A Column Generation Approach to Capacitated p-Median Problems

Luiz A. N. Lorena(*)
lorena@lac.inpe.br
FAX: +55(12)39456553
LAC - Laboratório Associado de Computação e Matemática Aplicada
INPE – Instituto Nacional de Pesquisas Espaciais
Caixa Postal 515
12245-970 São José dos Campos, SP – Brazil

Edson L. F. Senne
elfsenne@feg.unesp.br
FEG/UNESP - Universidade Estadual Paulista
Faculdade de Engenharia - Departamento de Matemática
Caixa Postal 205
12515-410 Guaratinguetá, SP – Brazil

Scope and purpose

The location of facilities is a central problem for strategic decisions. Many applications have been explored and the resulting mathematical problems are considered by heuristics and exact methods. We revive in this paper the application of column generation to a capacitated p-median location problem. There has been renewed interest in the column generation approach as it can be faster than nonlinear subgradient methods. However, in many cases a straightforward application of column generation may result in slow convergence. We explore in this paper an alternative to stabilize the column generation when applied to the location problem.

Abstract

The Capacitated p-median problem (CPMP) seeks to solve the optimal location of p facilities, considering distances and capacities for the service to be given by each median. In this paper we present a column generation approach to CPMP. The identified restricted master problem optimizes the covering of 1-median clusters satisfying the capacity constraints, and new columns are generated considering knapsack subproblems. The Lagrangean/surrogate relaxation has been used recently to accelerate subgradient like methods. In this work the Lagrangean/surrogate relaxation is directly identified from the master problem dual and provides new bounds and new productive columns through a modified knapsack subproblem. The overall column generation process is accelerated, even when multiple pricing is observed. Computational tests are presented using instances taken from real data from São José dos Campos' city.

Key words: Location problems, Capacitated p-median problems, Column generation, Lagrangean/surrogate relaxation.

(*) Corresponding author

1. Introduction

The search for *p-median* vertices on a network (graph) is a classical location problem. The objective is to locate *p* facilities (medians) so as to minimize the sum of the distances from each demand vertex to its nearest facility. The capacitated p-median problem (CPMP) considers capacities for the service to be given by each median. The total service demanded by vertices identified by p-median clusters cannot exceed their service capacity.

In general, the CPMP has not seem as intensively studied as the classical p-median problem. Related problems have appeared in Bramel and Simchi-Levi [3], Klein and Aronson [18], Mulvey and Beck [29] and Osman and Christofides [32]. An extensive bibliography of related problems, and also a set of test problems are presented in [32]. The CPMP is known to be NP-hard. Some earlier approaches applying Lagrangean heuristics to the CPMP are proposed in Koskosidis and Powell [19] and in [29]. Recent approaches apply metaheuristics, such as simulated annealing and tabu search (as in França, Sosa and Pureza [13] and in [32]), and genetic algorithms (Maniezzo, Mingozzi and Baldacci [25] and Lorena and Furtado [20]). Good results are reported for a set of standard test problems (OR-Library - http://mscmga.ms.ic.ac.uk/info.html [2]).

The Lagrangean/surrogate relaxation has been used recently to accelerate subgradient like methods, which are often used to optimize the corresponding Lagrangean dual problem as in Lorena and Lopes [21], Lorena and Narciso [22], Lorena and Senne [23, 24], Narciso and Lorena [30], and Senne and Lorena [34]. Lorena and Senne [24] explored the Lagrangean/surrogate relaxation combination with location-allocation heuristics, proposed by Cooper [5] and used before by Senne and Lorena [34].

Column generation is a powerful tool for solving large-scale linear programming problems. Such linear programming may arise when the columns in the problem are not known in advance and a complete enumeration of all columns is not an option, or the problem is rewritten using Dantzig-Wolfe decomposition (the columns correspond to all extreme points of a certain constraint set) [6]. It appears that this approach was not tried before for the CPMP, but has been successfully explored in several other applications, such as the well-known cutting-stock problem, vehicle routing and crew scheduling [1, 4, 6, 7, 8, 9, 14, 15, 35, 36]. However, in many cases a straightforward application of column generation may result in slow convergence. The Lagrangean/surrogate relaxation is showed in this paper to be an acceleration process to the column generation, generating new productive sets of columns.

In this paper we examine a column generation approach to the CPMP. The identified restricted master problem optimizes the covering of 1-median clusters satisfying a set of capacity constraints, and new columns are generated solving capacitated subproblems, which consider the restricted master dual variables and the clusters capacities. In this work the Lagrangean/surrogate relaxation is directly identified from the master problem dual and provides new bounds and new productive columns through a modified knapsack subproblem. The overall column generation process is accelerated, even when multiple pricing is observed. Computational tests are presented using instances taken from the literature.

The paper is organized as follows. Section two presents CPMP formulations used in this paper. The next section presents the Lagrangean/surrogate relaxation and the column generation approach to the CPMP, comparing the traditional process and the one improved by Lagrangean/surrogate multipliers. Section three presents the basic algorithms for the column generation, including the initial pool of columns and the management of columns. Section four provides computational results to illustrate the benefits of the new approach.

