-	35705.34	918	26.15	62552.35	1.90	-	75.19
17	100	-	-	-	-	-	42688.50	44278	452.83	73934.28	2.39	-	73.19
18	120	-	-	-	-	-	*	379412	7200	86563.41	5.50	-	*
19	150	-	-	-	-	-	-	-	-	113013.27	21.22	-	-
20	180	-	-	-	-	-	-	-	-	139047.61	59.52	-	-
21	200	-	-	-	-	-	-	-	-	156705.84	73.85	-	-
22	250	-	-	-	-	-	-	-	-	201738.80	268.30	-	-

-: Instance not solved.
*: No solution found with CPLEX in 2 hours.

Table 2: Numerical results obtained with MIP_3.

#	MIP_3					Gaps
	Opt	*B&Bn*	Time (s)	LP	Time (s)	Gap_3 %
1	945.90	0	0.40	1547.52	0.37	63.60
2	1223.95	0	0.40	1969.08	0.37	60.88
3	1574.80	0	0.37	2996.69	0.39	90.29
4	1992.63	0	0.42	5361.12	0.37	169.05
5	2781.27	9	0.49	5745.17	0.39	106.57
6	3795.17	8	0.49	6485.70	0.42	70.89
7	4233.11	0	0.40	6723.12	0.36	58.82
8	5920.61	10	0.94	10047.35	0.43	69.70
9	7979.07	0	0.54	13618.52	0.41	70.68
10	11272.16	0	0.75	16961.67	0.45	50.47
11	14325.01	0	0.79	25538.77	0.68	78.28
12	17772.48	0	1.17	30645.26	0.64	72.43
13	23021.58	675	10.46	39453.01	1.05	71.37
14	28178.72	8	5.11	47881.09	1.83	69.92
15	32776.31	248	17.79	55799.72	2.93	70.24
16	35705.36	17	15.86	62552.35	2.53	75.19
17	42688.51	621	46.65	73934.28	3.75	73.19
18	50786.93	526	133.94	86493.02	17.00	70.31
19	66770.38	163	123.39	113003.02	58.75	69.24
20	86307.85	4168	2459.89	139046.61	211.01	61.11
21	93444.25	5678	6878.94	156675.69	230.30	67.67
22	122719.37	549	2017.85	201734.01	901.54	64.39

MIP_3. Moreover, the number of cycles and iterations are less or equal than 140 and 11, respectively for all the instances. Notice that the total number of cycles for most of the instances is huge. However, the number of cycles required by Algorithm 4.2 to find the optimal solution is significantly small. In general, we see that Algorithm 3.1 can find tighter bounds when compared to the models LP_2 and LP_3. Regarding the number of cycles required by Algorithm 3.1, we observe that this number is significantly higher compared to Algorithm 4.2. On the opposite, we see that Algorithm 3.1 requires in average less iterations. Fi-

Table 3: Numerical results obtained with the iterative algorithmic procedures.

