Lower bounds for large scale multicommodity network design: a comparison between Volume and Bundle methods

Rui Sá Shibasaki
Universidade Federal de Minas Gerais
Av. Pres. Antônio Carlos, 6627 - Pampulha, Belo Horizonte - MG, 31270-901
ruishibasaki@ufmg.br

Mourad Baïou
Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes
Campus Universitaire des Cézeaux, 1 rue de la Chebarde, 63178 Aubière, França
baiou@isima.fr

Francisco Barahona
IBM Research, Thomas J. Watson Research Center
1101 Kitchawan Rd, Yorktown Heights, NY 10598, EUA
barahon@us.ibm.com

Philippe Mahey
Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes
Campus Universitaire des Cézeaux, 1 rue de la Chebarde, 63178 Aubière, França
philippe.mahey@isima.fr

Maurício Cardoso de Souza
Universidade Federal de Minas Gerais
Av. Pres. Antônio Carlos, 6627 - Pampulha, Belo Horizonte - MG, 31270-901
mauricio@dep.ufmg.br

ABSTRACT

Lagrangian relaxation has been proved to be a good alternative for solving linear relaxations when large scale problems are involved. In this paper, two non-differentiable optimization methods for solving the Lagrangian dual are compared: the Bundle method and the Volume algorithm. The Fixed-Charge Multicommodity Capacitated Network Design problem have been used for the comparison and the Volume algorithm seems to be preferable for the group of instances conceived, considering computing time, memory consumption and solution quality. Although the Bundle method produced good quality bounds for some instances, for many others it performed worse than the Volume algorithm. Moreover, the Bundle method takes more time per iteration, but it produces good bounds after few iterations.

KEYWORDS. Bundle methods. Volume Algorithm. Multicommodity Network Design.
Paper topics: Lagrangian Relaxation, Non-Differentiable Optimization

RESUMO

A relaxação lagrangiana provou ser uma boa alternativa para a solução de relaxação lineares, quando problemas de grande escala estão envolvidos. Neste trabalho, são comparados dois métodos de otimização não diferenciáveis, Bundle e Volume, para a resolução do dual lagrangiano. O problema de Design de Redes Multicommodity em grande escala foi testado e o algoritmo de Volume, no que diz respeito ao consumo de memória, tempo computacional e à qualidade da

solução, é preferível para as instâncias elaboradas. Embora o Bundle tenha fornecido bons limites, para muitos casos eles foram piores quando comparado com os do Volume. Em termos de tempo computacional, o Bundle mostrou ter iterações mais caras, mas consegue atingir bons limites em poucas iterações.

PALAVRAS CHAVE. Método de Feixes, Algoritmo de Volume, Desing de Redes Multicommodity.

Tópicos: Relaxação Lagrangiana, Otimização não diferenciável.

1. Introduction

Lagrangian Relaxation has been widely used to generate lower bounds for difficult constrained minimization problems and to serve as a basis to develop efficient approximation schemes, competing sometimes with the centralized exact approaches (see [Guignard, 2003] for the basic theory). As the resulting Lagrangian dual functions are generally non smooth but concave, the ability to lean on efficient subgradient algorithms is a crucial issue for the success of Lagrangian Relaxation. In this paper, we aim at comparing two classical versions of these algorithms, namely the Bundle method, early proposed in [Wolfe, 1975; Lemaréchal, 1989]) and the Volume algorithm proposed in [Barahona & Anbil, 2000]. Comparisons of non smooth optimization algorithms can be found in the literature (see [Frangioni, 2005; Briant et al., 2008]) but a direct comparison of these two algorithms applied to large-scale combinatorial models is missing and our work is an attempt to fill this gap. We have chosen to compare the performance of both algorithms on large-scale instances of the Fixed-Charge Multicommodity Capacitated Network Design (FCMC) problem because it presents many different characteristics which are favorable to our objectives, as the presence of different coupling constraints, potential candidates for the relaxation, the decomposable structure induced by these relaxations and the possibility to build very large instances, unreachable to most exact approaches but with relatively small duality gaps (see [Crainic et al., 2001]). Even if both algorithms have the ability to produce approximate, but fractional, primal solutions, we will not consider complementary techniques like Branch-and-Price or Lagrangian heuristics to solve the FCMC problem (see for instance [Gendron et al., 1999]).

The goal of this paper is to compare the Bundle and Volume methods in terms of computational time, memory consumption and quality of solutions, when dealing with the Lagrangian relaxation of a network design problem. The next sections will present the FCMC model followed by an explanation about the algorithms. Then, in Section 6, the computation experiments are detailed and the results are shown. Finally, conclusions are presented and future work is discussed.

2. The fixed charge multicommodity network design problem

The FCMC Problem consists in minimizing the total cost of multicommodity transport between pairs of origin-destination, so that the demand is satisfied and the capacity is respected. The objective function includes transportation costs for each commodity and arc installation costs, the latter being associated with a single facility of given capacity. Many additional features should be added to model real life network design problems, like the ones faced in Telecommunications or Transportation networks, but the model is sufficiently challenging and well adapted to our current purpose.

