A global descent heuristic for bilevel network pricing

P.A. Lotito [*†] E.M. Mancinelli [‡†] J.-P. Quadrat [‡] L. Wynter [§]

July 29, 2004

Abstract

We present a method for solving bilevel revenue management problem on networks. In this problem, a network manager seeks to set link prices optimally on a network so as to maximize revenue, where the users' response to the prices, their route choices, is given by a secondary, parametric optimization problem. This is an important special case of the general bilevel programming problem. We demonstrate that local solutions to this problem can be arbitrarily bad, and that local algorithms in many cases will stop at very poor solutions. Through a geometrical study of the non convex nature of the implicit objective, we develop a heuristic procedure that, in addition to being very easy to implement, performs remarkably well on this class of problems, often converging to the global optimum.

Keywords

Revenue management, bilevel program, MPEC, network equilibrium, route choice

1 Introduction

Bilevel programs are an important class of highly non convex, and non differentiable, optimization problems. These problems are characterized by two levels of decisions, the upper-level, or leader, has a control variable, $y \in \mathbb{R}_+^{m_1}$, and an objective that he seeks to optimize as a function both of his control, y, and the response, or state variable, to his control, $x \in \mathbb{R}_+^{m_2}$. The lower-level, or followers' problem, is a secondary (and conflicting) parametric optimization problem in x, where the parameter is the leader's control variable, y. This paradigm underlies Stackelberg leader-follower games from game theory, but also describes a number of problems in engineering (optimal network design [8], structural optimization [3], chemical process design with equilibrium, and applications of optimal pricing). We are interested in a particular class of applications of bilevel programs, that is optimal network pricing, also known as bilevel revenue management.

[*]INRETS, 2 Av. du G. Malleret-Joinville, F-94114 Arcueil, France
[†]Inst. de Matemática Beppo Levi, UNR-CONICET, Argentina
[‡]INRIA, Rocquencourt, Domaine de Voluceau, B.P. 105, 78153 Le Chesnay, France
[§]IBM Research, P.O. Box 704, Yorktown Heights, NY 10598, USA

The problem of optimal network pricing can be described in terms similar to the general bilevel program. A leader, or network manager, seeks to set prices, y on the links of a network so as to maximize his revenue; these prices are then the control variables. The network users seek, on the other hand, to minimize their overall or individual costs for using the network, where users' costs are both delay-related and monetary. The monetary costs are precisely the prices y set by the leader. Solving then a parametric optimization problems to describe the users' problem, a flow vector $x = x(y)$ is produced, given a particular set of prices, y. This notation assumes that the user's response vector, x, is unique for each price vector, y, and this blanket assumption will be in effect throughout this work. The network manager's revenue is then precisely the scalar product of users' flow on the network and link prices paid by each user, $x(y)^T y$.

Introducing a graph, $G = (N, A)$, with dimensions n for the node set N, and m for the arc set, A, the bilevel program for revenue management can be expressed as

$$\max_y f(x(y), y) = y^T x(y) = \sum_{a \in A} y_a x_a(y) \tag{1}$$

subject to

$$y \in Y \tag{2}$$

$$x = \arg\min_{x \in X} \{g(x, y) = \sum_{a \in A} t_a(x) + x_a y_a\} \tag{3}$$

The variables' dimensions are then $m_1 \leq m$ and $m_2 = m$ for y and x, respectively. That is, one may wish to price up to m_1 of the m links. We will assume henceforth that the upper- and lower-level feasible sets, Y and X are polyhedral. In particular, the lower-level feasible set, X is defined by usual network constraints, including flow conservation, here expressed over paths, and demand satisfaction, along with non-negativity:

$$X = \{Nh = d,\, \Lambda h = x,\, x \geq 0\}, \tag{4}$$

where $h \in \mathbb{R}^k_+$ is the vector of flows on the k acyclic paths, N is a $w \times k$ 0-1 incidence matrix, w being the number of origin-destination pairs, and $d \in \mathbb{R}^w_+$ the vector of origin-destination demands. Finally, Λ is a $m \times k$ 0-1 incidence matrix taking path flows to arc flows. Note also that the lower-level feasible set, X does not vary with y. While this involves a loss of generality, the bilevel revenue management problems encountered in practice do indeed satisfy this property. Indeed, when the price vector enters the constraints through, for example, a demand function, that is, $Nh = q(p)$, assuming that $q : \mathbb{R}^{m_1} \mapsto \mathbb{R}^w_+$ is invertible, the users' problem can be re-cast as an *elastic* demand equilibrium problem, as follows:

$$x = \arg\min_{x \in X} \{g(x, y) = \sum_{a \in A} t_a(x) + x_a y_a - \sum_{i \in W} q^{-1}(d)\}, \tag{5}$$

where W is the set of origin-destination pairs.