2. CPMP formulations

The CPMP considered in this paper is modeled in two ways. The first is the following binary integer-programming problem (P):

$$(P) \qquad v(P) = Min \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{j \in N} x_{ij} = 1 \ ; \ i \in N \tag{2}$$

$$\sum_{j \in N} x_{jj} = p \tag{3}$$

$$\sum_{i \in N} q_i x_{ij} \le Q x_{jj} \ ; \ j \in N \tag{4}$$

$$x_{ij} \in \{0,1\}; \ i \in N, \ j \in N \tag{5}$$

where:

$N = \{1,...,n\}$ is the index set of entities to allocate and also of possible medians, where p medians will be located;
q_i is the demand of each entity and Q the capacity of each possible median;
$[d_{ij}]_{n \times n}$ is a distance matrix;

$[x_{ij}]_{n\times n}$ is the allocation matrix, with $x_{ij}=1$ if entity i is allocated to median j, and $x_{ij}=0$, otherwise; $x_{jj} = 1$ if median j is selected and $x_{jj} = 0$, otherwise.
Constraints (2) and (3) impose that each entity is allocated to only one median. Constraint (4) imposes that a total median capacity must be respected, and (5) provides the integer conditions.

We assume equal capacities to simplify the alternative set covering formulation to be given in the sequel.

The CPMP problem can also be modeled as the following set partitioning problem with a cardinality constraint (SPP):

$$\text{(SPP)} \qquad v(SPP) = Min \sum_{k=1}^{m} c_k x_k$$

$$\text{subject to} \qquad \sum_{k=1}^{m} A_k x_k = 1 \qquad (6)$$

$$\sum_{k=1}^{m} x_k = p \qquad (7)$$

$$x_k \in \{0,1\},$$

where

$S = \{S_1, S_2, ..., S_m\}$, is a set of subsets of N;

A = $[a_{ik}]_{nxm}$, is a matrix with $a_{ik} = \begin{cases} 1, if\ i \in S_k \\ 0, otherwise \end{cases}$, satisfying $\sum_{i\in N} q_i a_{ik} \le Q$;

and $c_k = \underset{i\in S_k^1}{Min}\left(\sum_{j\in S_k^1} d_{ij}\right)$, considering $S_k^1 = \{i \in S_k | \, a_{ik} = 1\}$.

This is the formulation found in Minoux [28]. The same formulation can be obtained from the problem P by applying the Dantzig-Wolfe decomposition. For each subset S_k^1 , the open median is decided when the column cost ck is calculated, and so the columns of SPP implicitly consider the constraints set (4) in P. Constraints (1) and (2) are conserved and respectively updated to (6) and (7), according the Dantzig-Wolfe decomposition process [6].

If S is the set of all subsets of N, the formulation can give an optimal solution to the CPMP. However, the number of subsets may be huge, and a partial set of columns can be considered instead. The SPP defined above is also known as the restricted master problem in the column generation context [1].

3. Lagrangean/surrogate relaxation and Column generation

The equivalencies of Dantzig-Wolfe decomposition, column generation and Lagrangean relaxation optimization are well known. Solving a linear programming problem by Dantzig-Wolfe decomposition is the same as solving the Lagrangean by Kelley's cutting plane method [17]. However, in many cases a straightforward application of column generation may result in slow convergence [10, 11, 26, 31]. Below, the Lagrangean/surrogate will be related to the column generation, identifying the common optimization problems considered at the resolution process of the Lagrangean/surrogate and the reduced cost problems. The Lagrangean/surrogate is able to identify very good lower bounds and contributes with new columns that accelerate the column generation process.

For a given $\lambda \in R^n_+$ and $t \geq 0$, the Lagrangean/surrogate relaxation of CPMP is given by:

$$(L_tP^l) \qquad v(L_tP^l) = Min \sum_{i \in N} \sum_{j \in N} (d_{ij} - t\,l_i)x_{ij} + t \sum_{i \in N} l_i$$

subject to constraints (3), (4), and (5).

Problem L_tP_λ is solved considering implicitly constraint (3), and decomposing for index j obtaining the following n 0-1 knapsack problems:

$$v(knap^{t l})_j = \text{Min} \sum_{i \in N} (d_{ij} - t\, l_i) x_{ij} \tag{8}$$

subject to constraints (4) and (5).

Each problem is solved using Horowitz and Sahni code (see Martello and Toth [27]). Let J be the index set of the p smallest $v(knap^{t l})_j$, j $\in$ N (here constraint (3) is considered implicitly). The Lagrangean/surrogate value is given by:

$$v(L_tP^{l}) = \sum_{j \in J} v(knap^{t l})_j + t \sum_{i \in N} l_i \, . \tag{9}$$

The interesting feature of relaxation L_tP_λ is that, for t = 1, expression (9) is the usual Lagrangean relaxation with the multiplier λ. Two duals can be identified here, an external dual for the multidimensional variable λ, usually optimized by subgradient methods, and for a fixed multiplier λ, the best value designed for t can be found through an inner Lagrangean dual v(D_t^l) = $\underset{t \geq 0}{Max}$ $v(L_tP^{l})$. The best Lagrangean/surrogate relaxation value gives an improved bound to the usual Lagrangean relaxation and accelerates the overall optimization process. To find an approximated best Lagrangean/surrogate multiplier t , the dichotomous search procedure SH described in [33] is used.

