COLUMN GENERATION APPROACH FOR THE POINT-FEATURE CARTOGRAPHIC LABEL PLACEMENT PROBLEM

Abstract

This paper proposes a column generation approach for the Point-Feature Cartographic Label Placement problem (PFCLP). The column generation is based on a Lagrangean relaxation with clusters proposed for problems modeled by conflict graphs. The PFCLP can be represented by a conflict graph where vertices are positions for each label and edges are potential overlaps between labels (vertices). The conflict graph is decomposed into clusters forming a block diagonal matrix with coupling constraints that is known as a restricted master problem (RMP) in a Dantzig-Wolfe decomposition context. The clusters' sub-problems are similar to the PFCLP and are used to generate new improved columns to RMP. This approach was tested on PFCLP instances presented in the literature providing in reasonable times better solutions than all those known and determining optimal solutions for some difficult large-scale instances.

Keywords

- Combinatorial optimization
- Integer programming
- Column generation
- Map Labeling

Authors

Glaydston Mattos Ribeiro[1]

UFES – Universidade Federal do Espírito Santo

CEUNES – Centro Universitário Norte do Espírito Santo

Rua Humberto de Almeida Franklin, 257, Bairro Universitário, São Mateus – ES – Brazil

ZIP CODE 29933-480

Tel/FAX: +55 27 3763-6284-6555

E-mails: glaydstonribeiro@ceunes.ufes.br

Luiz Antonio Nogueira Lorena[2*]

INPE – Insituto Nacional de Pesquisas Espaciais

LAC – Laboratório Associado de Computação e Matemática Aplicada

Avenida dos Astronautas, 1758, Jardim da Granja, São José dos Campos – SP – Brazil

ZIP CODE 12227-010

Tel: +55 12 3945-6555 FAX: +55 12 3945-6357

E-mails: {glaydston[1], lorena[2]}@lac.inpe.br

*Author for correspondence

1. Introduction

The cartographic label placement problem is an important task in automated cartography and Geographical Information Systems (GIS). Labels convey information about objects (or features) in graphical displays like graphs, networks, diagrams, or cartographic maps (Wolff, 1999).

Features can be points (Cities), lines (Railways) or areas (States). However, the point-feature labeling is considered a hard problem to be solved in an automated process, and has consequently received more attention by researchers (Wolff and Strijk, 2006). One of these problems can be described as placing point labels at predefined positions generating a map without overlaps (See Figure 1). In the literature, this problem is known as the Point-Feature Cartographic Label Placement problem (PFCLP) and has been shown to be NP-Hard (Formann and Wagner, 1991; Marks and Shieber, 1991). Thus, exact techniques are not common and metaheuristics/heuristics dominate the solution process (Wolff and Strijk, 2006).

Figure 1 - An example of a map with some overlapping labels (see arrows)

In PFCLP, each point has a list of candidate positions where labels can be placed. This list is defined according to a cartographic standardization (Christensen et al., 1995). Figure 2 (a) shows 8 candidate positions for a point, where the numbers indicate the cartographic preferences in an increasing order.

Figure 2 - Set of 8 candidate positions for one point (Christensen et al., 1995).

Placing labels in candidate positions can generate overlaps (conflicts) compromising the map visibility. Thus, due to these potential overlaps, a PFCLP with *N* points can be represented through a graph $G=\{V,E\}$, where *V* is a set of the candidate positions (vertices) and *E* a set of edges representing overlaps or conflicts. Figure 3(b) shows a conflict graph of the example shown in Figure 3(a). This example has four points (districts at Espírito Santo State – Brazil), each one with 4 candidate positions. The candidate position v_3 has potential conflicts with positions v_1, v_2, v_4 and v_6; v_4 has potential conflicts with v_1, v_2, v_3, v_5 and v_6; and so on. Figure 3(c) shows a solution composed by v_1, v_5, v_9 and v_{15} that is optimal for this problem because it does not present overlaps between labels.

Figure 3 - Candidate positions (a), conflicts graph (b) and an optimal solution (c).

Starting from this conflict graph representation three different approaches are usually considered for PFCLP. The problem can be considered as a Maximum Independent Vertex Set Problem (MIVSP) (Zoraster, 1990; Strijk et al., 2000), as a Maximum Number of Conflict Free Labels Problem (MNCFLP) (Christensen et al., 1994; 1995) and as a Minimum Number of Conflicts Problem (MNCP) (Ribeiro and Lorena, 2006a; 2006c). In all these approaches, the optimal value refers to the number of points in the final solution whose labels are not conflicting. However, the constraints requiring the labeling of a point are treated differently.