When the lower-level, network flow or equilibrium, objective, $g(\cdot, y)$ is strictly convex for each $y \in Y$, the optimal user response, x is unique; we will make this assumption throughout. Note also that we have selected a particular functional form through which the prices enter the users' problem, $t(x) + x^T y$; they appear in an additive fashion for each marginal user, and do not appear in the

nonlinear term $t(x)$. Again, some generality is lost through this assumption, but we have chosen to enforce it for three reasons: (i) models of many practical problems do indeed take this form, (ii) the resulting problem is decidedly non-trivial, and (iii) we are able to obtain a highly effective algorithm for this case.

1.1 Review of the literature

Bilevel programs in general, and the bilevel revenue management problem is no exception, do not have convex or differentiable objective functions. Most methods reported in the literature for solving general bilevel programs are designed to stop at stationary points of the problem. The most practiced methods are arguably the basic subgradient method and the penalty method. The subgradient method relies upon sensitivity analysis of the lower-level problem to produce an estimate of the subgradient; a formula for obtaining the subgradient of a parametric network equilibrium problem and the conditions under which it holds were provided in [12, 13]. The authors in that reference actually suggest embedding the subgradient within a bundle algorithm so as to obtain much faster and more robust descent method; Bundle codes for nonconvex problems are not, however, readily available and are quite time consuming to implement. As a result, subgradients are often used alone. Indeed, in spite of their shortcomings in terms of excessively long computation times, practical results with subgradient methods for bilevel programs are reasonably good (see, for example, [3]).

Another popular approach to solving bilevel programs is the penalty method, in which the constraint that the response variable, x, solves a secondary optimization problem, is expressed in terms of its Karush-Kuhn-Tucker (KKT) conditions, and the complementarity term from the conditions is then penalized in the objective. As usual, a series of penalized problems is then solved, for increasing values of the penalty parameter. Depending on the choice of differentiable or non-differentiable penalty parameter, inexact or exact penalty approaches are obtained. (See [11] for example). Similar to the classic penalty method are the so-called smoothing methods, which approximate the non-differentiable penalized complementarity term from the KKT conditions within the objective by a smooth function, and then let the smoothing parameter tend towards zero progressively. (see [4]).

All of the methods described above can be guaranteed under appropriate conditions, to converge to a stationary point of the bilevel program. We will show in this paper, however, that stationary points of the bilevel revenue management problem can be very far from optimal. Furthermore, we will illustrate that there tend to be large regions of *strong stationarity*, or local, non-global optima close to regions of very low revenue. Therefore, the above approaches may often give very poor and non-robust results in practice.

What is therefore needed, if one is interested in solving bilevel programs, and especially the bilevel revenue management problem, in practice, is a global algorithm. However, global approaches for general bilevel programs are few and far between. Due to its non-convexity coupled with non-differentiability, the problem is inherently difficult. At the same time, there is a combinatorial nature to the problem, in that the non-convexities arise from the complementarity terms of the lower-level KKT conditions. If those complementarity equations are few enough to permit enumeration, then with the addition of a binary variable for each, a branch-and-bound type method can be used to solve the bilevel program exactly to its global optimum. This approach is adopted in [5, 15]. However, as one might expect, the number of variables cannot be too high for the effective use of solvers on

the resulting mixed-integer program (MIP). Another reference on the same topic is that of [1], who propose using a branch-and-cut approach on larger problems in which the lower-level function is quadratic.

In short, the lack of methods available for solving bilevel programs with general costs to global optimality, exactly or approximately, can be attributed to the intrinsic difficulty of this problem type. Here, we adopt a different approach, which is to study a *particular* class of bilevel programs, that of the bilevel revenue management problem of (1)–(3), and develop a method that exploits the particularities of that class.

In this paper, then, we begin by presenting a geometric study of the implicit objective function. Indeed, a particular structure of the implicit objective becomes apparent and can be shown to hold for many different lower-level cost functions. It becomes clear that stationary points, or local solutions, to the problem can be arbitrarily bad. We are, however, able to exploit the particular problem structure to develop a method that seeks a global optimum of the bilevel revenue management problem. The method is very easy to implement and can be shown to be remarkably effective in practice.