Lorena and Senne [24] applied the Lagrangean/surrogate to solve the dual by standard subgradient methods. The dual solutions should be made primal feasible and local search heuristics are combined with the dual process. The quality of feasible solutions obtained is comparable to metaheuristic approaches, but employs small computational times on large-scale real data obtained using Geographic Information Systems software [12].

The problem to be solved by column generation is the linear programming set covering (SCP):

$$\begin{aligned} &\textit{(SCP)} && v(SCP) = Min \sum_{k=1}^{m} c_k x_k \\ &\text{subject to} && \sum_{k=1}^{m} A_k x_k \geq 1 && (10) \\ & && \sum_{k=1}^{m} x_k = p \\ & && x_k \in [0,1]. \end{aligned}$$

Observe that $d_{ij} \geq 0,\ \forall i,j$ and (7) can be replaced with (10) in the linear model, and the main advantage is that problem SCP is easier solvable than SPP.

After defining an initial pool of columns, problem SCP is solved and the final dual costs p_i, i = 1,...,n and μ are used to generate new columns by solving the following sub-problem (to simplify the notation consider for each j , $x_{ij} = z_i$):

$$\textit{(Sub}_p\textit{P)} \qquad v(Sub_p P) = \underset{j \in N}{Min} \{v(knap^p)_j\}.$$

Problem $Sub_\pi P$ is solved decomposing for index *j*, obtaining the n 0-1 knapsack problems of equation (8). The associated reduced cost is *v(Sub$_p$P)* - μ, and column $\begin{bmatrix} z_i \\ \hline 1 \end{bmatrix}$ is added to SCP if *v(Sub$_p$P)* < $|m|$. In practice, for *j = 1,...,n,* all the corresponding columns satisfying

$$\sum_{i \in N} q_i z_i \leq Q \text{ and } \sum_{i \in N} (d_{ij} - p_i) z_i < |m| \qquad (11)$$

can be added to the pool of columns, accelerating the column generation process.

Note that for $\lambda = \pi$, the same n knapsack problems of equation (8) come out in the solution of L_1P^{π} (for the Lagrangean case, where t = 1) and $Sub_{\pi}P$. Then, if the multipliers p_i (i = 1,...,n) of problem SCP are given to problem L_1P^{π}, the corresponding Lagrangean problem can be used to generate columns and also to give a lower bound to P and SCP.

The Lagrangean/surrogate is integrated to the column generation process transferring the multipliers p_i (i = 1,...,n) of problem SCP to the problem $\underset{t \geq 0}{Max}$ $v(L_tP^{\pi})$, and returning the corresponding columns. If the new columns continue to satisfy expression (11), then they can be added to the pool of candidate columns. The resulting effect is an acceleration of the convergence, even when multiple columns are added to the pool at each iteration of the process.

The Lagrangean/surrogate is a valid lower bound to the column generation process. In particular it is better than the Lagrangean bound (t = 1) and can be useful to define convergence settings to the restricted master. Figure 1 shows a typical behavior of dual bounds integrated to the column generation.

Figure 1. Typical computational behavior of dual bounds

4. The algorithms

The parameter t is the differential in the Lagrangean/surrogate relaxation. Varying this parameter can reproduce the usual Lagrangean relaxation (t is fixed to 1) and better Lagrangean bounds (t is identified using the dichotomous search procedure SH described in

[33]). The column generation algorithms are then identified by the use of parameter t , and are labeled by this parameter. The algorithms can be stated as:

<u>Algorithm CG (t)</u>

(i) Set an initial pool of columns to SCP;

(ii) Solve SCP using the CPLEX [16] and return the dual prices p_j , j = 1,...,n and μ;

(iii) Solve approximately (by dichotomous search) a local Lagrangean/surrogate dual $\underset{t \geq 0}{Max}\ v(L_t P^p)$, returning the identified columns;

(iv) Append to SCP the columns $\begin{bmatrix} z_i \\ \hline 1 \end{bmatrix}$ satisfying expression (11);

(v) If no columns are found in step (iv) or [$\underset{t \geq 0}{Max}\ v(L_t P^p)$ – v(SCP)] < 1 then stop;

(vi) Perform tests to remove columns and return to (ii).

Steps (i) and (vi) are described in the sequel. If t = 1, CG(1) gives the traditional column generation process: step (iii) calculates the usual Lagrangean bound $v(L_1 P^p)$. In any case the bounds v(SCP) and $v(L_t P^p)$ are calculated at each iteration.

The following sub-routine is used in step (i):

```
Let nc = 0 and C = { }.
While nc < NCOLS do:
    dtotal = 0;
    N = { 1, ..., n };
    While Q ≥ dtotal do:
        Let k a random value of N;
        C = C ∪ { k }
        N = N – { k }
        dtotal = dtotal + $q_k$
    End_while;
    Find the best median on cluster C;
    Add to SCP the column corresponding to cluster C;
    nc = nc + 1.
    End_while.
```

NCOLS is set to 1000 in computational tests described in the next section. To prevent infeasibilities, a high cost dummy column formed of ones is also included on the initial set. In order to remove columns we have conserved in the process only the 3000 columns presenting the smaller reduced costs.

5. Computational experiments

The algorithms described in section 4 are coded in C and the computational tests were made on a Sun Ultra30 workstation.