In this paper, it is considered for a given directed graph $G = (N, A)$, N being the set of nodes and A the set of arcs, the problem of minimizing the total cost to satisfy the demands d^k of a set K of origin-destination pairs, while the arc capacity u_{ij} are respected. The total cost is represented by the sum of transportation cost plus the arc usage cost. The variable cost for the commodity k in the arc (i, j) is called $c^k_{ij} \geq 0$ and the fixed charge for each arc (i, j) is $f_{ij} \geq 0$. A single origin $O(k)$ and destination $D(k)$ are associated with each commodity k. Introducing the variables x^k_{ij} for the flow quantity of k on the arc (i, j) and binary variables y_{ij} for the arc use

($y_{ij} = 1$ if the arc is installed) or ($y_{ij} = 0$ else), the model is presented as follows [Magnanti & Wong, 1984]:

$$\text{Minimize} \quad \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij}$$

$$\sum_{j\in N_i^+} x_{ij}^k - \sum_{j\in N_i^-} x_{ji}^k = \begin{cases} d^k, & \text{if} \quad i = O(k) \\ -d^k, & \text{if} \quad i = D(k) \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in N, k \in K \qquad (1)$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} y_{ij}, \qquad \forall (i,j) \in A \qquad (2)$$

$$x_{ij}^k \leq b_{ij}^k y_{ij} \qquad \forall (i,j) \in A, k \in K \qquad (3)$$

$$x_{ij}^k \geq 0, \qquad \forall (i,j) \in A, k \in K$$

$$y_{ij} \in \{0,1\}, \qquad \forall (i,j) \in A$$

where $N_i^+ = \{j \in N | (i,j) \in A\}$ is the set of nodes j having an arc arriving from to node i and $N_i^- = \{j \in N | (j,i) \in A\}$ the set of nodes j having an arc arriving into the node i. The transportation costs are considered equal for all commodities $k \in K$ and constants $b_{ij}^k = min\{u_{ij}, d^k\} \forall (i,j) \in A, k \in K$.

Constraints (1) guarantee the flow conservation in the network, then come the capacity constraints (2) and finally the domain of the variables. One can note that the strong forcing constraints (3) must be redundant for the mixed-integer program, but they increase considerably the quality of lower bound when solving the linear relaxation of the program [Chouman et al., 2003].

3. Lagrangian Relaxation and subgradient-like methods

The reason for using Lagrangian relaxation to obtain lower bounds for the problem mentioned is explained when large scale instances are involved. Resuming the main features of Lagrangian Relaxation, we start from a primal problem, supposed to be linear with mixed-integer variables, defined as :

$$\text{Minimize} \quad c.x \quad \text{s.t.} \quad Ax \leq b, x \in S$$

where $Ax \leq b$ represent the difficult constraints we want to relax. The set S may be discrete and defined by linear constraints. The continuous (or linear) relaxation bound is defined as $Z_L = \min_x c.x$ s.t. $Ax \leq b, x \in \text{Conv}(S)$, where $\text{Conv}(S)$ is the convex hull of the set S.

For a given vector of Lagrange multipliers $u \geq 0$ associated with the difficult constraints (assumed here to be inequalities), the Lagrangian subproblem defines a lower bound for the optimal value of the primal problem :

$$L(u) = \inf_{x\in S} (c - A^T u).x + b.u$$

The dual problem is thus to look for the best lower bound, i.e. to maximize the dual function L which is indeed concave on any convex subset of its domain (see [Lemaréchal, 1989] for example). That function is generally non smooth and piecewise affine (with a huge number of pieces, theoretically the number of extreme points of the polyhedral set $\text{Conv}(S)$). This motivates the search for efficient algorithms for non smooth optimization. These take profit of the fact that, for any solution $x(u)$ of the Lagrangian subproblem, a subgradient of L at u is easily computed, indeed $g(u) = Ax(u) - b \in \partial L(u)$.

3.1. Volume

The Volume algorithm presented in [Barahona & Anbil, 2000], tries to find an approximate solution to the master problem of the Dantzig-Wolfe decomposition, using subgradients.