Figure 4 – Clusters provided by a conflict graph of a map labeling problem on 250 points. Adapted of Strijk et al. (2000)

Besides, the conflict graph generates clusters of candidate positions (Ribeiro and Lorena, 2006a). For example, Figure 4 shows a conflict graph generated by a problem with 250 points where each one has four candidate positions. The black vertices represent the maximum independent set (Strijk et al., 2000). It is easy to see that this graph is sparse and presents well-defined clusters of candidate positions (see stippled lines). Ribeiro and Lorena (2006a) relax in a Lagrangean way the edges that are connecting the clusters rising to sub-problems that are independently solved. This relaxation was called Lagrangean relaxation with clusters (LagClus).

Taking into account the idea behind the LagClus, this paper presents a column generation approach for the PFCLP. The original graph is partitioned into clusters forming blocks of constraints and the edges that are connecting the clusters form coupling constraints that are all used in a Restricted Master Problem (RMP). This column generation based on clusters was tested upon instances proposed in the literature and their results were successful compared with the best ones known. It provides better solutions than all those reported in the literature in reasonable computational times.

The structure of the paper is as follows. Section 2 has a brief review of the PFCLP. Section 3 presents the Lagrangean relaxation with clusters with the main steps. The column generation approach for the PFCLP is presented in Section 4. Section 5 presents our computational results and the final remarks are presented in Section 6.

2. Literature Review

The Maximal Independent Vertex Set Problem (MIVSP) presents a substantial research considering algorithms and heuristics in different fields. Specifically considering the MIVSP as a PFCLP, Zoraster (1986, 1990 and 1991) formulated mathematically the PFCLP working with conflict constraints and dummy candidate positions of high cost if the points could not be labeled. He also proposed a Lagrangean relaxation for the problem and obtained some computational results on small-scale instances. Strijk et al. (2000) proposed new mathematical formulations and examined a Tabu Search algorithm, obtaining interesting results for their instances. The authors explored some kind of constraints that are known as cut constraints, presented previously by Murray and Church (1996) and Moon and Chaudhry (1984).

The Maximum Number of Conflict Free Labels Problem (MNCFLP) was examined in several works. Christensen et al. (1994; 1995) proposed an Exhaustive Search Approach that alternates positions of the labels previously positioned, to find a better solution. Christensen et al. (1995) also proposed a Greedy Algorithm and a Discrete Gradient Descent Algorithm. These algorithms have difficulty to escape from local maxima. Hirsch (1982) developed a Dynamic Algorithm of label repulsion, where labels in conflicts are moved trying to remove all conflicts. Verner et al. (1997) applied a Genetic Algorithm with mask such that if a label is in conflict, the changing of positions is allowed by crossover operators.

Yamamoto et al. (2002) proposed a Tabu Search algorithm for the MNCFLP that provides good results compared to other methods from the literature. Schreyer and Raidl (2002) applied Ant Colony System but the results were not interesting when compared to the ones obtained by Yamamoto et al. (2002). Yamamoto and Lorena (2005) developed an exact algorithm for small instances of the PFCLP and applied the Constructive Genetic Algorithm (CGA) proposed by Lorena and Furtado (2001) to a set of large-scale instances. The exact algorithm was applied to instances of 25 points and the CGA was applied to instances up to 1000 points, providing better results than Yamamoto's Tabu Search.

Although the MNCFLP presents several different algorithms, it does not have a mathematical formulation like the model proposed by Zoraster (1991). However, almost all heuristics proposed for solving the MNCFLP uses the conflict graph as a base for their mechanism.

The PFCLP, considered as a MIVSP or MNCFLP, can generate large conflict graphs that become hard to deal with it. Wagner et al. (2001) presented an approach to reduce the conflict graph provided by a PFCLP. The authors proposed three rules to reduce the size of the conflict graph without altering the set of optimal solutions. Moreover, they combined these rules with heuristic yielding near-optimal solutions. These rules are presented bellow:

- If p has a candidate position p_i, without any conflicts, declare p_i to be part of the solution, and eliminate all other candidates of p (see Figure 5(a));
- If p has a candidate position p_i that is only in conflict with some q_k, and q has a candidate position q_j ($j \neq k$) that is only overlapped by p_l ($l \neq i$), then add p_i and q_j to the solution and eliminate all other candidates of p and q (see Figure

5(b));

- If *p* has only one candidate position p_i left, and the candidates overlapping p_i form a clique, then declare p_i to be part of the solution and eliminate all candidates that overlap p_i (see Figure 5(c)).

These rules are applied exhaustively. After eliminating a candidate p_i, we must check recursively whether the rules can be applied in the neighborhood of p_i.

Figure 5 – Rules to reduce the conflict graph (Wagner et al., 2001).

