Bilevel Optimization Model for Determining Highway Tolls

José Luis González Velarde[1], Gabriel Pinto[1] and Fernando Camacho[2]

[1]Tecnológico de Monterrey
Av. Eugenio Garza Sada 2501 Sur
Monterrey, Nuevo León, 64849, México
gonzalez.velarde@itesm.mx, gabrielpinto_s@hotmail.com

[2]Facultad de Ciencias Físico-Matemáticas
Universidad Autónoma de Nuevo León
Pedro de Alba S/N, Ciudad Universitaria
San Nicolás de los Garza, Nuevo León, 66450, México
jose.camachovl@uanl.edu.mx

ABSTRACT. *The problem of determining an optimal set for the tolls for the arcs of a multiproduct transportation network is presented. The problem is formulated as a bilevel optimization problem, where the upper level consists of an administrator who establishes the tolls on the network, while the lower level is represented by a group of users who travel along the shortest paths with respect to a given travel cost. The objective is not only to increase tolls, but also to maintain an optimal flow on the arcs of the network in order to maximize benefits. A methodology to solve this problem using an optimization software at the lower level and the metaheuristic Scatter Search at the upper level is proposed.*

Keywords: bi-level programming, scatter search, toll optimization problem.

1. Introduction.

The importance of transportation for the economic and productive growth of any organization, including countries, is unquestionable and is indeed an area that has taken a great relevance within this subject is the one of the highway tolls. Tolls help to reduce congestion, and as well, they put the monetary load on the shoulders of the users of this infrastructure.

Existing Literature about highway tolls is very extensive; nevertheless, the great majority focuses fundamentally in congestion reduction and demand regulation, leaving the studies on toll allocation for the maximization of benefits somewhat left behind. A level of the problem tackled in this work will be modeled as a problem of minimum cost flow, which, like the rest of the models for transportation problems, has been widely studied.

As mentioned before, research about highway quotas are abundant, nevertheless the majority of these focuses in congestion reduction and other negative effects entailed by this (like contamination) by means of demand regulation (Cropper & Oates, 1992). Congestion is an important and increasingly common problem. The efforts made so far to solve this problem have not been successful mainly because once the solutions are constructed it has not been guaranteed that the new capacity be appropriately used. The efforts to attract people outside their vehicles have been equally ineffective (Button, 2004).

Highway pricing is a very simple concept that extends the common practice of using prices to reflect shortage of a certain resource and to assign this resource for its more efficient use, and occurs in practically any sector of the economy. The concession of lines in Singapore in 1975 is the classic case of study of a pioneering application, but in spite of its success, it is not commonly seen as a good example for the cities of Western Europe Occidental and North America. Nevertheless the scheme of Singapore offers evidence of the effects of highway pricing. The initial policy, collect a fee from vehicles entering downtown between 7:30 and 9:30 a.m. demonstrated to be inadequate since the traffic simply moved outside the periods of concession. Just a short time later this period of extended until the 10:15 a.m. and a period in the afternoons was included. The impact of this new scheme was impressive from the beginning since it obtained a reduction of 24.700 automobiles during the rush hours.

“Traffic Assignment Problem: Models and Methods” (Patriksson, 1994) reviews the evolution of the models and methods for the problem of estimating the flows of traffic balance in the urban zones and show the scope and limitations of the present traffic models.

Focusing specifically in the subject of this work, which tries allocating highway tolls for transport networks aiming at maximizing benefits, Camacho Vallejo (2009), develops and compares three solution methods for this problem in order to determine which one obtains better results. Those methods are based on approximation by gradient, Quasi-Newton and a direct resolution method, respectively. In this work the optimization model presented by Camacho Vallejo is solved developing a metaheuristic procedure based on Scatter Search (Glover 1998) and on an implementation presented by Laguna and Marti (2003) interacting with CPLEX a commercial optimizer, reaching better results than those obtained by Camacho Vallejo.

This work is divided into five sections: Section 1 is just introductory, with a literature review of the more relevant articles for this research topic. Section 2 shows the structure of a bilevel program and the problem to be solved is presented. In Section 3 the formulation used to solve the problem is presented. Section 4 gives a general description of the optimization tools used to solve the problem. In Section 5 the computational experiments and results are presented. And finally Section 6 shows the conclusions and recommendations for future work.