Indeed, the Lagrangian dual problem can be formulated as (4) (which corresponds to the dual of the master problem in Dantzig-Wolfe decomposition, see [Lemaréchal, 1989]).

$$\begin{array}{ll} \text{Maximize} & Z \\ \text{s.t.} & Z \leq c.x^t + u.(b - Ax^t) \quad \forall t \\ & u \in \mathbb{R}^{n+}, Z \in \mathbb{R} \end{array} \tag{4}$$

The search for the optimal solution (u^*, Z^*), is based in a stability center $\bar{u}$, a step-size s_t and a subgradient-based direction. The stability center represents a point that have provided significant improvement in the optimization process. In its turn, the step-size represents how far one may move in the direction of $v^t = (b - A\bar{x})$, such as $u^t = \bar{u} + s_t \cdot v^t$. The directions are updated at each iteration according to the primal vector $\bar{x}$:

$$\bar{x} \leftarrow \alpha x^t + (1 - \alpha)\bar{x}$$

As stated in [Barahona & Anbil, 2000] at the end of an iteration (t), α, $(1-\alpha)\alpha$, $(1-\alpha)^2\alpha$, ... , $(1-\alpha)^t\alpha$ can serve as an approximation for the primal variables λ_1, ... , λ_t of the Dantzig-Wolfe's master problem, with respect to the dual constraints. Furthermore, those λ_i could be approximated by the volume below the active faces of (4), which explains the name of the method.

3.2. Bundle

Many Bundle algorithms have been proposed in the literature, but for this work the general one presented in [Crainic et al., 2001] was chosen. The main idea is to gather information throughout iterations in order to build a model for the Lagrangian dual, using the subgradient principles. It is expected that solving the model, the solution to the Lagrangian dual might be approximated.

Indeed, if g is a subgradient of the concave function L, then $L(u) \leq L(\bar{u}) + g.(u - \bar{u}) \quad \forall u \in \mathbb{R}^{n+}$ (extending the dual value with $-\infty$ if the Lagrangian subproblem is infeasible or unbounded). Assuming that there exists an initial Bundle $\beta = \{i \mid g_i \in \partial L(u_i)\}$, $\hat{L}(u)$ is the piecewise affine concave function such that :

$$L(u) \leq \hat{L}(u) := min\{L(u_i) + g_i.(u - u_i) : i \in \beta\} \quad \forall u \in \mathbb{R} \tag{5}$$

The model at this point is represented by a group of affine functions that together form an easier nondifferentiable optimization problem. The Moreau-Yosida regularization comes then as an alternative to this problem, since the function and its regularized function share the same minimum. Such regularization is defined by:

$$L_t(\bar{u}) = min_u \quad \hat{L}(u) + \frac{1}{2t}||u - \bar{u}||^2 \tag{6}$$

Assuming the Bundle has l parts, thanks to the information transfer property [Lemaréchal, 1989], it is convenient to rewrite the Bundle in terms of linearization errors regarding $\bar{u}$, such as $\tilde{e}_i := L(\bar{u}) - L(u_i) + g_i.(u_i - \bar{u}) \quad \forall i = 1, ..., l$. Then rewriting the Lagrangian dual as a regularized program, it turns into:

$$\begin{array}{ll} \text{Maximize} & Z + \frac{1}{2t}||u - \bar{u}||^2 \\ & Z \leq L(\bar{u}) + g_i(u - \bar{u}) - \tilde{e}_i \quad \forall i \in \beta \\ & u \in \mathbb{R}^{n+}, Z \geq 0 \end{array} \tag{7}$$

Further dualizing (7) with the dual coefficients $\alpha_i \geq 0$ we obtain :

$$\begin{aligned} \text{Minimize} \quad & -\frac{t}{2}||\sum_{i=1}^{l}\alpha_i g_i||^2 - \sum_{i=1}^{l}\alpha_i \tilde{e}_i + L(\bar{u}) \\ & \sum_{i=1}^{l}\alpha_i = 1 \\ & \alpha_i \geq 0 \qquad \forall i = 1, ..., l \in \beta \end{aligned} \tag{8}$$

Then the main search procedure is to get, at each iteration k, the solution α^k for (8) and set of new trial points along the direction of $\sum_{i\in\beta}\alpha_i g_i$ with a step of size t^k .

Bundle methods are now known to be very efficient for solving the Lagrangian dual problem, however, a great drawback is the fact that it demands the resolution of a quadratic subproblem at each iteration, which can decrease the algorithmic performance in a significant way. Frangioni, in [Frangioni, 1996], introduced a specially tailored algorithm to solve such quadratic programs (8) in a way to reduce the computational cost.

4. Review

The literature about Lagrangian relaxation and non-smooth optimization is extremely large. It embodies a range that goes from the way of conceiving the relaxed problem, until the methods with which Lagrangian duals are solved. In [Guignard, 2003] a few algorithms for it are described, and in [Crainic et al., 2001] different ways to relax FCMC are described.

Frangioni, in [Frangioni, 2002], presented a generalized Bundle method, which can be seen as similar to the Augmented Lagrangian Method [Bertsekas, 1996]. Still in that paper, a version for cases in which the Lagrangian dual can be decomposed is given. Furthermore, in [Frangioni & Gorgone, 2014] and [Frangioni & Gendron, 2013], the authors presented a version of the method that consider only some parts of the Lagrangian dual function to build the model, leaving the rest of it as its explicit form. In that last paper, a comparison with the Volume Algorithm is made and this partial decomposed Bundle have performed better than the Volume. The problem considered for this work is suitable for the three Bundle versions mentioned, however it is the one in the previous section that has been chosen to be tested. This is because we hope to be able to extend results for more problems where the Lagrangian dual cannot be decomposed.