The geometric study of the implicit objective function is provided in Section 2. The global method is then described in Section 3, and computational results with the method are provided in Section 4. A discussion of interesting topics for further study is presented in Section 5.

2 Geometry of the leader's implicit objective

Searching for the global optimum of a non-convex function over unknown terrain is a daunting task. While methods and off-the-shelf software packages do exist, it is clearly preferable to have some notion of the surface that one wishes to optimize over, whenever possible. As it turns out, the surface generated by the implicit objective function of a bilevel revenue management problem over a network possesses a very particular structure.

2.1 The structure of the implicit objective surface

In this series of illustrations, we solve the lower-level network routing (or network equilibrium) problem over the 9-node, 36-link network given in Figure 1, varying the different parameters of the lower-level problem, such as polynomial degree (in flow variable, x) and constants of the lower-level objective, as well as the constant demand vector d, of the right-hand side.

The Figure 2 illustrates this surface, when the lower-level users' objective function is given as

$$g(x, y) = \sum_{a \in \mathcal{A}} (t_{0a} x_a + \alpha_a \, x_a^2 + x_a y_a), \tag{6}$$

with link-specific constants, t_{0a} and α_a. That is, the users' cost is quadratic in the flow variable, x. The dimension of the upper-level control variable, that is the link prices, y, is $m_1 = 2$ whereas $m = 36$ links in the first set of figures. Later, we will provide 3-dimensional slices of a surface, for

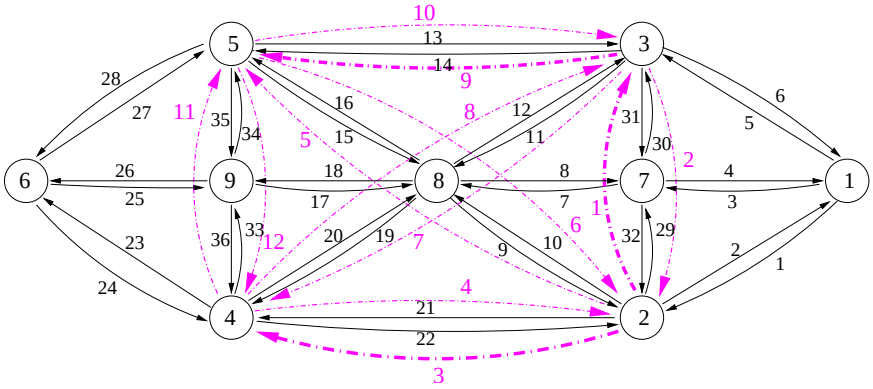


Figure 1: The 9-node, 36-link network

the same network, where the number of links with prices, $m_1 > 2$. Before interpreting this structure, we illustrate the surface for a number of different functional forms at the lower-level.

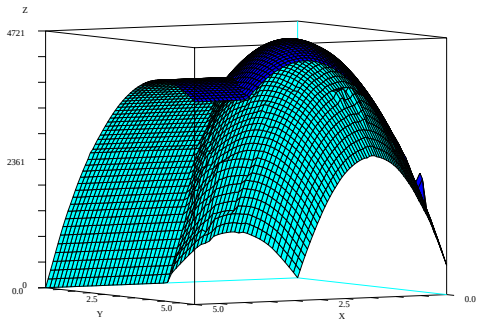


Figure 2: The upper-level implicit objective surface with quadratic lower-level costs

Figure 3 illustrates the upper-level response surface when $g(\cdot, y)$ is a cubic function for each y given by

$$g(x, y) = \sum_{a \in \mathcal{A}} t_{0a} x_a + \alpha_a \, x_a^3 + x_a y_a \tag{7}$$

and in Figure 4, the lower-level function is quartic in x for each y:

$$g(x, y) = \sum_{a \in \mathcal{A}} t_{0a} x_a + \alpha_a \, 10^{-3} \, x_a^4 + x_a y_a \tag{8}$$

However, modifying the degree of the lower-level polynomial network cost function is not the only criteria that can change across relevant users' problems at the lower-level. It is important as well to verify that the fundamental structure persits when the other parameters of the problem vary.