Considering now the PFCLP as a MNCP, Ribeiro and Lorena (2006a; 2006c) have proposed two models based on integer linear programming and also a Lagrangean heuristic that have presented the best-known solutions in the literature for the instances proposed by Yamamoto et al. (2002). The second formulation proposed by the authors reduces the number of constraints generated by the first model. This formulation is described bellow.

$$v(MNCP) = Min\left[\sum_{i=1}^{N}\sum_{j=1}^{P_i}\left(w_{ij}x_{ij} + \sum_{c\in C_{ij}} y_{ijc}\right)\right] \tag{1}$$

Subject to:

$$\sum_{j=1}^{P_i} x_{ij} = 1 \quad \forall\, i = 1...N \tag{2}$$

$$\left|C_{ij}\right| x_{ij} + \sum_{(k,t)\in S_{ij}} x_{kt} - \sum_{c\in C_{ij}} y_{ijc} \le \left|C_{ij}\right| \quad \forall\, i = 1...N \quad \forall\, j = 1...P_i \tag{3}$$

$$x_{ij}, x_{kt} \text{ and } y_{ijc} \in \{0,1\} \quad \begin{array}{l} \forall\, i = 1 \ldots N \\ \forall\, j = 1 \ldots P_i \\ c \in C_{ij} \end{array} \tag{4}$$

Where:

- N is the number of points to be labeled and P_i is the set of candidate positions of point i;
- x_{ij} is a binary variable such as $i \in N$ and $j \in P_i$;
- w_{ij} is the cartographic preference assigned to each candidate position. It allowed us to prioritize some candidate positions as described in Figure 2(b);
- S_{ij} is a set of index pairs $(k,t){:}k>i$ of candidate positions such that x_{kt} has potential conflict with x_{ij};
- C_{ij} is a set of all points that contain candidate positions in conflict with the candidate position x_{ij}; and
- y_{ijc} is a conflict variable between the candidate position x_{ij} and the point $c \in C_{ij}{:}c>i$.

Constraint (2) ensures that each point must be labeled with one candidate position. Constraint (3) ensures that if vertices with potential conflicts are chosen to compose the solution, the objective function described in Equation (1) will be penalized. And Equation (4) indicates that all variables in the model are binaries.

In an opposite way, there are some models where the labels can move around of its feature. They are known as slider models (Doddi et al., 1997; van Kreveld et al., 1998; Klau and Mutzel, 2000; 2003), but the present work does not take into account sliding.

3. Lagrangean Relaxation with Clusters (LagClus)

The LagClus (Ribeiro and Lorena, 2006a; 2006b; 2006c) is a stronger relaxation that can be useful for several theoretical and practical large-scale problems. The first application of the LagClus was performed on point-feature instances. Later, Ribeiro and Lorena (2006b) applied this relaxation on pallet loading instances obtaining good results for instances that are considered difficult for a Lagrangean relaxation. Besides, the authors proposed a column generation for that problem using the cluster relaxation idea. Another interesting application was performed on woodpulp stowage context (Ribeiro and Lorena, 2005). This problem consists of arranging items into holds of dedicated maritime international ships. Recently, Côrrea et al. (2006) applied the LagClus to uncapacitated facility location instances providing better bounds than the ones presented in the literature for a set of difficult instances.

For the PFCLP, the best solutions provided by LagClus uses a point graph instead the original conflict graph. Figure 6 shows an example where the original conflict graph (b) is obtained from problem (a). The original graph is transformed in a point graph collapsing the cliques and a graph partitioning heuristic is applied (Figure 6 (c)). Starting from (c), the cliques and the original graph are restored (d). At the end, the edges with terminations in different clusters (e) are relaxed in a Lagrangean way generating small sub-problems that are independently solved (f).

Therefore LagClus follows these steps (Ribeiro and Lorena, 2006a):

i. Apply a graph partitioning heuristic to divide G into $\overline{P}$ clusters. The PFCLP can be written through the objective function defined in (1) subject to (2)-(4)

where the conflict constraints (3) is now divided into two groups: one with conflict constraints corresponding to edges intra clusters and other formed by conflict constraints that correspond to edges connecting the clusters.

ii. Using distinct non-negative multipliers, relax in a Lagrangean way, the conflict constraints corresponding to edges connecting the clusters.

iii. The resultant Lagrangean relaxation is decomposed into $\overline{P}$ sub-problems and solved.

Figure 6 - Partitioning the conflict graph. (Ribeiro and Lorena, 2006a).

Observe that the constraints (2) are not relaxed and appear in clusters, so all relaxed solutions are feasible to PFCLP. Thus, Ribeiro and Lorena (2006a) also used an improvement heuristic that receives a relaxed solution obtained during a subgradient algorithm, and tries to improve it.

Ribeiro and Lorena (2006a) also performed experiments partitioning the original graph, but the best solutions were found partitioning the point graph. For more details, see Ribeiro and Lorena (2006a; 2006c).

4. A Column Generation Approach for the PFCLP

The model (1)-(4) can be rewritten using the decomposition of the original graph into clusters. Let $\overline{P}$ be the number of clusters obtained after the partitioning of the conflict graph *G=(V,E)*. Thus, *G* is partitioned in $G_1(V_1, E_1), G_2(V_2, E_2),\ldots,G_{\overline{P}}(V_{\overline{P}}, E_{\overline{P}})$ and set

$\hat{E} = E \setminus \bigcup_{p=1}^{\overline{P}} E_p$, i. e., $\hat{E}$ represents a set of all edges of G whose ends lie in different clusters.