According to [Barahona & Anbil, 2000], the Volume algorithm has similarities with the subgradient and Bundle methods. Regarding their proximities, [Bahiense et al., 2002] revised the Volume and managed to obtain an algorithm halfway in between the original and the Bundle one. Moreover, the results for the rectilinear Steiner problems showed that the new version could be competitive.

Some authors also focused their efforts on comparing some of the algorithms for non-differentiable optimization. This type of work was done in [Briant et al., 2008] where the authors compare different algorithms including Bundle, column generation and the Volume for five different problems. With respect to the Volume-Bundle comparison, the results have showed that in general they behave similar but Bundle enjoys more reliable stopping criteria, even thought it may be fairly expensive to reach it. According to the paper, the Bundle reaches better bounds with less iterations, thought we believe that its average time per iteration is fairly more expensive than Volume one. Considering that, this article based the comparison rather in computing time than in number of iterations.

In addition, [Escudero et al., 2012] and [Haouari et al., 2008] also have made comparisons. The first one tested the performance of the Volume, a variant of Cutting-Plane method

and other two algorithms for a stochastic problem and conclude that the volume provided stronger bounds in less time. The second paper worked with the prize collecting Steiner tree problem and put in test multiple variants of deflected subgradient strategies, the Volume Algorithm and a generalized cutting plane technique, finally concluding that the Volume Algorithm is outperformed by the different deflected subgradient algorithms.

5. Lagrangian Dual

The chosen approach for relaxing the Fixed-Charge Multicommodity Capacitated Network Design problem is made through the relaxation of flow-conservation constraints. The Lagrangian Dual corresponds to the maximization of $L(v)$, such that v is the vector of the Lagrangian coefficients $v_i^k \in \mathbb{R}, \forall i \in N, k \in K$ corresponding to the relaxed constraints. Such relaxation enables the subproblem to be decomposed in $|A|$ smaller knapsack subproblems $g_{ij}(v)$. To solve it one can easily verify the reduced costs $rc_{ij} = f_{ij} + g_{ij}(v)$ for each arc $(i,j) \in A$:

$$L(v) := Min \sum_{(i,j)\in A} [f_{ij} + g_{ij}(v)]y_{ij} + \sum_{k\in K} d^k(v^k_{D(k)} - v^k_{O(k)})$$

$$y_{ij} \in \{0,1\}, \qquad \forall (i,j) \in A$$

Then, for each $(i,j) \in A$ there is a continuous knapsack problem $g_{ij}(v)$, very simple to be solved. It suffices to fill up the arc with the commodities having the most negative reduced costs if any, until the arc flow equals the capacity.

$$g_{ij}(v) = Min \sum_{k\in K} (c^k_{ij} + v^k_i - v^k_j)x^k_{ij}$$

$$\sum_{k\in K} x^k_{ij} \leq u_{ij}$$

$$0 \leq x^k_{ij} \leq d^k, \qquad \forall k \in K$$

6. Computational Experiments

To solve the Lagrangian dual the two algorithms were implemented in C++, compiled with Apple LLVM version 6.1.0 (clang-602.0.53) and ran with 1,3 GHz Intel Core i5 in a Macbook 8 GB 1600 MHz DDR3. The linear relaxation to the problem was implemented in order to have some reference values. The linear program was solved by CPLEX 12.6.0.0, written in C++ and compiled with a g++ 5.4.0, using a 8GB Linux machine, Intel Core i7-2600 3.40GHz. The Volume Algorithm implementation has been provided by the COIN-OR project `https://projects.coin-or.org/Vol` and the Bundle implementation, by Antonio Frangioni, [Frangioni, 2013].

6.1. Instances

Instances were elaborated using the generator *Mulgen* implemented by Crainic, Frangioni and Gendron, in `http://www.di.unipi.it/optimize/Data/MMCF.html`. Their instance generator has a number $|N|$ of nodes, a number $|A|$ of arcs and a number $|K|$ of commodities as parameters. Two nodes are randomly connected until the number of arcs is achieved, with parallel arcs not allowed. A similar procedure is adopted for the commodities.

Costs, capacities and demands are uniformly distributed inside an interval given also as parameter. However, costs and capacities are recomputed in order to obtain different difficulty levels among the instances. Two ratios are used to do so: one for the capacities (C) and another (F) for fixed charges. Given that $T = \sum_{k\in K} d^k$:

$$C = |A|T / \sum_{(i,j)\in A} u_{ij}$$

$$F = |K| \sum_{(i,j)\in A} f_{ij} / (T \sum_{k\in K} \sum_{(i,j)\in A} c_{ij}^k),$$

In general, when C is close to 1, the network is lightly capacitated and becomes more congested as C increases. When F is close to zero, fixed costs are not relevant if compared with transportation costs. Their values increase as F increases as well.