2. Bilevel Programming.

Bilevel programming problems are problems of hierarchic optimization, where a decision maker may be able to influence in the behavior of another decision maker in a lower level without completely taking control on his actions. The importance of bilevel programming is based on the fact that decision making in any large organization rarely comes from a single point of view.

Multilevel systems share the following characteristics (Bard, 1998):

- Existence of interactive decision making between the different hierarchical levels.
- Each subordinated level carries out its policies only after a superior level carries out its decisions.
- Each unit optimizes its net benefits independently, but they are affected by the actions of other units.
- The external effects on the problem of a decision maker are reflected in the objective function and in the feasible set of solutions.

Multilevel programming was defined for the first time in the decade of 1970s by Chandler and Norton as a generalization of mathematical programming, in their work they illustrate how two levels of programming can be used to analyze the dynamics of a regulated economy, concentrating in the agricultural development of the north of Mexico. (Chandler & Norton, 1977).

In order to be able to mathematically formulate the problem, it is assumed that the leader has control on the vector $x \in X \subseteq R^n$, and as well, that the follower has control on the vector $x \in X \subseteq R^n$. The leader starts selecting a vector x trying to minimize $F(x, y(x))$, which can be subject to certain constraints. The component $y(x)$ indicates that the leader's problem is implicit in the y variables. After observing the actions of the leader, the follower reacts selecting a y to minimize his objective function $f(x, y)$, subject to a set of restrictions.

This problem may be defined as follows:

$$\min_{x \in X} \quad F(x,y)$$
$$\textit{subject to} \quad G(x,y) \leqq 0$$

$$\min_{y \in Y} \quad f(x,y)$$
$$\textit{subject to} \quad g(x,y) \leqq 0$$

where $F, f: R^n \times R^m \rightarrow R^1$, $G: R^n \times R^m \rightarrow R^p$ and g:$R^n \times R^m \rightarrow R^q$. The sets X and Y may have additional constraints on their variables, such as no negativity or integrity of his variables. This formulation may be handled to generate new forms for the problem, changing for example the "min" operator by "max" according to the specifications of the particular problem.

The problem of highway toll allocation may be formulated as a bilevel optimization problem, where the upper level consists of an administration that establishes the tolls for the network aiming at maximizing the income, whereas the lower level is represented by a group of users who travel in the shortest paths with respect to a generalized cost. In order to be able to obtain a bilevel model a constraint of the type *argmin* has been added to the objective function of the follower that optimizes the flows in the arcs of the transportation network once tolls are allocated, in order to find in this way a balance between the established quotas and the number of users who use these roads.

3. Problem Formulation.

As already mentioned, the problem of optimization of highway tolls can be approached from the point of view of a leader and his follower, which interact in a multiproduct network $G = (K, N, A)$ defined by a product set (origin - destination) K, a set of nodes N and the set of arcs A. This last set is in turn partitioned in subset A_1 that represents the arcs with tolls and its complementary subset A_2, representing the toll free arcs. Each arc $a \in A$ is provided with a generalized cost C_a that represents the minimum cost to travel through each arc. One must also consider that each arc $a \in A$ in the network has a limit q_a in its capacity. As well as there is a fixed capacity for each arc and a product set K, there is also a parameter n_k that represents the existing demand for each product between the origin and destination nodes associated with product $k \in K$. Finally, the parameter t_a in the toll arcs $a \in A_1$ represents an extra cost to be determined to travel through these arcs. Leaving x_a^k to represent the flows between the arcs to fulfill the demands of each one of the existing products.

Considering that the quotas t_a cannot exceed a pre-established maximum value of t_a^{max} and that the flows x_a^k must be part of the optimal solution of the lower level which in turn is parameterized by the toll vector decided at the upper level, this problem can be formulated as a bilevel program with linear constraints as follows:

$$\max_{t,x} \sum_{k\in K} \sum_{a\in A_1} t_a x_a^k \qquad (1)$$
$$subject\ to: \quad t_a \le t_a^{max} \qquad \forall a \in A_1 \qquad (2)$$
$$t_a \ge 0 \qquad \forall a \in A_1 \qquad (3)$$
$$\left.\begin{array}{ll} x^k \in \min_x \sum_{a\in A_1} (c_a + t_a) x_a + \sum_{a\in A_2} c_a x_a & \\ subject\ to: \; -\sum_{a\in i^-} x_a + \sum_{a\in i^+} x_a = b_i^k & \forall i \in N \\ x_a \ge 0 & \forall a \in A \end{array}\right\} \forall k \in K \qquad (4)$$
$$\sum_{k\in K} x_a^k \le q_a \qquad \forall a \in A \qquad (5)$$