Thus, the PFCLP formulation (1)-(4) can be rewritten as:

$$v(PFCLP) = Min\left[\sum_{p=1}^{\overline{P}} (\mathbf{x}^p + \mathbf{y}^p) + \overline{\mathbf{y}}\right] \tag{5}$$

Subject to:

$$\begin{bmatrix} A_1 & A_2 & \cdots & A_{\overline{P}} & A_{\overline{\mathbf{y}}} \\ B_1 & 0 & \cdots & 0 & 0 \\ 0 & B_2 & \cdots & 0 & 0 \\ & & \ddots & & \\ 0 & 0 & \cdots & B_{\overline{P}} & 0 \end{bmatrix} \begin{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^1 \\ \vdots \\ \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^{\overline{P}} \\ \overline{\mathbf{y}} \end{bmatrix} \sim \mathbf{R} \tag{6}$$

$$\mathbf{x}^p \; and \; \mathbf{y}^p \in B^{|V_p|} \quad \forall p \in \{1,\ldots,\overline{P}\} \quad \overline{\mathbf{y}} \; is \; a \; vector \; of \; variables \in \{0,1\} \tag{7}$$

Where:

- $\mathbf{x}^p$ is a vector of decision variables (candidate positions) assigned to cluster p;
- $\mathbf{y}^p$ is a vector of variables assigned to conflict constraints defined by (3) with vertices in the same clusters;
- $\overline{\mathbf{y}}$ is a vector of variables assigned to M conflict constraints in (3) with vertices in different clusters;
- A_p is a matrix that represents the variable coefficients assigned to cluster p and also appearing at the M conflict constraints defined by (3) corresponding to $\hat{E}$;

- $A_{\bar{\mathbf{y}}}$ is a matrix that represents the coefficients of the conflict variables inter clusters (See Figure 7);
- B^p is a matrix (block) representing the variable coefficients assigned to cluster *p*;
- **R** is vector with coefficients of the right-hand side of constraints defined in (2) and in (3) for each E_p and $\hat{E}$; and
- ~ are the relational operators = or $\leq$ depending on the respective constraint.

Figure 7 illustrates how to proceed with the formulation above. Note that the relaxed constraint (shaded rectangle) is obtained from a decomposition of the constraints in original formulation that presents vertices in different clusters.

Figure 7 – An example to illustrate variables and matrices in model (5)-(7)

So, relaxing in a Lagrangean way the constraints generated by matrices $A^p \forall p = 1,\ldots,\bar{P}$, the model (5)-(7) can be decomposed into $\bar{P}$ sub-problems. Sub-problem *p* is defined as:

$$v(PFCLP)_p = Min\left[\left(\mathbf{1} + A_p^T \mu\right)\mathbf{x}^p + \mathbf{y}^p : \mathbf{x}^p \ and\ \mathbf{y}^p \in Q_p\right] \qquad (8)$$

Where:

- $\mu \in R_+^M$ are the Lagrangean multipliers assigned to the *M* lines (relaxed constraints) of the matrix A_p; and
- Q_p is the set of constraints embedded in cluster *p*.

Therefore, the LagClus can be written as:

$$v(L_\mu PFCLP) = \sum_{p=1}^{\overline{P}} v(PFCLP)_p + (1 - A_{\mathbf{y}}^T \mu)\overline{\mathbf{y}} - \sum_{m=1}^{M} \mathbf{R}_m \mu_m \tag{9}$$

This formulation was implicitly used by Ribeiro and Lorena (2006a).

The classic implementation of a column generation approach uses a coordinator problem and sub-problems. This coordinator problem also known as Restricted Master Problem (RMP), guides the sub-problems by their dual variables for searching new columns that introduce new information in the RMP.

Thus, applying the Dantzig-Wolfe decomposition (DW) for a linear relaxation (LP) of the blocked constrained problem (5)-(7) generates the following RMP:

$$v(PFCLP_{DW})_{LP} = Min\left[\sum_{p=1}^{\overline{P}} \sum_{j \in J_p} (\mathbf{x}^p + \mathbf{y}^p)\lambda_{jp} + \overline{\mathbf{y}}\right] \tag{10}$$

Subject to:

$$\sum_{p=1}^{\overline{P}} \sum_{j \in J_p} \lambda_{jp} \left[A_p\right] \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^{jp} + \overline{\mathbf{y}} \leq \mathbf{R} \tag{11}$$

$$\sum_{j \in J_p} \lambda_{jp} = 1 \quad \forall p \in \{1,\ldots,\overline{P}\} \tag{12}$$

$$\lambda_{jp} \geq 0 \quad \forall p \in \{1,\ldots,\overline{P}\} \; and \; j \in J_p \tag{13}$$

Where:

- λ_{jp} is a decision variable that represents the extreme point $j \in J_p$; and
- J_p is a set of extreme points of the cluster (sub-problem) *p*.