The set of instances comprising real data were collected using the Geographical Information System ArcView (ESRI [12]), and report the central area of São José dos Campos city. Six instances (100x10), (200x15), (300x25), (300x30), (402x30) and (402x40) are created, containing 100, 200, 300 and 402 nodes. Each point is located on a block, which presents a demand node and is also a possible place to locate medians. Demand was estimated considering the number of houses (apartments) at each block. An empty block received value 1. Capacities are then estimated as $C = \left\lceil \frac{\sum \text{demands}}{\text{number of medians} \times \alpha} \right\rceil$, where α was set equal to 0.9 or 0.8. These instances are available at http://www.lac.inpe.br/~lorena/instancias.html, and Figures 2 and 3 show, respectively, the set of points for a instance of 100 nodes and the corresponding solution for 10 medians.

Figure 2: Instance of 100 nodes (central area of São José dos Campos)

Figure 3: Instance of 100 nodes (solution for 10 medians)

Tables 1 and 2 report the computational results obtained by the CG(t) and CG(1) algorithms. The primal-dual gaps compare the best known feasible solutions of (P), obtained using the location-allocation heuristics reported in [24] (see also Table 3), and the respective dual and linear programming bounds ($v(LtP\pi)$ and $v(SCP)$).

Table 1 – Results for CG(t)

Table 2 – Results for CG(1)

Observe from Tables 1 and 2 that algorithm CG(t) is faster and able to generate fewer productive columns than CG(1). This result can be very interesting for large scale instances.

These results are even better when compared with the Lagrangean/surrogate relaxation associated to a subgradient method. Table 3 reports the results obtained by the LSLSH(t) heuristic, described in [24], to the same set of instances. Heuristic LSLSH(t) is a Lagrangean/surrogate heuristic combined with a traditional subgradient method and location-allocation primal heuristics.

Table 3 – Results for LSLSH(t)

It is interesting to investigate whether the columns generated by CG(t) are productive when the number of columns is limited at the master problem. Table 4 presents the results for the time-consuming instance sjc4a. The number of columns at the master problem is set to values on the interval [2800, 3500].

Table 4 – Restricting the number of columns at the master problem

Note that as the number of fixed columns decreases the CG(t) algorithm remains operational finding the respective solutions of Table 1. Algorithm CG(1) presented a degraded effect when the number of columns decreases, especially for 2900 and 2800 columns. The gaps increase and the results of Table 2 are not reproduced. All the computational tests reported have made with an iteration limit of 300.

The computational tests proceeded with a large-scale instance. The Pcb3038 instance in the TSPLIB, compiled by Reinelt [33], was considered for the tests. The capacities were estimated as $C = \left\lceil \frac{\sum \text{demands}}{\text{number of medians} \times 0.8} \right\rceil$ and the number of columns at the master problem was fixed to 20000. These instances are also available at http://www.lac.inpe.br/~lorena/instancias.html. The results presented in Tables 5 and 6 confirm that CG(t) is able to generate better quality columns than CG(1). We can also observe that when the number of required medians decreases the problems are more difficult and time consuming. The CG(t) expends almost half the computational time of CG(1), which seems to be an important consideration for such large scale instances.

Table 5. Computational results for CG(t) on Pcb3038 instances

Table 6. Computational results for CG(1) on Pcb3038 instances

6. Conclusions

This paper presented column generation approaches for a CPMP. The approaches integrate the traditional column generation to the Lagrangean/surrogate relaxation context, identifying new productive columns and accelerating the computational process.

The computational results show that the Lagrangean/surrogate sub-problem generates a small number of productive columns and the restricted master is also manageable with a small number of columns.

The Lagrangean/surrogate lower bounds can be useful to branch-and-price trees and are currently being explored in this context.

Acknowledgments: The authors acknowledge Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP (proc. 99/06954-7) and Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (processes 302408/88-6 and 300837/89-5) for partial financial research support. We also acknowledge the very useful suggestions and comments of two anonymous referees.

References

1. Barnhart, C.; Johnson, E.L.; Nemhauser, G.L.; Savelsbergh, M.W.P.; Vance, P.H. Branch-and-Price: Column Generation for Solving Huge Integer Programs. Operations Research 1998; 46: 316-329.
2. Beasley, J.E. Or-library: Distributing test problems by electronic mail. Journal Operational Research Society 1990; 41: 1069-1072.
3. Bramel, J,; Simchi-Levi, D. A location based heuristic for general routing problems. Operations Research 1995; 43: 649-660.