Five groups of instances were conceived with three different ratios, each with five randomly generated instances. So for example, Group A-0 has 15 instances including 100 nodes 1000 arcs and 2000 commodities, 5 of them with C-ratio = 8 and F-ratio = 10 and so on. Table 1 describes all the classes of instances generated. The goal has been to test large scale instances with different levels of difficulty. The higher the ratios, the more the problem tends to be difficult due to the large importance of fixed charges and great tightness.

Group	Nodes	Arcs	Commodities	C ratio	F ration
CLASS A-0	100	1000	2000	8 10 14	10 10 10
CLASS B-0	100	1000	500	6 10 14	10 10 10
CLASS C-0	100	1000	800	2 14 2	0.001 0.001 10
CLASS D-0	100	1200	1000	14 20 1	12 0.001 20
CLASS E-0	100	2000	2000	1 20 1	20 0.001 0.001

Table 1: Instances

6.2. Calibration

In the interest of setting the best compromise between parameters of both methods, a calibration phase has been done. In this section, the best set of parameters found for each method is presented. Two additional stopping criteria have been set to both methods: a iteration limit of 1000 and a time limit of 3600 seconds.

Concerning the Volume Algorithm, there is a factor for the step-size that enlarges or decreases it. In order to do so, after 10 consecutive red iterations the factor is decreased and after 4 yellow iterations and 1 green iteration such factor is increased. Its initial value was set to 0.1. The value of α in its turn, is manipulated in a more delicate way, since its role is essential to the algorithm. For an initial $\alpha_{init} = 1$ the method reduces it, in order to enhance the precision of the primal solution ($\alpha_{init} = 0.1$ also work well). The decrease is made by multiplying α by a factor set to 0.3, when the $\bar{z}$ has not improved at least 1%, after 10 iterations. A lower bound set to 0.01 allows the algorithm to stop decreasing α in case it is necessary. The stopping criteria concerning a gap precision have been set to $1e-4$.

Likewise, the Bundle implemented has also a considerable number of parameters, although it appears to be a more robust method with respect to parameter settings. Basically, two strategies are involved when setting Bundle parameters: the Bundle-strategy and the t-strategy. Almost all parameters have been set according to [Crainic et al., 2001].

Concerning the Bundle strategy, the size has been set to 10 items and for every 20 iterations one item is discarded and a new one is included. In terms of t-strategy, it has a similar procedure of increasing and decreasing the value of t. It has been stablished three different approches to update such parameter and the one chosen is the Hard-Longterm t-Strategy. An initial t had to be chosen and depending on the instance, 1 and 10 were the most suitable values for it.

Two parameters, *tStar* and *EpsLin*, are employed as stopping criteria, so that for an iteration k, if *tStar* $* ||\hat{g}^k||^2 + \hat{e}^k \leq$ *EspLin* $* |L(\bar{u})|$ the algorithm stops. The *tStar* is an estimate of the largest step to move from a solution to another, which represents an estimate of improvement that can be obtained moving one step in the direction of any subgradient. In its turn, *EspLin* is a relative precision required [Frangioni, 2013] and it has been also set to $1e-4$. Still according to [Crainic et al., 2001], an interesting value for *tStar* must have one degree of magnitude greater than the initial value of t.

6.3. Results

In order to verify the validity of instances and methods, the linear relaxation have been solved by the simplex-based Network Optimizer implemented by *Cplex*, with a time limit of 3600 seconds. It has been observed that for some instances the Simplex method provided very poor bounds. The comparison has been made in terms of solution quality and time and memory consuming.

The marks (*) represent the best lower bound among the ones provided by each method, therefore gaps are computed with respect to that best lower bound. Table 2 presents the average gaps for each group of instances of three classes tested. Since for every group there are five instances, the mark (*) means that for all five instances the method has given the best bound. For classes D-0 and E-0 one can observe that the same does not occurs (see Table 4).

In terms of problem difficulty, one can verify that the more ratios are high the more the problem tends to be difficult. According to the results in Table 2 when the fixed-charge is not that relevant (F-ratio = .001) a simplex-based method might easily deal with the linear relaxation, depending on the size of the instance. Furthermore, the Bundle method seems to deal better with such instances, returning better bounds than the Volume ones, for those size of problems.