The $\overline{P}$ sub-problems for this column generation approach are the same shown in

Equation (8), however, the Lagrangean multipliers are replaced by the M dual variables (Δ) corresponding to constraints (11):

$$v(PFCLP)_p = Min\left[\left(\mathbf{1} + A_p^T \Delta\right)\mathbf{x}^p + \mathbf{y}^p : \mathbf{x}^p \text{ and } \mathbf{y}^p \in Q_p\right] \quad \forall p = 1,\ldots,\overline{P} \quad (14)$$

For the RMP, a new column provided by p^{th} cluster is an improving column if $v(PFCLP)_p - \beta_p < 0$, where β_p is a dual variable associated with the p^{th} convexity constraint (12).

The LagClus presented in Equation (9) can also be obtained directly from RMP model (10)-(13) using the formulation:

$$v(L_\Delta PFCLP) = \sum_{p=1}^{\overline{P}} v(PFCLP)_p + (1 - A_{\mathbf{y}}^T \Delta)\overline{\mathbf{y}} - \sum_{m=1}^{M} \mathbf{R}_m \Delta_m \quad (15)$$

Figure 8 describes a diagram of our column generation approach. Note that the number of new columns is used as a stopping condition, so the column generation stops when no more columns present negative reduced costs. After the end of the column generation all decision linear variables present in RMP are transformed to binaries and a binary RMP is solved. This procedure can be considered a heuristic method for solving the PFCLP. A Branch & Price procedure could be used to search optimal solutions. See the recent survey (Desrosiers and Lübbecke, 2005) of column generation for a comprehensive understanding.

Figure 8 – A diagram of the column generation approach

5. Computational Results

The computational tests are performed on instances proposed by Yamamoto et al. (2002) that are available at http://www.lac.inpe.br/~lorena/instancias.html. The set of instances is composed by twenty five instances for 25, 100, 250, 500, 750 and 1000 points. The code in C++ and the tests were done in a computer with Pentium IV (3.33 GHz) processor and 1.0 GB of RAM memory. As done by Zoraster (1990), Christensen et al. (1995) and Yamamoto and Lorena (2005), for all problems the cartographic preferences were not considered. It allowed us to compare our results to the ones presented in literature considering the cost or penalty equal to 1 for all the candidate positions, where the number of those positions is equal to 4: $w_{i,j}=1\ \forall i=1\ldots N$ and $\forall j=1\ldots 4$.

The sub-problems were solved by CPLEX 10 (ILOG, 2006) and the partitioning of conflict graphs were obtained by METIS (Karypis and Kumar, 1998), a well-known heuristic for graph partitioning. Given a conflict graph and a pre-defined number $\overline{P}$ of clusters, METIS divides the graph into $\overline{P}$ sub-graphs of approximately same size minimizing the number of edges whose ends lie in different clusters of the partition. Before divide the conflict graph and test the column generation approach, we applied the technique proposed by Wagner et al. (2001) to reduce the graph.

Table 1 presents results using CPLEX 10 with formulation (1) – (4) upon reduced conflict graphs. The first column represents the average number of points followed by CPLEX's average lower and upper bounds. The fourth column presents *GAP = (Upper Bound - Lower Bound)/Upper Bound*100*, followed by the number of instances optimally solved, average elapsed time in seconds, average number of labels in conflict and the average proportion of free labels found. This last column was used to compare the results with the literature.

These results were found running CPLEX until the instances are solved or reach an out of memory condition. CPLEX solved all instances with 100, 250 and 500 points. For instances with 750 points, it solved 14 among 25 and zero instances with 1000 points. It shows that even with reduced graphs, instances with 1000 points are hard instances to be solved.

Table 1 – Results using CPLEX 10 with reduced conflict graph.

Table 2 reports the main average results provided by the column generation approach. We considered in this paper the same number of clusters used by Ribeiro and Lorena (2006a): 2 clusters for instances with 25, 100, 250 and 500 points, 10 for instances with 750 points and 25 for instances with 1000 points. The sub-problems are solved by CPLEX.

The initial pool of columns is composed of randomly generated solutions followed by an improvement heuristic. The algorithm used to generate these initial columns is shown in Figure 9.