4. Carvalho, J.M.V. Exact Solution of Bin-Packing Problems Using Column Generation and Branch-and-Bound, Universidade do Minho, Departamento Produção e Sistemas, Working Paper, 1996.
5. Cooper, L. Location-allocation problems. Operations Research 1963; 11: 331-343.
6. Dantzig, G.B.; Wolfe, P. Decomposition principle for linear programs. Operations Research 1960; 8: 101-111.
7. Day, P.R.; Ryan, D.M. Flight Attendant Rostering for Short-Haul Airline Operations. Operations Research 1997; 45: 649-661.
8. Desrochers, M.; Desrosiers, J.; Solomon, M. A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows. Operations Research 1992; 40: 342-354.
9. Desrochers, M.; Soumis, F. A Column Generation Approach to the Urban Transit Crew Scheduling Problem. Transportation Science 1989; 23: 1-13.
10. du Merle, O.; Goffin, J.L.; Vial, J.P. On Improvements to the Analytic Centre Cutting Plane Method. Computational Optimization and Applications 1998; 11: 37-52.
11. du Merle, O.; Villeneuve, D.; Desrosiers, J.; Hansen, P. Stabilized column generation. Discrete Mathematics 1999; 194: 229-237.
12. ESRI Environmental Systems Research Institute, Inc. Avenue Customization and Application Development for ArcView, 1996.
13. França, P.M.; Sosa, N.M.; Pureza, V. An adaptive tabu search algorithm for the capacitated p-median problem. International Transactions in Operations Research 1999; 6: 665-678.
14. Gilmore, P.C.; Gomory, R.E. A linear programming approach to the cutting stock problem. Operations Research 1961; 9: 849-859.
15. Gilmore, P.C.; Gomory, R.E. A linear programming approach to the cutting stock problem - part ii. Operations Research 1963; 11: 863-888.
16. ILOG CPLEX 6.5. ILOG Inc. Cplex Division, 1999.
17. Kelley, J.E. The Cutting Plane Method for Solving Convex Programs. Journal of the SIAM 1960; 8: 703-712.
18. Klein, K.; Aronson, J.E. Optimal clustering: a model and method. Naval Research Logistics 1991; 38, 447-461.
19. Koskosidis, Y.A.; Powell, W.R. Clustering algorithms for consolidation of costumes orders into vehicle shipments. Transportation Research B 1992; 26: 365-379.
20. Lorena, L. A. N.; Furtado, J. C. Constructive Genetic Algorithm for Clustering Problems. Evolutionary Computation 2001; 9(3): 309-327.
21. Lorena, L.A.N.; Lopes, F.B. A surrogate heuristic for set covering problems. European Journal of Operational Research 1994; 79: 138-150.
22. Lorena, L.A.N.; Narciso, M.G. Relaxation heuristics for a generalized assignment problem. European Journal of Operational Research 1996; 91: 600-610.
23. Lorena, L.A.N.; Senne, E.L.F. Improving traditional subgradient scheme for Lagrangean relaxation: an application to location problems. International Journal of Mathematical Algorithms 1999; 1: 133-151.
24. Lorena, L.A.N.; Senne, E.L.F. Local search heuristics for capacitated p-median problems. Networks and Spatial Economics 2002; to appear.
25. Maniezzo, V.; Mingozzi, A.; Baldaci, R. A bionomic approach to the capacitated p-median problem. Journal of Heuristics 1998; 4: 263-280.
26. Marsten, R.M.; Hogan, W.; Blankenship, J. The Boxstep method for large-scale optimization. Operations Research 1975; 23: 389-405.
27. Martello, S.; Toth, P. Knapsack problems: Algorithms and computer implementations, John Wiley & Sons, 1990.
28. Minoux, M. A Class of Combinatorial Problems with Polynomially Solvable Large Scale Set Covering/Set Partitioning Relaxations. RAIRO 1987; 21 (2): 105–136.

29. Mulvey, J.M.; Beck, M.P. Solving capacitated clustering problems. European Journal of Operational Research 1984; 18: 339-348.
30. Narciso, M.G.; Lorena, L.A.N. Lagrangean/surrogate relaxation for generalized assignment problems. European Journal of Operational Research 1999; 114: 165-177.
31. Neame, P.J. Nonsmooth Dual Methods in Integer Programming Phd Thesis - Department of Mathematics and Statistics, The University of Melbourne, 1999.
32. Osman, I.H.; Christofides, N. Capacitated clustering problems by hybrid simulated annealing and tabu search. International Transactions in Operational Research 1994; 1: 317-336.
33. Reinelt, G. The traveling salesman problem: computational solutions for TSP applications. Lecture Notes in Computer Science 840, Springer Verlag, Berlin, 1994.
34. Senne, E.L.F.; Lorena, L.A.N. Lagrangean/Surrogate Heuristics for p-Median Problems. In Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research, M. Laguna and J.L. Gonzalez-Velarde (Eds.), Kluwer Academic Publishers, 2000. p. 115-130.
35. Vance, P. Crew scheduling, cutting stock and column generation: solving huge integer programs. PhD thesis, Georgia Institute of Technology, 1993.
36. Vance, P.H.; Barnhart, C.; Johnson, E.L.; Nemhauser, G.L. Solving Binary Cutting Stock Problems by Column Generation and Branch-and-Bound. Computational Optimization and Applications 1994; 3: 111-130.

Vitae

Luiz Antonio Nogueira Lorena is a senior researcher of the Applied Mathematics and Computation Laboratory (LAC) at INPE (Brazilian Space Research Institute), Brazil. He received a Dr. Degree in System Engineering and Computation (1985) from COPPE at Federal University of Rio de Janeiro, Brazil. His main research activities are on combinatorial optimization applications with particular emphasis in heuristics. He has published in European Journal of Operational Research, Journal of the Operational Research Society, Evolutionary Computation, Geoinformatica, IEEE TCAD, International Journal of Industrial Engineering, and International Journal of Mathematical Algorithms.