Figure 5 illustrates a cubic lower-level cost function with a right-hand side demand vector, d

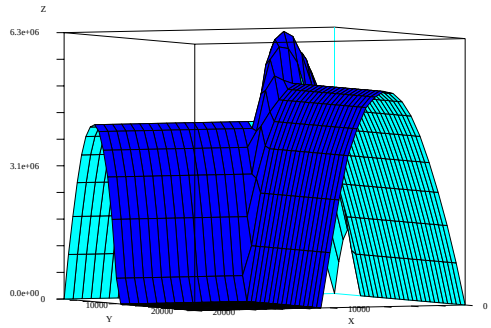


Figure 3: The upper-level implicit objective surface with cubic lower-level costs

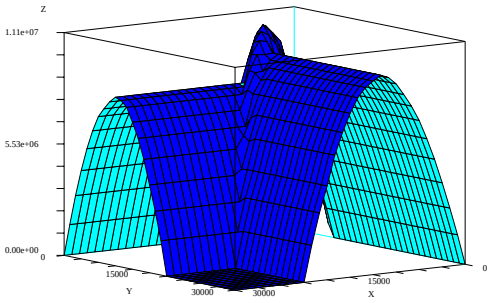


Figure 4: The upper-level implicit objective surface with quartic lower-level costs

which is doubled. This has the effect of shifting flow to the more nonlinear portion of the (here, cubic) delay function, $t(x)$.

Letting now the dimension of y increase to more than two, we provide in Figure 6 a number of 3-dimensional slices of the upper-level response curves.

2.2 An interpretation: network effects

It is clear that the figures of the previous section all exhibit a common fundamental structure. A first and very important observation is that algorithms which seek a stationary point can become hopelessly "stuck" on the long regions of constant revenue, along the axes. While far from the global optimum, this infinite number of locally optimal solutions are not robust in the sense that a slight deviation from this region of local optimality would lead to zero revenue.

An interpretation of why this shape arises so frequently can be found in the network structure. Recall that the control variables are prices on the links of the network, whereas demand is satisfied

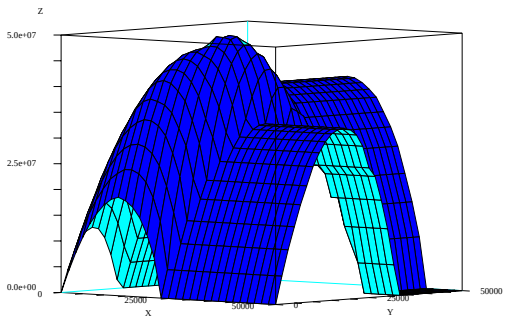
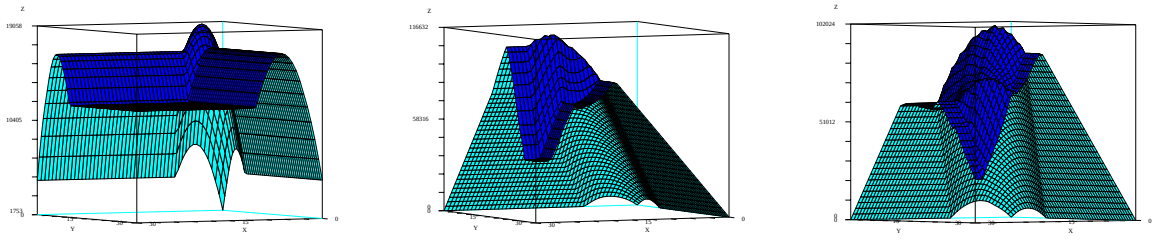


Figure 5: The upper-level implicit objective surface with higher demand levels



(a) $y_{10} = 2.4$ (b) $y_{21} = 27$ (c) $y_2 = 26.5$

Figure 6: Three slices of a 4-dimensional response curve, fixing one of the three link prices, y_j, $j = 10, 21$ and 2

at the lower-level of the problem along routes of the network. That is, demand can continue to be satisfied by a shift across routes, if the price on a link of the path becomes too high, relative to the other path costs. However, the upper-level objective gives the revenue over the entire network. Since there is a high multiplicity of routes, and demands, it is usual for the revenue to remain non-zero along axes of increasing price, up to a quite distant point, since revenue is non-zero on other links of the network.