The columns in Table 2 represent:

- Problem – Number of points;

- Best solution inserted in RMP – Best solution inserted in RMP provided by a improvement heuristic (see Figure 9);
- Initial number of columns – Initial number of columns inserted in RMP. This number is calculated by $\overline{P} * Initial\_Solutions$, where *Initial_Solutions* is the initial number of solutions generated by algorithm shown in Figure 9. Note that a solution is decomposed into $\overline{P}$ small solutions to be inserted in RMP;
- Initial RMP – Represent $v(PFCLP_{DW})_{LP}$ with the initial pool of columns;
- Final number of columns – Number of columns after the end of column generation process;
- Final RMP – Represent $v(PFCLP_{DW})_{LP}$ with the final pool of columns;
- Time$_1$ (s) – Elapsed time until the end of column generation process;
- ILP – Final RMP has all decision variables transformed to binaries and solved. This column represents the objective function value found for the binary RMP (See Figure 8);
- Time$_2$ (s) – Elapsed time using binary RMP;
- # of instances solved – Number of instances optimally solved by the column generation approach;
- Labels in conflicts – Number of labels in conflicts found in binary RMP solution;
- Proportion of free labels (%) – Proportion of free labels found in binary RMP solution.

Figure 9 – Algorithm used to create and insert initial columns in RMP.

Table 2 – Average results using column generation approach.

The results reported in Table 2 are very promising. The column generation inserts a small set of new columns into RMP: 52 new columns in average for the instances with 1000 points. The computational times varied from 0.00 to 84.04 seconds for column generation process and from 0.00 to 2.64 seconds for solving the binary RMP, so in the worst case, the column generation approach takes 86.68 seconds to be concluded. Looking at the best solutions initially inserted into RMP, we can note that they are worst than the Final RMP and ILP, showing that our column generation really inserts good columns into RMP.

As our coefficients in objective function (6) are integer, we considered that an instance is optimally solved if the difference between solutions of binary RMP and final RMP is less than 1, besides for all instances the lower bound provided by column generation (column final RMP in Table 2) were the same of the LagClus defined by equation (15). Thus, this column generation approach found the optimal solutions for all instances with 100, 250, 500 and 750 points. For the instances with 1000 points the column generation found 10 optimal solutions against zero optimal solutions of the direct CPLEX application to (1) – (4).

Table 3 reports the main results found in this paper and compares them to the ones provided by CPLEX. Note that the column generation approach surpasses CPLEX for the hard instances (750 and 1000 points).

Table 3 – Main results found with CPLEX and column generation approach

The best results found in this paper were compared to the best-known of the literature described in works of Yamamoto and Lorena (2005) and Ribeiro and Lorena (2006a). Table 4 reports the average proportion of free labels found using our column generation approach and the best-known results found in the literature. Once more, note that those approaches have different objectives however the column generation found better results to PFCLP than all those reported in the literature. The computational times are not compared since the computational tests were performed in different machines.

Table 4 - Comparison with the literature

6. Conclusions

This paper presented a column generation approach for the point-feature cartographic label placement problem. This method provided good solutions in reasonable computational times, improving the best-known solutions in the literature.

This column generation is an interesting technique and can be used for solving several related problems that can be formulated on conflict graphs. It takes in advantage the conflict graph partitioning to form a special mathematical formulation with coupling constraints of edges with vertices in different clusters and block constraints (clusters) that are all considered in a restricted master problem. The Dantzig-Wolfe decomposition generates independent sub-problems (clusters) that are used to introduce new improving columns for the restricted master problem.

Despite our column generation provides interesting results, there are aspects to be explored. In this paper, we performed a static partitioning whose graph is partitioned directly into $\overline{P}$ clusters. However, this partitioning task can be performed hierarchically. This feature can be very useful to provide good solutions and to define the ideal number $\overline{P}$.

We also believe that a heuristic or metaheuristic can be used for solving the sub-problems instead CPLEX. This is an important note because there are several algorithms that find optimal solutions for small-scale problems in reduced computational times, so a hybrid column generation can be a useful approach for several large-scale problems. Finally, we think that a Branch & Price procedure can be easily designed using the column generation approach proposed in this work.

Acknowledgements

The authors acknowledge Conselho Nacional de Desenvolvimento Científico – CNPq for partial financial support.

References

1. A. Wolff, "Automated label placement in theory and practice," PhD thesis, Fachbereich Mathematik und Informatik, Freie Universität Berlin, pp. 153, 1999.
2. A. Wolff and T. Strijk, "The map labeling bibliography," URL on July 10th. http://i11www.ilkd.uni-karlsruhe.de/~awolff/map-labeling/bibliography/, 2006.

3. A.T. Murray and R.L. Church, “Solving the anti-covering location problem using lagrangian relaxation,” Computers and Operations Research, vol. 24(2), pp. 127-140, 1996.

4. F. Wagner, A. Wolff, V. Kapoor and T. Strijk, “Three rules suffice for good label placement,” Algorithmica, vol. 30, pp. 334-349, 2001.

5. F.A. Côrrea, L.A.N. Lorena and E.L.F. Senne, “Lagrangean relaxation with clusters for the uncapacitated facility location problem,” in XIII CLAIO - Congreso Latino-Iberoamericano de Investigación Operativa, Uruguay, Motevideo, 2006.