Edson Luiz França Senne is a computer science professor in the Mathematics Department at São Paulo State University, Campus of Guaratinguetá. He received the Doctorate in Applied Computation in 1987 from INPE – Brazilian Space Research Institute. His research interest includes combinatorial optimization applications, decision support systems and geographical information systems. He has published in European Journal of Operational Research, Journal of Intelligent Systems and International Journal of Mathematical Algorithms.

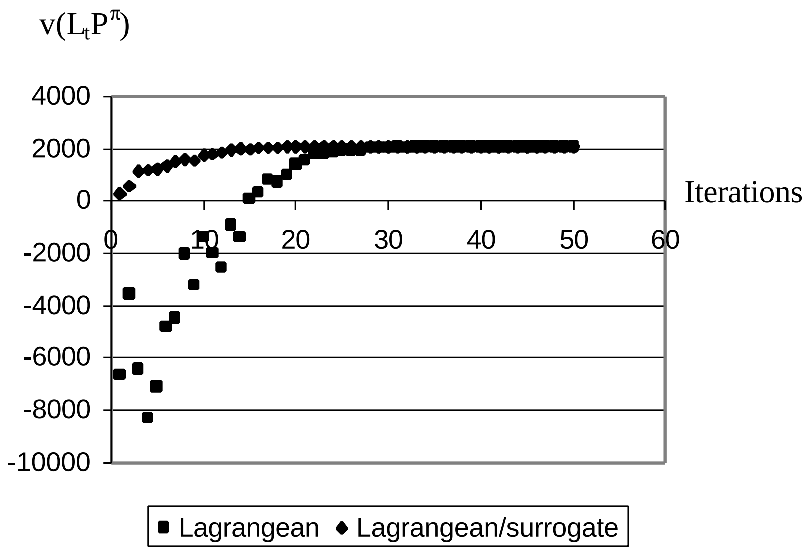

Figure 1. Typical computational behavior of dual bounds

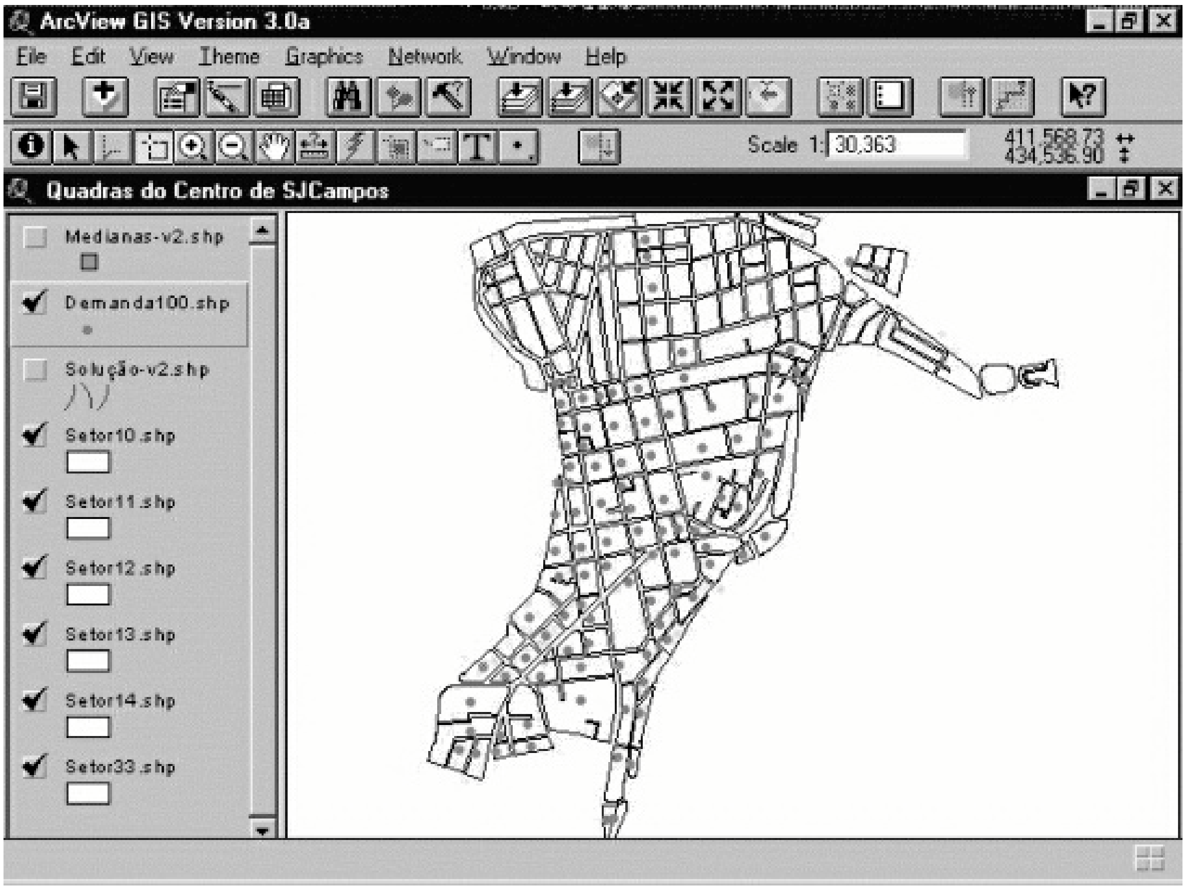

Figure 2: Instance of 100 nodes (central area of São José dos Campos)