Instance		Volume		Bundle		Linear Relaxation	
		gap(%)	time	gap(%)	time	gap(%)	time
Class A-0	8_10	*	482	0.29	472	5.83	3623
	10_10	*	397	0.31	441	6.35	3620
	14_10	*	441	0.29	507	8.34	3624
Class B-0	14_10	*	124	2.19	132	10.58	3605
	10_10	*	122	2.53	134	5.60	3605
	6_10	*	125	3.00	128	2.06	3606
Class C-0	2_10	*	189	1.58	206	0.63	3610
	2_001	0.04	146	0.02	152	*	187
	14_001	0.05	150	0.03	152	*	381

Table 2: Results for 1000 iterations classes: A, B and C

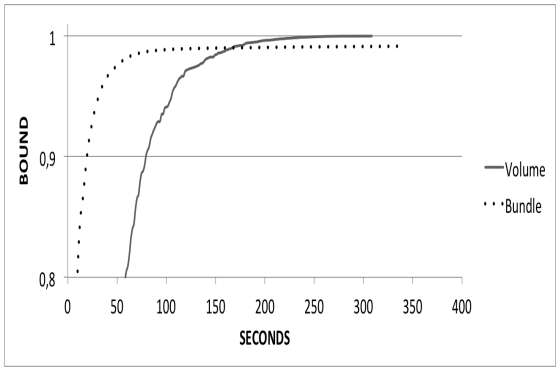
Contrary to A, B and C, classes D and E do not behave uniformly. Table 4 describes the results individually for each instance in those classes. Once again regarding class D-0, except for instance 20_001e, the Bundle method seems to perform better when dealing with low values of F-ratio (smaller values of fixed-charge). Nevertheless, as the number of arcs and demands grows the Volume algorithm manages to provide bounds close to the ones of Bundle or even better (see Class E-0 in Table 4). The capacity ratio (C-ratio), in its turn, do not appear to have a significant role in the performance of the methods.

Considering large scale instances, in general when the objective function depends mostly in the design variables (fixed-charge), the Volume algorithm reaches better bounds in less time than Bundle, until 1000 iterations. Such a running time difference can be explained by the fact that each iteration of Bundle algorithm demands the solution of a quadratic program, which can be more expensive in terms of time consumption. When the instance size increases from class D to E, it can be observed that the time per iteration can become a bottleneck for the method (Table 4).

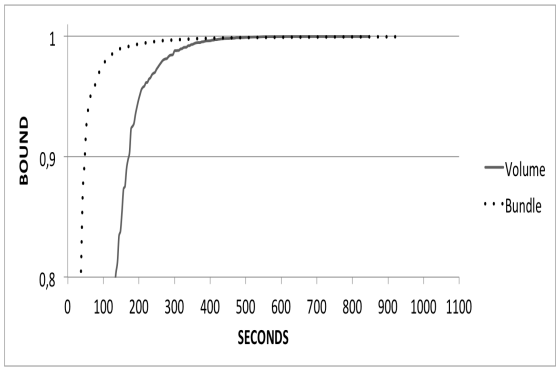
With respect to memory expenses, Table 3 shows the average amount of memory in gigabytes spent by each method to process each group of instances. As expected, Bundle needs in general more space in memory, since more "information" need to be gathered in the Bundle during the optimization process. Moreover, that need grows as the difficulty increases.

Figure 1 presents the average bound progression of both methods throughout the computation time. Such progression is computed with respect to the best bound given by one of the three methods (marks (*) in Tables 2 and 4). As expected from the Tables 2 and 4, both algorithms converge to almost the same bounds when fixed-charge ratios are low (Figures 1b, 1e and 1f), while for high values of F-ratio the Volume bounds are visibly greater. Furthermore, for all classes Volume curves have reached 100% of the best bound, or close to it.

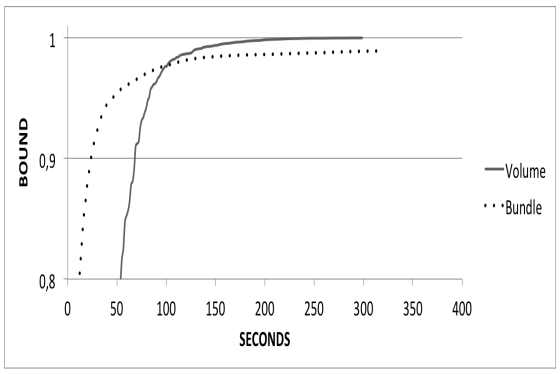
As one can see in Figures 1e, 1f and 1b, even though Bundle have expensive iterations, it could provide good quality bounds in the beginning of the optimization process, taking few iterations to do it.



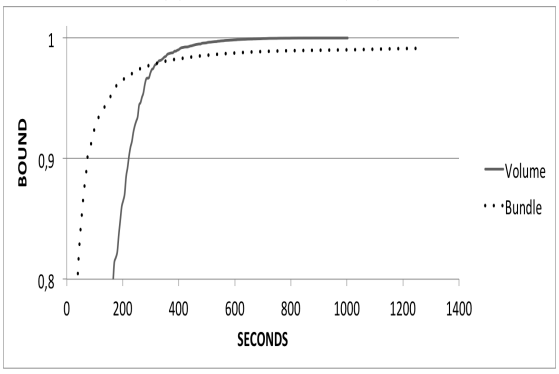
(a) Class D, 14_12 (a-e)