#	Algorithms 4.1 and 4.2 adapted from (Adasme et al., 2015)				Algorithm 3.1					Gaps	
	Opt_{It}	Time (s)	*#Cycles*	*#Iter*	Opt^R_{It}	Time (s)	*#Cycles*	*#Iter*	Opt^F_{It}	Gap_R %	Gap_F %
1	945.90	0.51	0	1	1443.61	1.09	10	2	945.90	52.62	0
2	1223.95	0.80	3	2	1899.73	0.80	8	1	1239.34	55.21	1.26
3	1574.80	0.76	4	2	2896.01	0.76	12	1	1574.80	83.90	0
4	1992.63	0.80	5	2	4269.07	1.08	32	2	2036.42	114.24	2.20
5	2781.27	1.26	8	3	4447.16	1.20	41	2	2787.50	59.90	0.22
6	3795.17	1.66	9	4	5800.14	1.26	41	2	3995.48	52.83	5.28
7	4233.11	1.25	8	3	5963.58	1.22	45	2	4328.84	40.88	2.26
8	5920.61	2.40	18	6	8952.44	1.23	58	2	6233.36	51.21	5.28
9	7979.07	0.89	12	2	13244.42	1.36	75	2	8441.78	65.99	5.80
10	11272.16	1.34	14	3	15160.96	1.51	97	2	11342.97	34.50	0.63
11	14325.01	1.63	18	3	23008.49	3.05	207	3	14661.88	60.62	2.35
12	17772.48	2.89	27	4	27771.65	2.95	168	2	18442.49	56.26	3.77
13	23021.58	12.30	45	11	36082.31	6.66	312	3	23959.25	56.73	4.07
14	28178.71	7.87	42	6	44084.55	6.28	235	2	29376.14	56.45	4.25
15	32776.31	16.27	55	9	51378.66	20.60	536	4	33437.84	56.76	2.02
16	35705.34	5.95	50	3	56294.82	10.57	323	2	36700.49	57.66	2.79
17	42688.50	27.17	71	10	65957.50	24.66	537	3	44117.89	54.51	3.35
18	50786.93	38.96	83	9	77440.29	25.01	413	2	52118.73	52.48	2.62
19	66770.38	71.56	92	9	101128.60	199.45	1090	4	68005.67	51.46	1.85
20	86307.85	153.10	125	11	127706.66	410.10	1007	3	87895.55	47.97	1.84
21	93444.25	122.41	127	7	141811.63	325.40	706	2	96921.84	51.76	3.72
22	122719.37	122.20	140	4	186448.77	948.96	919	2	126746.65	51.93	3.28

nally, we observe that solving MIP_1 with all the cycle elimination constraints found with Algorithm 3.1 allows to compute tight bounds when compared to the optimal solution of the problem. More precisely, these bounds are computed with gaps which are lower than 6% for most of the instances.

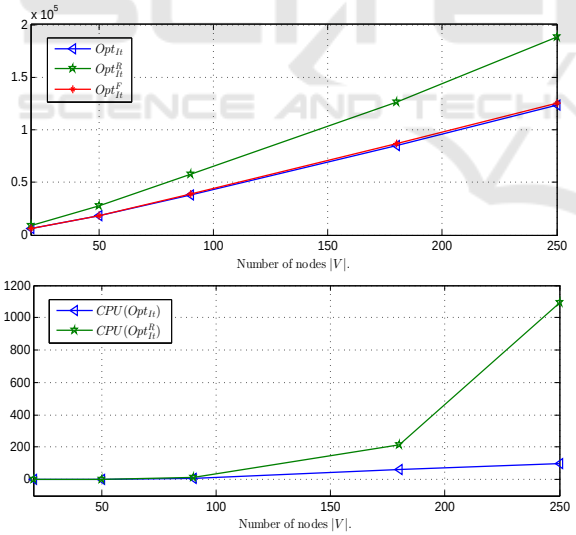

Figure 1: Average optimal solutions, upper bounds and CPU times obtained with the iterative Algorithms 4.2 and 3.1.

In order to give more insight with respect to the performance of Algorithms 4.2 and 3.1 when solving MIP_1. In Figure 1, we present some average numerical results for 20 instances randomly generated with dimensions of $|V| = \{20, 50, 90, 180, 250\}$ nodes, respectively. From Figure 1, we mainly confirm the trends observed in Table 3. We observe that in average obtaining optimal solutions with the iterative Algorithm 4.2 is more effective than computing upper bounds with Algorithm 3.1 in terms of CPU time. This is an interesting result as it confirms that Algorithm 4.2 allows to obtain optimal solutions for the large scale instances more easily. More precisely, in less CPU time than the instances presented in Table 3. Finally, we observe that the upper bounds obtained with Algorithm 3.1 are very tight when solving MIP_1 with all SECs, although they are obtained at a higher CPU time. In Figure 1, we do not plot averages for the number of cycles and iterations as they remain nearly the same as in Table 3 for all the instances. Similarly, we do not plot the average CPU times for Opt^F_{It} since they are slightly larger than those obtained for Opt^R_{It}.