Family of equations 4:

$$x^k \in \min_x \sum_{a\in A_1} (c_a + t_a) x_a + \sum_{a\in A_2} c_a x_a \qquad (4.1)$$
$$subject\ to: \; -\sum_{a\in i^-} x_a + \sum_{a\in i^+} x_a = b_i^k \qquad \forall i \in N \qquad (4.2)$$
$$x_a \ge 0 \qquad \forall a \in A \qquad (4.3)$$

The objective of the leader is to maximize total benefits, which are the sum of the product of the tolls t_a and the users flow through the arc a (equation 1). However the set of lower constraints (family of equations 4) enforce the follower to assign flows to the shortest paths with respect to the present tolls, that is, the objective of the lower level is to minimize the total cost of the routes selected by the users (equation 4.1). The constraints at the lower level are in charge of the flow conservation (equation 4,2) and the nonnegativity in the flows (equation 4,3). Finally a constraint is added to prevent exceeding the capacities on the arcs with the assigned flows (equation 5).

4. Scatter Search.

Scatter Search is an evolutionary metaheuristic that has been successfully applied to solve hard optimization problems. It is based on formulations and strategies developed in 60s, but it was not but until 1977 that was officially proposed by Glover as a method by itself. In this year Glover described Scatter Search as "a method that uses a succession of coordinated initializations to generate solutions" (Glover, 1977).

As described in several articles in the literature (Glover, 1998; Laguna & Martí, 2003) and other implementations based on this framework, the methodology includes the following basic elements:

- Generation of a population *P*
- Extraction of a reference set *R*
- Combination of elements from *R* and update of *R*

The dimension and structure of the solution set in different evolutionary algorithms may vary. Genetic Algorithms, for example, work with the whole created population (typically 100 solutions), Memetic Algorithms work with small (and sometimes structured) population (Moscato, 2000), while SS works with a subset of 10 to 20 solutions from the set of created solution. This subset called reference set *R* is built from the population *P* (generated by the diversification generation method) with only a few solutions from *P*. The way the reference set is initialized, updated and rebuilt is a crucial aspect in Scatter Search performance. If the construction of a reference set was made only based on the solution quality, the reference set would be formed by selecting the best *b* solutions in *P*. Nevertheless, a desired characteristic in general search procedures, and particularly in SS, is an adequate balance between intensification and diversification.

Next step in the Scatter Search methodology is the combination of elements in the reference set. To accomplish this, two or more elements from *R* are chosen in a systematic way with the purpose of creating new solutions. This is achieved by the construction of certain subsets of solutions from *R* and by applying the combination method to solutions in each one of these subsets. This combination is intended to be intelligent so a better solution than those in the subset may be created. At this point, it is possible that as a result of the combination method an infeasible solution be created. In this case, the combination method must have a procedure to restore feasibility. As new solutions are being created, these will gain membership to the reference set not only by their quality, but by their degree of diversity. The general procedure may iterate several times to achieve a better quality in the created solutions. The way SS combines solutions and updates the set of reference solutions used for combination sets this methodology apart from other population-based approaches.

In order to solve the problem presented in this article, the implementation developed by the authors (Laguna & Martí, 2003) of the metaheuristic Scatter Search in their book "Scatter Search: Methodology and Implementations in C" was taken as a basis to solve the upper level problem where it seeks to determine the optimum toll combination to maximize the benefits in the transport network. The optimizer CPLEX was used to solve the problem at the lower level, where the optimal flows in the transport network is determine with the objective of minimizing the costs of traveling through the network.

The developed model starts generating P, an initial set of diverse solutions for the toll variables of the upper level in a random but controlled form to guarantee that the generated initial solutions are within the pre-established ranks (in this case between 0 and t_a^{max}). In order to generate these solutions one starts dividing the ranks of each variable (between 0 and 0 and t_a^{max}) in 4 sub ranks of same size. Next one of these sub ranks is randomly selected and a solution within the selected sub rank is generated.