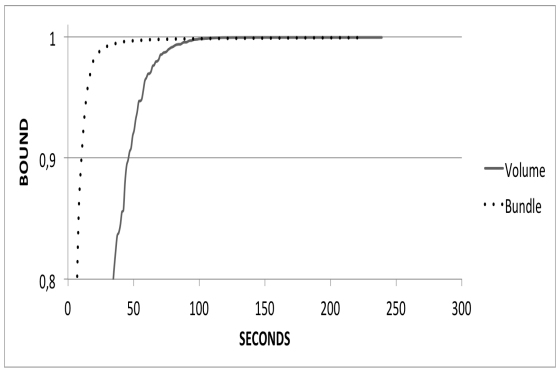
(b) Class E, 1_001 (a-e)



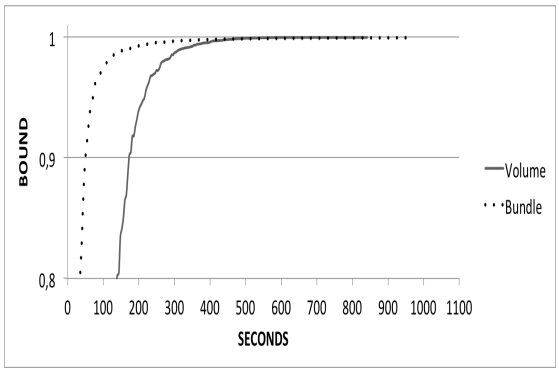
(c) Class D, 1_20 (a-e)



(d) Class E, 1_20 (a-e)



(e) Class D, 20_001 (a-e)



(f) Class E, 20_001 (a-e)

Figure 1: Average bound progression for large instances with respect to computation time

Instance		Volume RAM Gb	Bundle RAM Gb
Class A-0	8_10	3.5	4.7
	10_10	3.5	5.0
	14_10	3.7	4.7
Class B-0	14_10	3.3	5.4
	10_10	3.2	5.3
	6_10	3.0	5.1
Class C-0	2_10	3.7	5.1
	2_001	0.1	0.2
	14_001	0.1	0.2
Class D-0	14_12	4.6	5.0
	20_001	0.2	0.3
	1_20	3.4	4.7
Class E-0	1_001	0.4	0.8
	20_001	0.4	0.8
	1_20	4.9	5.1

Table 3: Average RAM consuming

Instance		Volume		Bundle		Linear Relaxation	
		gap(%)	time	gap(%)	time	gap(%)	time
Class D-0	14_12a	*	307	0.80	327	8.77	3603
	14_12b	*	299	0.81	329	7.44	3603
	14_12c	*	316	0.90	350	5.23	3603
	14_12d	*	297	0.68	337	7.89	3603
	14_12e	*	325	1.00	334	8.22	3604
	20_001a	0.05	248	0.04	242	*	381
	20_001b	0.05	244	0.03	211	*	189
	20_001c	0.04	234	0.01	222	*	38
	20_001d	0.08	233	0.05	233	*	468
	20_001e	0.05	237	0.26	206	*	672
	1_20a	*	313	1.37	329	0.00	3603
	1_20b	*	309	1.47	340	0.01	3603
	1_20c	0.09	306	1.43	342	*	3603
	1_20d	*	307	1.04	310	0.18	3604
	1_20e	*	259	0.13	259	4.57	3603
Class E-0	1_001a	0.01	840	0.00	949	*	28
	1_001b	0.02	849	0.10	901	*	3610
	1_001c	*	842	0.06	960	0.06	3610
	1_001d	0.01	849	*	960	1.20	3610
	1_001e	0.04	848	*	947	0.46	3610
	20_001a	0.01	841	0.00	954	*	30
	20_001b	*	848	0.13	968	0.90	3609
	20_001c	*	850	0.05	968	0.06	3610
	20_001d	*	844	0.00	909	1.05	3610
	20_001e	0.04	828	*	961	0.49	3611
	1_20a	*	972	0.96	1336	0.73	3611
	1_20b	*	978	0.68	1312	0.35	3610
	1_20c	*	1064	1.19	1312	0.61	3609
	1_20d	*	1049	1.12	1282	0.64	3612
	1_20e	*	955	0.35	1105	0,11	3610

Table 4: Results for 1000 iterations classes: D and E

7. Conclusions

Lagrangian relaxation has proved to be a good alternative to deal with linear relaxations of large scale problems. Indeed for some instances the simplex-based optimizer has not given the

best bounds within one hour of computation time.

The Bundle and the Volume algorithms both have provided good quality bounds, however both methods appear to struggle to stop with reliable stopping criteria. In [Briant et al., 2008], results showed that the Bundle method enjoys good accuracy for the Cutting Stock problem but it may be fairly expensive to reach it, as well as in some large instances of the Travelling Salesman problem. Considering smaller instances of FCMC, [Frangioni & Gorgone, 2014] presented results showing that the Bundle provided better bounds, however with much higher computation time (It is not clear which stopping criterion was set to the Volume algorithm). In the present work, both methods have reached great bounds but they have run until the limit of iterations, not being able to converge with the given accuracy ($1e-4$).