This very clear *network effect* is confirmed by the fact that, in the figures, the origin gives (naturally) zero revenue, when only two links can have non-zero prices (that is, when $m_1 = 2$, as in the first set of figures). This, again naturally, is not so, when the figure represents a slice of a higher-dimensional control variable simulation. In that case, once the flow on the link whose price is increased no longer has flow, the revenue will remain flat, but non-zero, due to the presence of non-zero prices on other links. These long regions will thus present zones of an infinite number of uninteresting local, non-global, optima. When the network possesses at least one non-priced route serving each OD pair, the revenue will, eventually, return to zero, since the longer, free, routes will be preferred for sufficiently high prices.

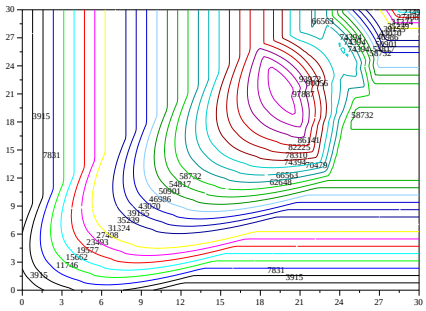


Figure 7: Level curves for the upper-level surface with a quadratic lower-level objective

The network effect is quite dramatic. While rendering the use of generic local algorithms quite risky, in that they may converge to non-robust and far-from-optimal solutions, it provides at the same time a very particular structure that can be exploited by a specialized algorithm.

3 The algorithm: Cyclic decomposition method for bilevel programs over networks

An illustration of the level curves of the upper-level implicit objective function surface, provided in Figure 7, helps to motivate the algorithm.

While the details may vary, the level curves of the upper-level objective surfaces are all similar, as the 3-dimensional curves of the previous section were all similar. From these level curves, a cyclic decomposition approach appears as a natural solution to finding the global optimum, and especially, to avoid stopping in the regions of local non-global optima.

Indeed, it is easy to see that if one were to optimize globally along each coordinate axis, with a one-dimensional global solver, and then change axes from the optimal point of the previous axis searched, then in a few steps, one would reach the global maximum of the implicit objective surface. The following algorithm does precisely that. An illustration of the path of the algorithm on the example used to produce the level set of Figure 7 is provided in Figure 8.

The steps of the algorithm are given below:

1. Set iteration counter $j = 0$, block index counter $k = 1$, $\epsilon > 0$, $norm1 = norm2 = 1$.[1] Find an initial starting point y^0 and a corresponding flow vector x^0.
2. Solve the one-dimensional global optimization problem: Find y_k^{j+1} solution of:

$$\max_{z} \quad \Psi_k(y_1^{j+1}, .., y_{k-1}^{j+1}, z, y_{k+1}^{j}, ..., y_{m_1}^{j}) \tag{9}$$

[1]Note that in this implementation, a block is equivalent to a link of the network.

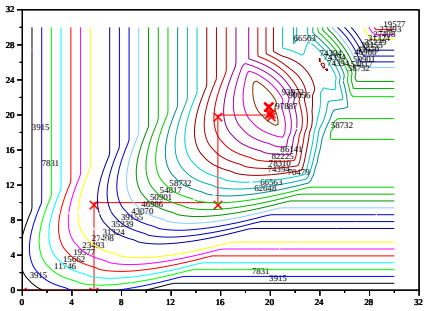


Figure 8: Solution path of the algorithm

where

$$\Psi_k(y) = x(y) \cdot y$$

and

$$x(y) = \arg\min_{x \in X} \sum_{a \in A} t_a(x_a) + xy$$

3. If $k \neq m$, then update block index counter $k = k + 1$ and return to Step 2. Otherwise Compute $norm1 := ||x(y^{j+1})y^{j+1} - x(y^j)y^j)||$ and $norm2 := ||y^{j+1} - y^j||$
If $\min\{norm1, norm2\} \leq \epsilon$ then terminate with y^{j+1} as the approximate solution.
Otherwise update iteration counter $j = j + 1$, set $k = 1$, and return to Step 2.

Remark 1 (Gauss-Seidel versus Jacobi) *Note that this algorithm is analogous to the Gauss-Seidel method for the solution of nonlinear equation systems, rather than the similar Jacobi method for nonlinear equations, as it updates the values of variables in blocks already examined, even within a single (outer) iteration.*

The following proposition presents a particular continuity property of the implicit upper-level objective.