6. G. Karypis and V. Kumar “Multilevel k-way partitioning scheme for irregular graphs,” Journal of Parallel and Distributed Computing, vol. 48(1), pp. 96-129, 1998.

7. G.M. Ribeiro and L.A.N. Lorena, “Woodpulp stowage using lagrangean relaxation with clusters,” Journal of the Operational Research Society, 2005, To Appear.

8. G.M. Ribeiro and L.A.N. Lorena, “Lagrangean relaxation with clusters for point-feature cartographic label placement problems,” Computers and Operations Research, 2006a. To Appear.

9. G.M. Ribeiro and L.A.N. Lorena, “Lagrangean relaxation with clusters and column generation for the manufacturer’s pallet loading problem,” Computers and Operations Research, 2006b. To Appear.

10. G.M. Ribeiro and L.A.N. Lorena, “Heuristics for cartographic label placement problems,” Computers and GeoSciences, vol. 32(6), pp. 739-748, 2006c.

11. G.W. Klau and P. Mutzel, “Optimal labeling of point features in the slider model,” in D.Z. Du, P. Eades, V. Estivill-Castro, X. Lin and A. Sharma (Eds), Proc. 6th Annual International Computing and Combinatorics Conf. (COCOON'00), vol. 1858. Springer-Verlag, pp. 340-350, 2000.

12. G.W. Klau and P. Mutzel, “Optimal labeling of point features in rectangular labeling models,” Mathematical Programming Ser. B, vol. 94, pp. 435-458, 2003.

13. I.D. Moon and S. Chaudhry, “An analysis of network location problems with distance constraints,” Management Science, vol. 30, pp. 290-307, 1984.

14. ILOG. CPLEX 10, Reference Manual. Mountain View, CA, 2006.

15. J. Christensen, J. Marks and S. Shieber, “Placing text labels on maps and diagrams,” in P. Heckbert (Ed.), Graphics Gems IV. Academic Press, pp. 497-504, 1994.

16. J. Christensen, J. Marks and S. Shieber, “An empirical study of algorithms for point-feature label placement,” ACM Transactions on Graphics, vol. 14(3), pp. 203-232, 1995.

17. J. Desrosiers and M.E. Lübbecke, “A primer in column generation,” in G. Desaulniers, J. Desrosiers and M.M. Solomon (Eds.), Column Generation (GERAD 25th anniversary series), Springer Science+Business Media Inc., New York, 2005, pp. 1-32.

18. J. Marks and S. Shieber, “The computational complexity of cartographic label placement”, Technical Report TR-05-91, Advanced Research in Computing Technology, Harvard University, 1991.

19. L.A.N. Lorena and J.C. Furtado, “Constructive genetic algorithm for clustering problems,” Evolutionary Computation, vol. 9(3), pp. 309-327, 2001.

20. M. van Kreveld, T. Strijk and A. Wolff, “Point set labeling with sliding labels,” in Proceedings of the 14th Annual ACM Symposium on Computational Geometry (SoCG'98), pp. 337-346, 1998.

21. M. Formann and F. Wagner, “A packing problem with applications to lettering of maps,” in Proceedings of the Seventh Annual ACM Symposium on Computational Geometry, New Hampshire, pp. 281-288, 1991.

22. M. Schreyer and G.R. Raidl, “Letting ants labeling point features”, in Proc. of the 2002 IEEE Congress on Evolutionary Computation at the IEEE World Congress on Computational Intelligence, pp. 1564-1569, 2002.

23. M. Yamamoto, G. Câmara and L.A.N. Lorena, “Tabu search heuristic for point-feature cartographic label placement,” GeoInformatica and International Journal on Advances of Computer Science for Geographic Information Systems, vol. 6(1), pp. 77-90, 2002.

24. M. Yamamoto and L.A.N. Lorena, “A constructive genetic approach to point-feature cartographic label placement,” in T. Ibaraki, K. Nonobe and M. Yagiura (Eds), Metaheuristics: Progress as Real Problem Solvers, Kluwer Academic Publishers, pp. 285-300, 2005.

25. O.V. Verner, R.L. Wainwright and D.A. Schoenefeld, “Placing text labels on maps and diagrams using genetic algorithms with masking,” INFORMS Journal on Computing, vol. 9, pp. 266-275, 1997.

26. S. Doddi, M.V. Marathe, A. Mirzaian, B.M.E. Moret, and B. Zhu, “Map labeling and its generalizations,” in Proc. 8th ACM-SIAM Symposium on Discrete Algorithms (SODA'97), pp. 148-157, 1997.

27. S. Zoraster, “Integer programming applied to the map label placement problem,” Cartographica, vol. 23(3), pp. 16-27, 1986.

28. S. Zoraster, “The solution of large 0-1 integer programming problems encountered in automated cartography,” Operations Research, vol. 38(5), pp. 752-759, 1990.