5 CONCLUSIONS

In this paper, we consider a linear bilevel programming problem where both the leader and the follower maximize their profits subject to budget constraints. Additionally, we impose a Hamiltonian cycle topology constraint in the leader problem. In particular, models of this type can be motivated by telecommunication companies when dealing with traffic network flows from one server to another one within a ring topology framework. We transform the bilevel programming problem into an equivalent single level optimization problem and derive mixed integer linear programming (MILP) formulations. The topology constraint is handled by the means of two compact

formulations and an exponential one from the classic traveling salesman problem. Our preliminary numerical results show that one of the compact models allows to solve instances with up to 250 nodes to optimality with CPLEX in less than two hours. Finally, we propose iterative procedures that allow to compute optimal solutions in significantly less computational effort when compared to the compact models. Our main contribution in this paper is not theoretical, but mainly focussed on computational numerical results on a novel problem in the domain of bilevel programming. Our numerical results clearly show that solving the exponential model with the iterative procedure is by far more convenient than using the compact formulations which are more theoretical based approaches. In fact, we solve to optimality instances with up to 250 nodes so far, in less than 70 seconds in average compared to the higher CPU times required by the compact formulations.

As part of future research, we plan to develop further tests in order to confirm the behaviour of the proposed models and algorithms. Finally, we will propose new stochastic models and algorithmic approaches for this type of bilevel programming problems.

ACKNOWLEDGEMENTS

The first author acknowledges the financial support of the USACH/DICYT Project 061513VC_DAS.

REFERENCES

Adasme, P., Andrade, R., Letournel, M., and Lisser, A. (2015). Stochastic maximum weight forest problem. *Networks*, 65:289–305.

Audet, C., Hansen, P., Jaumard, B., and Savard, G. (1997). Links between linear bilevel and mixed 0-1 programming problems. *Journal of Optimization Theory and Applications*, 93:273–300.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to algorithms*. MIT Press and McGraw-Hill, Cumberland, RI, USA.

Dempe, S. (2002). *Foundations of bilevel programming. (1st ed.).* Freiberg: Kluwer Academic Publishers.

Dempe, S. (2003). Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints. *Optimization*, 52:333–359.

Floudas, C. and Pardalos, P. (2001). *Encyclopedia of optimization.* Dordrecht: Kluwer Academic Publishers.

Gavish, B. and Graves, S. C. (1978). The travelling salesman problem and related problems. *Operations Research Centre, Massachusetts Institute of Technology*.

Lee, D., Attias, R., Puri, A., Sengupta, R., Tripakis, S., and Varaiya, P. (2001). A wireless token ring protocol for intelligent transportation systems. In *IEEE Proceedings of: Intelligent Transportation Systems*, pages 1152–1157.

Letchford, A. N., Nasiri, S. D., and Theis, D. O. (2013). Compact formulations of the steiner traveling salesman problem and related problems. *European Journal of Operational Research*, 228:83–92.

Migdalas, A., Pardalos, P., and Varbrand, P. (1997). *Multilevel optimization: Algorithms and Applications.* The Netherlands: Kluwer Academic Publishers.

Miller, C. E., Tucker, A. W., and Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *J. Assoc. Comput. Mach.*, 7:326–329.

Scholtes, S. (2004). Nonconvex structures in nonlinear programming. *Operations Research*, 52:368–383.

Song, J. and Yang, O. (1997). Backbone networks using rotation counters. *IEEE Transactions on Parallel and Distributed Systems*, 8:1288–1298.

Thirwani, D. and Arora, S. (1998). An algorithm for quadratic bilevel programming problem. *International Journal of Management and System*, 14:89–98.

Vicente, L., Savard, G., and Judice, J. (1994). Descent approaches for quadratic bilevel programming. *Journal of Optimization Theory and Applications*, 81:379–399.

Wang, S., Wang, Q., and Rodriguez, R. (1994). Optimality conditions and an algorithm for linear-quadratic bilevel programs. *Optimization*, 31:127–139.