Proposition 1 *Let $g(\cdot, y)$ be strongly convex and continuously differentiable for each $y \in Y$, the lower-level feasible set X be nonempty, and the partial gradients, with respect to x, of its active constraints, be linearly independent. Suppose that, for a subset of values of y, $\Gamma \subset Y$, the one-dimensional implicit objective (9) in each subproblem has a unique* global *optimum for all $y \in \Gamma$. Then, the decomposed implicit partial objective, Ψ_i, is continuous on Γ for every i.*

Proof: Under the assumptions imposed upon g and X, the optimal users' response $x(y)$ exists, is unique, and is locally Lipschitz continuous and directionally differentiable for every $y \in Y$. (See, for example, [14, 16]). Since Ψ_i is one-to-one and $\arg\,\text{global}\,\max_{y \in \Gamma} \Psi_i$ is unique, the continuity of the mapping on Γ follows.

As can be seen from its description in the previous section, the algorithm presented is essentially a cyclic decomposition, or block coordinate descent method applied to the bilevel program, where each block is an individual scalar variable, y_i. There are two technical reasons for which convergence of the algorithm to even a local optimum, or a stationary point, of the original bilevel network pricing problem cannot be guaranteed.

1. The first issue is the inherent non-differentiability of the upper-level, decomposed, implicit objective function, 1 (1). It is well known that, when the objective function is non-differentiable, cyclic decomposition methods may stop prematurely at a non-stationary point of the problem. See Figure 9 for an illustration of how non-differentiability may lead to this undesirable result.

2. The second stumbling block to ensuring convergence of the algorithm is the possible discontinuity of the decomposed, implicit mapping $\Psi_i(y^n_{\neq i}, y_i)$. In Proposition 1, we assume that the global optimum of the decomposed implicit objective is unique over a subset of the feasible region of y. Indeed, when multiple local optima exist, and the same value does not remain the global optimum for all $y \in Y$, there will be a set of points (of measure zero) for which at least two local optima have the same objective value. At the point where the global optimum shifts from one to the other value, the response curves ($y_1(y^n_2)$ and $y_2(y^n_1)$, for the case of a two-price model) do not intersect, and no solution exists. See Figures 10 and 11. For this reason, we assume, as above, that it is possible to isolate a subset, $\Gamma \subset Y$, in which the global optimum, y^*, remains globally optimal for all $y \in \Gamma$.

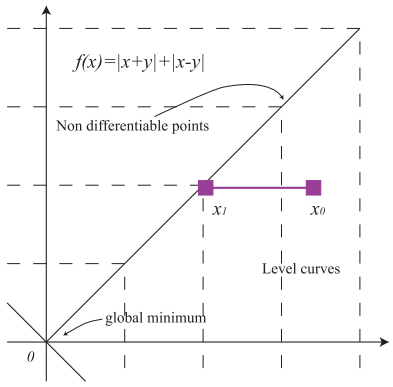


Figure 9: Premature stop of the relaxation method in the case of non differentiability

The above two points summarize the technical difficulties associated with the algorithm. However, the second point in particular poses little to no practical problem for the algorithm's convergence. As mentioned above, while the subset $\Gamma \subset Y$ can generally not be identified a priori, it is also true that the set of values, y, at which the response curves do not intersect is of measure zero. Therefore, short of starting the algorithm precisely at such a point y^0, there is little chance of stopping in such a region. The first issue, of using a method intended for differentiable problems on a non-differentiable one, must be handled with care. One way of adapting the method to reduce the risk of stopping at a non-stationary point is to include some (possibly infinite) number of steps in which at least two variables are updated simultaneously.

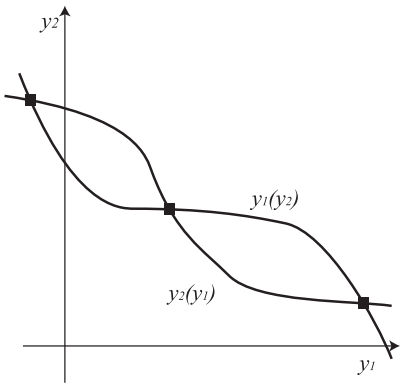


Figure 10: Multiple Nash equilibrium points

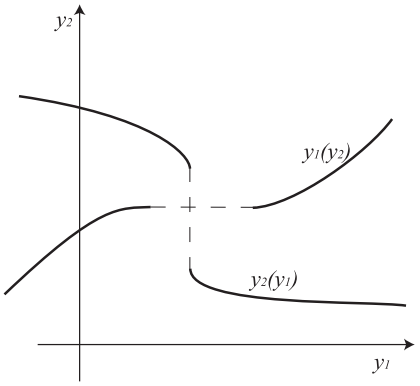


Figure 11: No Nash equilibrium points (discontinuity of one of the optimal responses).