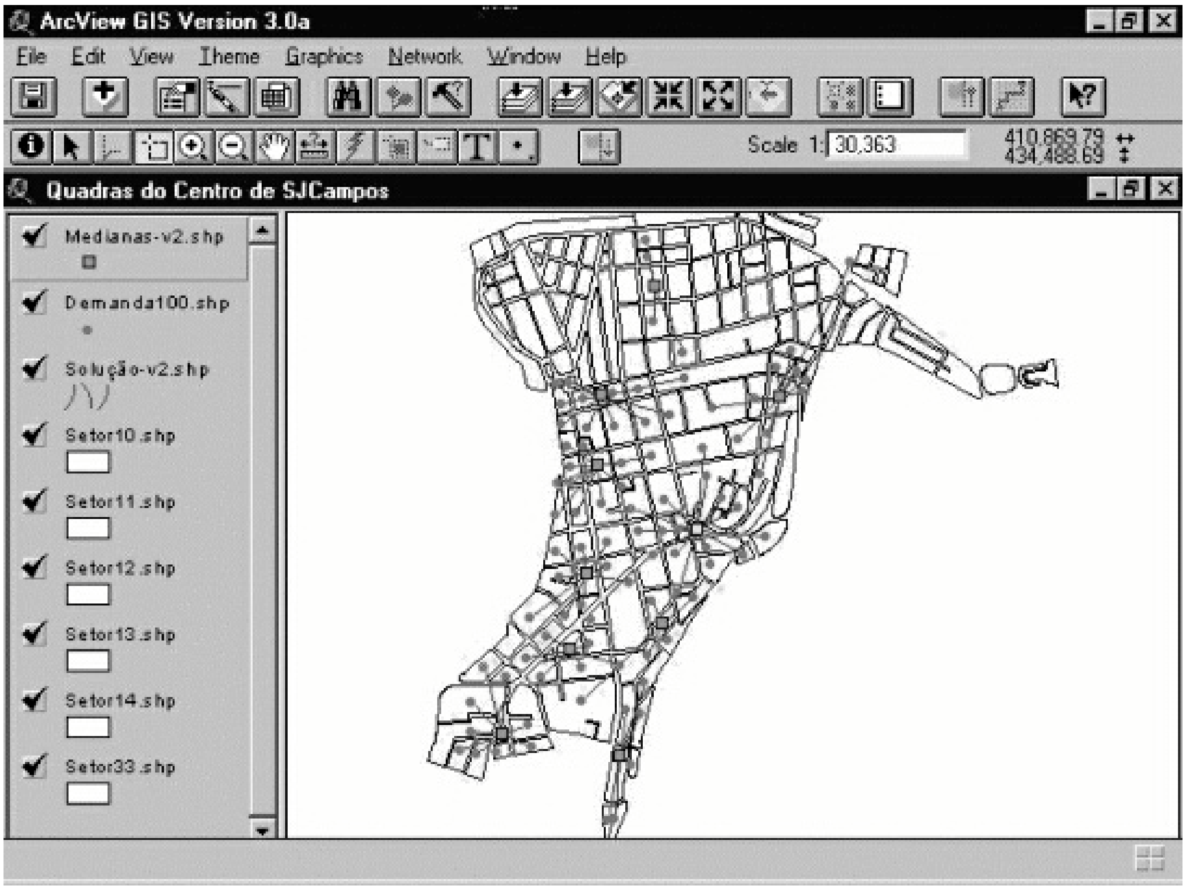

Figure 3: Instance of 100 nodes (solution for 10 medians)

Instance	**n**	**p**	**Best v(SCP)**	**Best v(L_tP^p)**	**Gap-CG**	**Columns Generated**	**Time (s)**
sjc1	100	10	17149.57	17149.56	0.806	3297	10.80
sjc2	200	15	33232.59	33231.89	0.487	9277	114.20
sjc3a	300	25	45244.70	45243.83	0.263	16480	351.51
sjc3b	300	30	40634.36	40634.34	0.002	17011	316.40
sjc4a	402	30	61850.64	61850.20	0.241	30931	1125.08
sjc4b	402	40	52403.89	52403.89	0.451	28643	729.01

Table 1 – Results for CG(t)

The columns contain:

Instance = the instance identification;

n and p = number of nodes and the required number of medians;

Best v(SCP) = the best value obtained to (SCP);

Best v(L_tP^π) = the best (dual) lower bound obtained using the Lagrangean/surrogate relaxation;

Gap-CG = 100 * (Best known feasible solution – Best v(SCP))/(Best known feasible solution);

Columns Generated = number of columns generated;

Time = total computational time (in seconds).