Once the set initial of solutions has been generated, the method proceeds to improve these solutions, and given that the solution generation step constructs only solutions that are within the allowed ranks, the improvement method will always start to work with a feasible solution. The improvement procedure consists of the Nelder and Mead Simplex Method (Nelder & Mead, 1965) which is a classic optimizer for unconstrained nonlinear problems. This method requires of an input parameter that specifies the number of evaluations of the objective function, naturally, as the number of evaluations increases so does the quality of the solutions.

The next step consists of extracting the best generated solutions to form the reference set. This set must be formed 50% by solutions selected attending to their quality with respect to the objective function and 50% by its diversity. In order to obtain this solutions are sorted in decreasing order and the first 50% of the amount which is desired for the reference set are selected (for example, if it is desired to form a subgroup of reference of size 10, the top 5 solutions of the list are selected) and they are deleted from the initial set *P*. Next, the minimum Euclidean distance between the remaining solutions in the initial set *P* and the solutions selected for the reference set is computed. The solution with the maximum of the minimum computed distance is selected and is added to the reference set and deleted from the initial set *P*. Once this is done the process of is repeated until completing the desired size of the reference set, which as a result of this construction will contain the solutions with the greater quality and diversity.

Once this has been done, subsets of the reference set are generated to apply them a combination method. This combination method consists of creating 3 test solutions for each pair of solutions of the reference set.
The first test solution is computed as: $C1.x = x^{'} - d$
The second test solution is computed as: $C2.x = x^{'} + d$
The third test solution is computed as: $C3.x = x^{''} + d$
Where *d* is equal to: $d = r\frac{x^{''} - x^{'}}{2}$
And *r* is a random number from the interval (0, 1).

After the solutions have been generated with the combination method, they are processed by the improvement method and the best resulting solution is selected. This solution will replace the worst in the reference set, given that it is better. This cycle is repeated until the reference set no longer changes and all the solutions in the reference set have been processed by the combination method. At this point the diversification method is used to reconstruct half of the reference set and the search continues. This procedure is repeated for a pre established number of iterations, once this number is reached the method stops.

It should be mentioned that each time a new set of solutions is generated for the variables at the upper level, there are passed to the lower level and the flows are computed using these values as the costs of the arcs in the network, these new flows are used to measure the quality of the generated solutions when the objective value is computed.

5. Computational Results

In order to evaluate the bilevel model considered in this work, 3 different multicommodity transportation networks were considered. For the numerical experimentation 8 examples for the first network and 6 examples for the second and third network, respectively, were solved. The graphs and the parameters are the same as the data shown in the doctoral thesis “Comparing various algorithm’s performance; application to bilevel toll setting problem” (Camacho Vallejo, 2009). Finally, attempting to show that the methodology proposed is able to solve larger instances for the problem without any complications, the last example consisted in solving a variation from the second network where an additional commodity was considered.

The first graph consists in a transportation network with 7 nodes and 12 arcs (7 toll arcs and 5 toll-free arcs). The second graph is composed by 20 nodes and 35 arcs (15 toll arcs and 20 toll-free arcs). The third graph is formed by 25 nodes and 40 arcs (20 toll arcs and 20 toll-free arcs). The parameters considered in each example are presented in the work done by (Camacho, 2009).

The developed program to solve this problem was coded in C language and compiled with Microsoft Visual C++ 6.0. All the examples were executed in a personal computer Dell OptiPlex 960 with a processor Intel Core 2 Quad with 2.66 GHz and 3.21 GB of RAM, under the operating system Microsoft Windows XP Professional with Service Pack 3.

As mentioned in the previous section, the methodology designed to solve the upper level problem is the metaheuristic Scatter Search. Martí and Laguna (2003) (Martí & Laguna, 2003) developed a C implementation for this methodology, taking advantage of this, we considered it for this research. The CPLEX 11.1's software optimization libraries from ILOG were used in order to solve the lower level problem.