All this having been said, the empirical performance of the algorithm is excellent. In the next section, we perform tests with the original version of the algorithm and demonstrate its empirical convergence not only to a solution, but further to the *global* optimum of the bilevel network pricing problem of (1)–(3)). We provide computation times that demonstrate its practical superiority to the global approach of dividing the feasible domain into a grid and performing a systematic exploration of the space.

4 Numerical experience

In spite of the possibility of the proposed algorithm to stop at non-stationary points of the bilevel network pricing problem, we found that the method works remarkably well, in that it stops not only at a stationary point of the original problem but, in the examples for which we were able to obtain the global solution through another algorithm, our method converged to that global optimum.

This section illustrates the results obtained on small and medium-sized networks with different

functional forms for the (lower-level) network costs, as well as different dimensions of the price vector. We compare the algorithm with a brute force method for identifying the global optimum of the problem, namely a global grid search over the induced feasible region. The grid was made finer and the grid search re-run around a solution, when our algorithm found a better solution than the grid search did. In all cases, solutions found by our algorithm were at least as good as those found by the initial grid search, and in all cases, were confirmed by a more refined grid search. Naturally, our algorithm was several orders of magnitude faster than the grid search.

Two variants of our decomposition method were implemented: in the first, for each uni-variate subproblem optimization, a 1-dimensional grid search was employed. The other variant involved using the one-dimensional global optimization routine of Neumaier, known as GLS, (see [6]) for the subproblems, which as can be seen from the tables, accelerated the convergence rate of the method.

Example 1 Consider again the network of Figure 1. The link cost functions are linear, giving rise to quadratic lower-level problems. We add the following fixed quantities to the time function of links 2 and 29: $t_2^0 = t_2^0 + 15$ and $t_{29}^0 = t_0^{29} + 20$. These fixed costs represent fixed, non-zero, prices on those links.

We allow the link prices that are to be optimized on links 10 and 21, namely, $y_{10},\ y_{21}$, to lie in the interval $[0, 30]$.

Algorithm	Time (s)	Outer iter.	Inner iter.	Function val.	Precision
Global grid search	4h23m42s	40000	-NA-	53288.987	2e-4
Decomp./1-dim grid search	7m13s	6	200	53288.987	2e-4
Decomp./GLS	1m42s	9	25	53293.986	$1e-4$

The benchmark, or reference, value for the maximum revenue in this example is 53301.383, and was computed with a local, refined, grid search around the optimum produced by the initial grid search summarized in the table. The precision was computed with respect to this benchmark value as: Precision=(Benchmark value-Function value)/Benchmark value.

The best value obtained with our algorithm, that is, Decomposition/GLS, after increasing the search interval from for the link prices from [0,30] to [0,45] was 53301.210782291. The obtained precision for this value was 3e-6. The additional computation time for this increase in precision was negligible.

Example 2 In this second example, we add the following fixed quantities to the cost functions of links 2 and 29: $t_2^0 = t_2^0 + 0.5$ and $t_{29}^0 = t_0^{29} + 0.6$.

We consider the range of link prices on links 10 and 21 to be $y_{10},\ y_{21} \in [0, 5]$, respectively.

Algorithm	Time (s)	Outer iter.	Inner iter.	Function val.	Precision
Global grid search	4h15m51s	40000	1	4722.814	$7.9e-6$
Decomp./1-dim grid search	29m33s	9	200	4722.814	$7.9e-6$
decomp./GLS	49.5s	6	35	4722.071	$1e-4$

As before, the column *Precision* compares the values of the revenue function obtained with each algorithm to the highest value obtained (here, equal to 4722.839) computed with a refined, local grid search around the optimum noted in the table obtained with the global grid search.

The best value obtained with our Decomposition/GLS method by modifying the search interval of the link prices to [0,7] was 4722.839, and the precision obtained for this value was essentially zero: 2e-11.

The primary observation to derive from these examples is that the brute force method took orders of magnitude more time to achieve a comparable solution to our decomposition method. In the first example, the brute force method was run until it achieved roughly the same precision as our method; the computation time with our method was 2 minutes, as opposed to 5 hours with the global grid search. In the second example, the brute force method was stopped with a precision level significantly lower than we were able to achieve with our decomposition method; to get there, it still took more than 20 times as long.