Instance	**n**	**p**	**Best v(SCP)**	**Best $v(L_1P^p)$**	**Gap-CG**	**Columns Generated**	**Time (s)**
sjc1	100	10	16889.02	16893.91	2.313	4287	14.05
sjc2	200	15	33232.59	33231.78	0.487	12167	151.59
sjc3a	300	25	45240.03	45239.55	0.273	22499	553.76
sjc3b	300	30	40635.90	40635.89	0.002	21196	382.24
sjc4a	402	30	61816.25	61815.36	0.241	37240	1522.31
sjc4b	402	40	52369.48	52369.26	0.517	34769	894.25

Table 2 – Results for CG(1)

The columns contain:

Instance = the instance identification;

n and p = number of nodes and the required number of medians;

Best v(SCP) = the best value obtained to (SCP);

Best $v(L_1P^\pi)$ = the best (dual) lower bound obtained using the Lagrangean relaxation;

Gap-CG = 100 * (Best known feasible solution – Best v(SCP))/(Best known feasible solution);

Columns Generated = number of columns generated;

Time = total computational time (in seconds).

Instance	**n**	**p**	**Best known feasible solution**	**Best v(L_tPl)**	**Gap-LS**	**Time**
sjc1	100	10	17288.99	17252.12	0.213	68.62
sjc2	200	15	33395.38	33223.66	0.514	2083.37
sjc3a	300	25	45364.30	45313.43	0.112	2604.92
sjc3b	300	30	40635.90	40634.91	0.002	867.68
sjc4a	402	30	62000.23	61842.49	0.254	27717.11
sjc4b	402	40	52641.79	52396.54	0.466	4649.47

Table 3 – Results for LSLSH(t)

The columns contains:

Best known feasible solution = solution to P obtained using location-allocation heuristics [24];

n and p = number of nodes and the required number of medians;

Best $v(L_tP\lambda)$ = the best (dual) lower bound obtained using the Lagrangean/surrogate relaxation and subgradient method;

Gap-LS = 100 * (Best known feasible solution – Best $v(L_tP\lambda^{)}$)/(Best known feasible solution).

Time = total computational time (in seconds).

	CG(t)						
Number of Columns	**iterations**	**Columns Generated**	**Best v(SCP)**	**Gap-CG**	**Best $v(L_tP^p)$**	**Gap-LS**	**Time**
3500	101	30149	61850.64	0.241	61849.90	0.011	1169.96
3300	102	30544	61850.64	0.241	61850.64	0.013	1097.56
3100	113	32678	61849.29	0.243	61849.26	0.011	1175.25
3000	106	30931	61850.64	0.241	61850.20	0.013	1112.29
2900	96	29156	61850.64	0.241	61850.63	0.013	1079.85
2800	111	32429	61850.64	0.241	61850.23	0.012	1115.98

	CG(1)						
Number of Columns	**iterations**	**Columns Generated**	**Best v(SCP)**	**Gap-CG**	**Best $v(L_1P^p)$**	**Gap-LS**	**Time**
3500	120	39334	61665.98	0.539	61666.28	- 0.284	1672.67
3300	126	38273	61844.14	0.251	61843.67	0.010	1578.46
3100	134	40982	61667.16	0.537	61667.16	- 0.283	1638.28
3000	119	37240	61816.25	0.296	61815.36	- 0.043	1525.35
2900	300	82790	62108.74	- 0.175	54691.75	- 11.562	1855.60
2800	300	90360	62483.74	- 0.779	44403.46	- 28.129	1832.64

Table 4 – Restricting the number of columns at the master problem

---

The columns contain:

Number of columns = fixed number of columns at the master problem.

Columns Generated = number of columns generated;

Best v(SCP) = the best value obtained to (SCP);

Gap-CG = 100 * (Best known feasible solution – Best v(SCP))/(Best known feasible solution);

Best $v(L_tP^{\pi})$ = the best (dual) lower bound obtained using the Lagrangean/surrogate relaxation;

Best $v(L_1P^{\pi})$ = the best (dual) lower bound obtained using the Lagrangean relaxation;

Gap-LS = 100 * (Best known feasible solution – Best $v(L_tP\lambda)$)/(Best known feasible solution);

Time = total computational time (in seconds).

p	**iterations**	**Columns Generated**	**Best v(SCP)**	**Best $v(L_tP^p)$**	**Time**
1000	33	87438	83012.98	83231.58	20210.25
900	36	92578	90131.62	90239.65	25306.54
800	38	98445	98483.26	98530.99	33844.27
700	42	106365	108657.04	108685.59	46705.53
600	48	116623	122020.69	122020.66	59593.02

Table 5. Computational results for CG(t) on Pcb3038 instances

---

The columns contain:

Columns Generated = number of columns generated;

Best v(SCP) = the best value obtained to (SCP);

Best $v(L_tP^\pi)$ = the best (dual) lower bound obtained using the Lagrangean/surrogate relaxation;

Time = total computational time (in seconds).

P	**iterations**	**Columns Generated**	**Best v(SCP)**	**Best $v(L_1P^p)$**	**Time**
1000	83	234140	82876.12	83063.37	38888.77
900	85	243657	89950.80	90009.72	45456.65
800	96	266708	98309.25	98378.65	64686.50
700	103	283213	108658.92	108684.88	90724.47
600	111	311157	121960.16	121980.34	123581.41

Table 6. Computational results for CG(1) on Pcb3038 instances

---

The columns contain:

Columns Generated = number of columns generated;

Best v(SCP) = the best value obtained to (SCP);

Best $v(L_1P^\pi)$ = the best (dual) lower bound obtained using the Lagrangean relaxation;

Time = total computational time (in seconds).