Through this research 21 instances were solved in order to validate the proposed methodology. To simplify the analysis we can divide the examples as follows:

- Graph 1: Conformed by 12 arcs. 8 instances.
- Graph 2. Conformed by 35 arcs. 6 instances.
- Graph 3. Conformed by 40 arcs. 6 instances.
- Variation of graph 2 considering 4 commodities. 1 instance.

The purpose of considering the first three sets of instances (not the last case) is to directly compare the results against the ones obtained by Camacho-Vallejo (Camacho Vallejo, 2009) in which the performance of four different algorithms were measured and the corresponding results are presented. The four algorithms are: a penalization method, an algorithm based in the gradient's approximation, a Quasi-Newton algorithm and a direct algorithm based in the Nelder-Mead's optimization method (Nelder and Mead, 1965). The results obtained in that work as well as the obtained in this research are shown in the next three tables.

- Results obtained from the instances considered for Graph 1:

Instance	Gradient approx	Quasi-Newton	Direct	**SS/CPLEX**
1	162.850	162.880	162.360	**180.990**
2	274.890	274.890	274.340	**274.950**
3	109.850	109.850	108.870	**199.990**
4	150.860	150.840	150.320	**146.550**
5	112.860	112.860	112.040	**133.840**
6	203.950	203.960	202.820	**199.973**
7	41.970	41.960	41.650	**41.976**
8	104.950	104.950	104.110	**125.990**

Table 1. Objective function value for each method for instances of Graph 1.

- Results obtained from the instances considered for Graph 2:

Instance	Gradient approx	Quasi-Newton	Direct	**SS/CPLEX**
1	1,342.235	1,342.763	1,341.835	**1,328.983**
2	7,184.852	7,184.804	7,184.422	**7,112.128**
3	1,577.953	1,577.911	1,577.661	**1,686.028**
4	420.701	420.772	420.257	**1,022.000**
5	764.931	764.961	763.890	**1,514.000**
6	2,350.855	2,350.875	2,350.356	**3,004.492**

Table 2. Objective function value for each method for instances of Graph 2.

- Results obtained from the instances considered for Graph 3:

Instance	Gradient approx	Quasi-Newton	Direct	**SS/CPLEX**
1	1,456.802	1,456.833	1,456.034	**1,631.644**
2	2,247.873	2,247.851	2,246.889	**2,611.994**
3	3,891.831	3,891.873	3,891.445	**3,263.977**
4	5,621.804	5,621.820	5,621.109	**11,339.610**
5	3,433.720	3,433.794	3,432.771	**3,419.900**
6	544.871	544.893	543.908	**868.594**

Table 3. Objective function value for each method for instances of Graph 3.

- Results obtained from the variation considered in Graph 2:

Instance	**SS/CPLEX**
1	**1,368.657**

Table 4. Objective function value in variation of Graph 2.

As can be seen from the tables presented above, the leader's objective function value were improved in most of the cases; and in cases in which the algorithm proposed did not improve the result, anyways obtained a close leader's objective function value compared with the results obtained in (Camacho Vallejo, 2009).

Now are listed the tables showing the percentages increase obtained by the bilevel method based in Scatter Search and CPLEX (for each of the different examples considered) with respect to the best result obtained in the previous work.

- Percentage increase for the instances considered for Graph 1:

Instance	Increase
1	11.12%
2	0.02%
3	82.06%
4	-2.86%
5	18.59%
6	-1.95%
7	0.01%
8	20.05%

Table 5. Percentage increase for the examples considered for Graph 1.

- Percentage increase for the instances considered for Graph 2:

Instance	Increase
1	-1.03%
2	-1.01%
3	6.85%
4	142.89%
5	97.92%
6	27.80%

Table 6. Percentage increase for the examples considered for Graph 2.

- Percentage increase for the instances considered for Graph 3:

Instance	Increase
1	12.00%
2	16.20%
3	-16.13%
4	101.71%
5	-0.40%
6	59.41%

Table 7. Percentage increase for the examples considered for Graph 3.

The mean obtained for the objective function value increase is **28.66%** for the examples tested in this work with respect to the previous work.

As is mentioned at the beginning of this section, also were measured the times required for obtain the objective function value reached for each of the considered examples. The results obtained are shown in the following tables, also are presented the times required by the methods proposed in (Camacho Vallejo, 2009).