With respect to the comparison made in this paper, for almost all small instances Volume-times and Bundle-times are very close, but when the size of instances is enlarged from Class D to Class E, it becomes evident the advantages of the Volume algorithm. Regarding memory consumption, the Volume algorithm performed better for all instances. Moreover, Volume algorithm provided better bounds for all instances with high levels of F-ratio. For those with low levels of F-ratio (0.001), *Cplex* provided the best bounds for Classes C(2_001) , C(14_001) and D(20_001). For Classes E(20_001) and E(1_001), even with low values of F-ratio, Volume provided the best bounds for half of the instances in those classes.

One can say that for the tests put in practice the Volume algorithm has performed well no matter the instance characteristics, in general if we count the number of best bounds (*) in the tables, Volume presents 25 and Bundle, only 3. In addition, Bundle performed worse for those with very large values of fixed charge and small values of transportation costs. Other types of design problems may be tested so one can verify if such features can be generalized.

Roughly, Volume algorithm demands less time per iteration and less memory to run, providing bounds as good as the Bundle ones, or even better. However, the Bundle method is able to provide good quality bounds in very few iterations, which can be very useful depending on the application.

Since the Bundle time-consuming per iteration might be a bottleneck for its performance, future work aims to test even larger instances, also considering the traditional subgradient method for comparison. Still, one could also include other problems like set partition (see [Boschetti et al., 2008] for example).

It is important to keep in mind that there are other versions of the same Bundle method, such as the decomposable one and the partial one. Moreover, different ways of relaxing the Multicommodity Network Design Problem are possible, which make it not advisable to generalize the results obtained is this paper. More research has to be done, to verify the performances of the Bundle method under these other circumstances.

8. Acknowledgements

We would like to thank Antonio Frangioni for providing his Bundle implementation. Moreover, we would like to thank FAPEMIG for the financial support.

References

Bahiense, L., Maculan, N., and Sagastizábal, C. (2002). The volume algorithm revisited: relation with bundle methods. *Mathematical Programming*, 94(1):44–69.

Barahona, F. and Anbil, R. (2000). The volume algorithm: producing primal solutions with a subgradient method. *Mathematical Programming*, 87:385–399.

Bertsekas, D. P. (1996). *Constrained optimization and Lagrange multiplier methods*. Athena Scientific.

Boschetti, M. A., Mingozzi, A., and Ricciardelli, S. (2008). A dual ascent procedure for the set partitionning problem. *Discrete Optimization*, 5:735–747.

Briant, O., Lemaréchal, C., Meurdesoif, P., Michel, S., Perrot, N., and Vanderbeck, F. (2008). Comparison of bundle and classical column generation. *Mathematical Programming*, 113(2): 299–344.

Chouman, M., Crainic, T. G., and Gendron, B. (2003). A cutting-plane algorithm based on cut-set inequalities for multicommodity capacitated fixed charge network design. Technical report, Centre de recherche sur les transports, Université de Montréal.

Crainic, T. G., Frangioni, A., and Gendron, B. (2001). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112(1-3): 73–99.

Escudero, L. F., Garín, M. A., Pérez, G., and Unzueta, A. (2012). Lagrangian decomposition for large-scale two-stage stochastic mixed 0-1 problems. *TOP*, 20(2):347–374.

Frangioni, A. (1996). Solving semidefinite quadratic problems within nonsmooth optimization algorithms. *Computers Ops. Res.*, 23:1099–1118.

Frangioni, A. (2002). Generalized bundle methods. *SIAM J. Optim*, 13:117–156.

Frangioni, A. (2005). About lagrangian methods in integer optimization. *Annals of Operations Research*, 139(1):163–193.

Frangioni, A. *The NDOSOlver + FiOracle Project*, 2013.

Frangioni, A. and Gendron, B. (2013). A stabilized structured dantzig-wolfe decomposition method. *Math. Program., Ser. B*, 140(1):45–76.

Frangioni, A. and Gorgone, E. (2014). Bundle methods for sum-functions with ”easy”components: applications to multicommodity network design. *Math. Program., Ser. A*, 145(1):133–161.

Gendron, B., Crainic, T., and Frangioni, A. (1999). Multicommodity capacitated network design. In *Telecommunications Network Planning*. Kluwer Academics.

Guignard, M. (2003). Lagrangean relaxation. *TOP*, 11(2):151–200.

Haouari, M., Layeb, S. B., and Sherali, H. D. (2008). The prize collecting steiner tree problem: models and lagrangian dual optimization approached. *Computational Optimization and Applications*, 40(1):13–39.

Lemaréchal, C. (1989). Nondifferentiable optimization. In *Handbooks in OR and MS*. Elsevier Science Publishers.

Magnanti, T. L. and Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55.

Wolfe, P. (1975). A method of conjugate subgradients for minimizing nondifferentiable functions. In *Nondifferentiable Optimization*. Springer Berlin Heidelberg.