5 Conclusions and Discussion

While we have found that in all networks that we have constructed, the algorithm we propose does converge to the global solution, we have also shown that as defined, the method need not converge under all circumstances. Consequently, it is of interest to develop a version of the algorithm which could be proven to converge to at least a stationary point of the original problem (1)–(3).

To this end, we conclude with a conjecture: that the following modification of the algorithm may ensure convergence to a stationary point of the original bilevel network pricing problem.

The proposed modification involves inserting every several iterations a *mixed* search, rather than the coordinate searches used in the algorithm presently. In other words, the algorithm would cycle over each coordinate direction one or more times, and then the next iterate would involve updating more than one, and possibly all, price variables simultaneously. Then, the following iteration(s) would again cycle over the coordinate directions, and so on, repeating the sequence of coordinate- and mixed-searches.

The idea of such a modification would be to maintain the empirically good convergence rate of the heuristic, while attempting to overcome the risk of stopping at a non-differentiable, but non-stationary point, as illustrated in Section 3.

Acknowledgments

This work was supported in part by a grant from the French Programme de Recherche et Développement sur les Transports (PREDIT).

References

[1] C. Audet, P. Hansen, B. Jaumard, and G. Savard, 'A branch and cut algorithm for nonconvex quadratically constrained quadratic programming', *Mathematical Programming* 87:1 (2000), 131–152.

[2] Auslender, A. *Optimisation : Méthodes Numériques*, Masson, Paris, 1976.

[3] S. Christiansen, M. Patriksson, and L. Wynter, 'Stochastic Bilevel Programming in Structural Optimization' *Structural Optimization*, 21 (2001) 361–371.

[4] F. Facchinei, H. Jiang, and L. Qi, 'A smoothing method for mathematical programs with equilibrium constraints'. *Mathematical Programming*, 85 (1999) 107–134.

[5] Z.H. Gumus and C.A. Floudas, 'Global Optimization of Nonlinear Bilevel Programming Problems', *Journal of Global Optimization*, 20, (2001) 1–31.

[6] W. Huyer and A. Neumaier, 'Global optimization by multilevel coordinate search', *J. Global Optimization* 14 (1999), 331-355. For codes, see http://www.mat.univie.ac.at/ñeum/glopt/software_g.html

[7] P.A. Lotito, E.M. Mancinelli, J-P. Quadrat and L. Wynter,'CiudadSim: Scilab Traffic assignment toolboxes', see http://www-rocq.inria.fr/scilab/CiudadSim/index.htm

[8] P. Marcotte, 'Network design problem with congestion effects: A case of bilevel programming', *Mathematical Programming* 34 (1986) 142–162.

[9] P. Marcotte, G. Savard and D. Zhu, 'A trust region algorithm for nonlinear bilevel programming', *Operations Research Letters* 29 (2001) 171–179.

[10] P. Marcotte and G. Savard, "Bilevel Programming: Algorithms", *Encyclopedia of Optimization*, Floudas and Pardalos, eds. Kluwer Academic Publishers (2001)

[11] P. Marcotte and D.L. Zhu 'Exact and inexact penalty methods for the generalized bilevel programming problem' *Mathematical Programming* 74 (1996) 141–157.

[12] J.V. Outrata and J. Zowe, 'A numerical approach to 1992. optimization problems with variational inequality constraints' *Mathematical Programming* 68 (1995) 105–130.

[13] M. Patriksson, 'Sensitivity analysis of traffic equilibria', to appear in *Transportation Science*.

[14] M. Patriksson and L. Wynter, 'Stochastic Mathematical Programs with Equilibrium Constraints' *Operations Research Letters*, 25 (1999) 159–167

[15] V. Visweswaran, C.A. Floudas, M.G. Ierapatritou, and E.N. Pistikopoulos, 'A Decomposition-based Global Optimization Approach for Solving Bilevel Linear and Quadratic Programs', *State of the Art in Global Optimization*, (Eds. C.A. Floudas and P.M. Pardalos), Kluwer Academic Publishers, (1996). http://citeseer.nj.nec.com/visweswaran96decompositionbased.html

[16] L. Wynter, 'Stochastic Bilevel Programs', *Encyclopedia of Optimization*, C. Floudas and P. Pardalos, Eds., Kluwer Academic Publishers, Dordrecht (2001).