- Time required to solve the instances considered for the Graph 1:

Instance	Gradient approx	Quasi-Newton	Direct	**SS/CPLEX**
1	2.107	2.023	2.950	**1,680.000**
2	2.449	2.433	3.235	**1,320.000**
3	1.573	1.426	2.035	**1,503.000**
4	3.414	3.173	3.415	**1,527.000**
5	2.006	1.940	2.002	**1,461.000**
6	2.178	2.156	3.782	**1,442.000**
7	1.308	1.277	1.259	**1,449.000**
8	3.065	2.899	1.959	**1,620.000**

Table 8. Time required (in seconds) for solve the examples considered for Graph 1.

- Time required to solve the instances considered for the Graph 2:

Instance	Gradient approx	Quasi-Newton	Direct	**SS/CPLEX**
1	601.000	517.000	723.000	**1,710.000**
2	781.000	748.000	917.000	**5,007.000**
3	492.000	437.000	714.000	**2,656.000**
4	278.000	220.000	439.000	**1,853.000**
5	644.000	591.000	835.000	**2,095.000**
6	831.000	806.000	980.000	**2,593.000**

Table 9. Time required (in seconds) for solve the examples considered for Graph 2.

- Time required to solve the instances considered for Graph 3:

Instance	Gradient approx	Quasi-Newton	Direct	**SS/CPLEX**
1	1,004.000	985.000	1,054.000	**4,175.000**
2	525.000	498.000	893.000	**2,820.000**
3	439.000	421.000	922.000	**3,480.000**
4	1,723.000	1,755.000	1,480.000	**2,778.000**
5	764.000	785.000	761.000	**3,030.000**
6	89.000	84.000	835.000	**3,520.000**

Table 10. Time required (in seconds) for solve the examples considered for Graph 3.

- Time required to solve the variation of Graph 2:

Instance	**SS/CPLEX**
1	**3,281.000**

Table 11. Time required (in seconds) to solve the variation considered for Graph 2.

The notorious increase in the time required for solve the examples by the methodology proposed based in Scatter Search and CPLEX is expected due to the nature of the metaheuristic. In order to make an equable analysis the difference obtained from the times required with this methodology and the best result obtained in the previous work, are presented the next tables:

- Differences between the times for the instances considered for the Graph 1:

Instance	Difference
1	1,677.977
2	1,317.567
3	1,501.574
4	1,523.827
5	1,459.060
6	1,439.844
7	1,447.741
8	1,618.041

Table 12. Differences (in seconds) between the time required and the best value obtained before.

- Differences between the times for the instances considered for the Graph 2:

Example	Difference
1	1,193.000
2	4,259.000
3	2,219.000
4	1,633.000
5	1,504.000
6	1,787.000

Table 23. Differences (in seconds) between the time required and the best value obtained before.

- Differences between the times for the instances considered for the Graph 3:

Example	Difference
1	3,190.000
2	2,322.000
3	3,059.000
4	1,298.000
5	2,269.000
6	3,436.000

Table 34. Differences (in seconds) between the time required and the best value obtained before.

The obtained mean of difference between the times required for the tested examples using the methodology here proposed against the results obtained in (Camacho Vallejo, 2009) is **2,007.732** seconds (33.462 minutes).

6. Conclusions and future work.

Upon completion of this research we can conclude that it is possible to maintain a profitable toll road system against competition from toll-free roads, as long as the right tools and a good planning are used. Since as we saw, although for each origin-destination pair there exist at least one that had at least one toll-free path benefits were always obtained, indicating that users are motivated to use this infrastructure.

It was also noted that the resolution's method (based on the metaheuristic Scatter Search and the optimizer CPLEX) that is proposed in this work to solve the problem of determining tolls for the roads gave significant improvements over the methods proposed in previous research which was taken as the basis for this research (an increase of 28.56% to be precise among all examples tested).

An important point to consider is the determination of the upper bound for rates in the toll arcs, because if this value is established in the wrong way could exists the case in which the model is too restricted and does not get the maximum benefit from the network (in the case when set a very low toll) or that the network is underutilized forcing customers to go through the toll free roads (in the case when set a very high toll).

On the other hand, if it is true that the benefits (in terms of value achieved in the objective function) when using this methodology are significantly higher, must be considered the increase in the time to obtain the results (33.5 minutes in average for each example), it is recommended to do a cost/benefit analysis in the particular situation which might be needed to solve this problem. Sacrifice some benefits in favor of achieving results in a more immediate form or sacrifice time for greater benefits in the results.

For further research, some improvements and recommendations are presented below:

- Use data from real cases to observe the behavior of the model proposed in this paper.
- Use another metaheuristic to compare their performance against Scatter Search.
- Reduce the time required by this method. Try to streamline the code or use other programming languages.
- Develop new models to solve the problem of assigning toll roads to compare it with the model considered in this research and in the previous works.

References

Anandalingam, G., & Friesz, T. (1992). Hierarchical optimization. *Annals of OperationsResearch*, 34, 1-4 .

Bard, J. F. (1998). *Practical Bilevel Optimization. Algorithms and Applications.* Kluwer Academic Publishers.

Bergendor, P., Hearn, D. W., & Ramana, M. (1997). *Congestion in Toll Pricing of Traffic Networks.* Gainesville, FL.: Center for Applied Optimization.

Button, K. (2004). *Road Pricing.* Fairfax: Center for Transportation Policy Operations and Logistics.

Camacho Vallejo, J. F. (2009). Comparing various algorithms performance: application to bilevel toll setting problem. *ITESM - Ph.D. Dissertation.*

Camacho, J. F., Kalashnikov, V., & Kalashnykov, N. (2008). A penalty function approach to solve the bilevel tolls problem . *Innovative Computing Information and Control*, 217 - 217 .

Celaya Bustamante, C. A. (2010). A Bilevel Optimization Model for Port Selection. Monterrey, *ITESM - Ph.D. Dissertation*.

Chandler, W., & Norton, R. (1977). Multi-Level Programming and development policy. *Working Paper No. 258. World Bank.*

Colson, B., Marcotte, P., & Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research*, 153 , 235–256.

Cropper, M. L., & Oates, W. E. (1992). Environmental economics: a survey. *Journal of Economic Literature*, 30:2, 675-740.

Dempe, S. (2003). Bilevel Programming - A survey. *Technical University Bergakademie Freiberg, Germany* .

Edmonds, K. & Karp, R.M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM* . 19:2, 248-264.

Garrido, A., & Onaindía, E. (1999). Un algoritmo para a optimización de rutas de transporte. *Working paper.*

Glover, F. (1977). Heuristics for Integer Programming Using Surrogate Constraints. *Decision Sciences*, 8, pp 156-166.

Glover, F. (1998). A template for scatter search and path relinking. In **Hao, J.K.; Lutton, E.; Ronald, E.; Schoenauer, M. & Snyers, D.** (Editors), *Artificial Evolution: Third European Conference*, Lecture Notes in Computer Science, 1363, 13-54. Springer.

González-Velarde, J.L., & Martí, R. (2008). Adaptive memory programing for the robust capacitated international sourcing problem. *Computers & Operations Research*, 35, 797-806.

González-Velarde, J. L., Ada M., Á., & De Alba, K. (2005). Grasp Embedded Scatter Search for the Multicommodity Capacitated Network Design Problem. *Journal of Heuristics*, 11:3, 233 - 257.

Laguna, M., & Martí, R. (2003). *Scatter Search: Methodology and Implementations in C.* Kluwer Academic Publishers.

Malisani, E. A. (1989). *Optimización Industrial: Vol. 1.* Barcelona: Marcombo.

Martí, R., & Laguna, M. (2003). Scatter Search: Basic Design and Advanced Strategies. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. 19*, 123-130.

Moscato, P. (2000). Memetic Algorithms. In Pardalos, P.M. and Resende, M.G.C. (Editors), *Handbook of Applied Optimization*. Oxford University Press, USA.

Nelder, J. A. and Mead, R. (1965). *A simplex algorithm for function minimization* The Computer Journal, 7:4, 308-313.

Patriksson, M. (1994). *The Traffic Assignment Problem: Models and Methods.* VSP International Science Publishers.

YALMIP. (s.f.). *YALMIP WIKI*. Recuperado el 2010, de http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php?n=Examples.Bilevel

Zadeh, N. (1973). A bad network problem for the simplex method and other minimum cost flow algorithms. *Matheatical Programming*, 5, 255-266.