29. S. Zoraster, “Expert systems and the map label placement problem,” Cartographica**, vol.** 28(1), pp. 1-9, 1991.

30. S.A. Hirsch, “An algorithm for automatic name placement around point data,” American Cartographer, vol. 9(1), pp. 5-17, 1982.

31. T. Strijk, B. Verweij and K. Aardal, “Algorithms for maximum independent set applied to map labeling,” Available at ftp://ftp.cs.uu.nl/pub/RUU/CStechreps/CS-2000/2000-22.ps.gz., 2000.

Figures

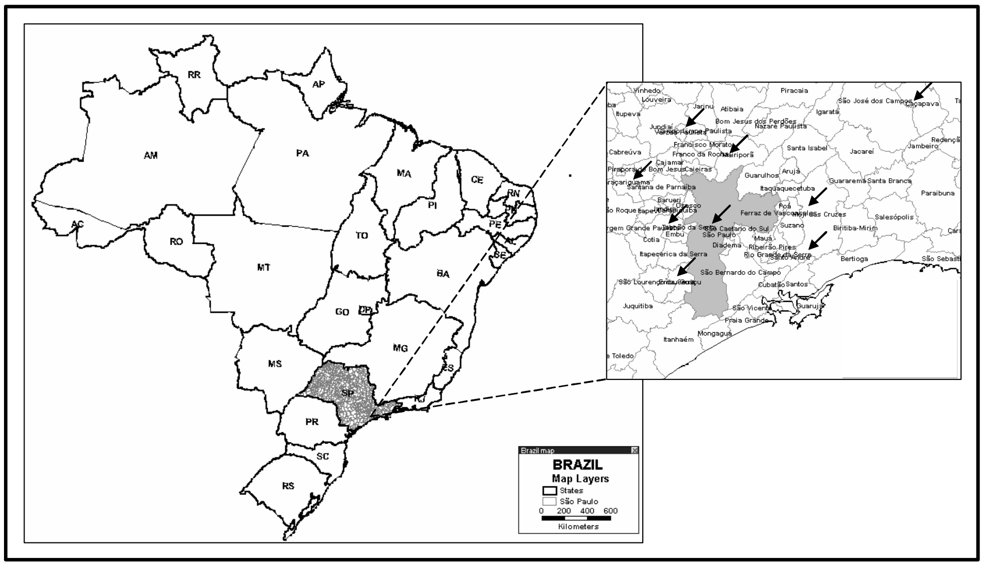


Figure 1 - An example of a map with some overlapping labels (see arrows)

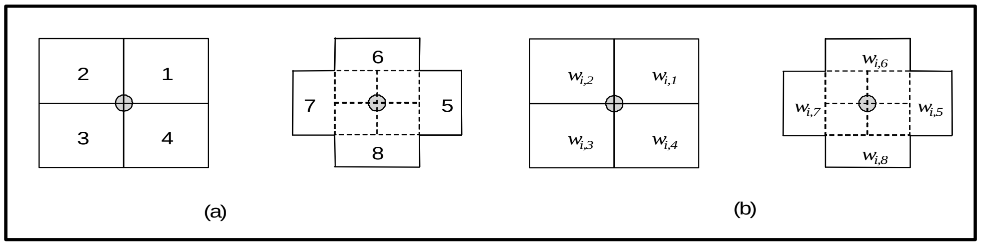
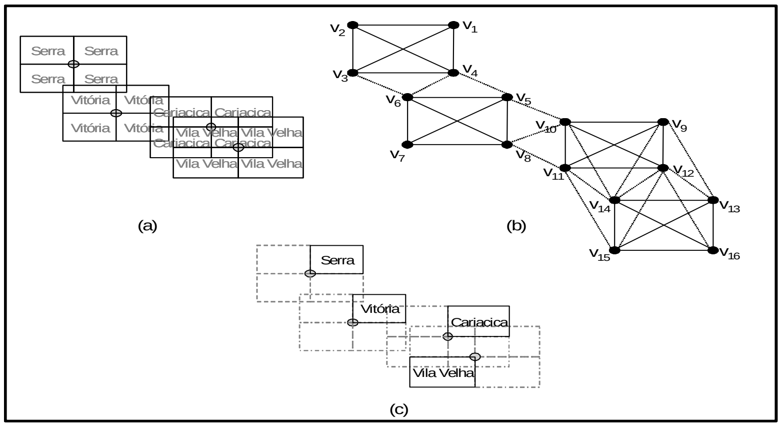


Figure 2 - Set of 8 candidate positions for one point (Christensen et al., 1995).



Serra
Vitória
Cariacica
Vila Velha
(a)
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8 v_9 v_{10} v_{11} v_{12} v_{13} v_{14} v_{15} v_{16}
(b)
(c)

Figure 3 - Candidate positions (a), conflicts graph (b) and an optimal solution (c).

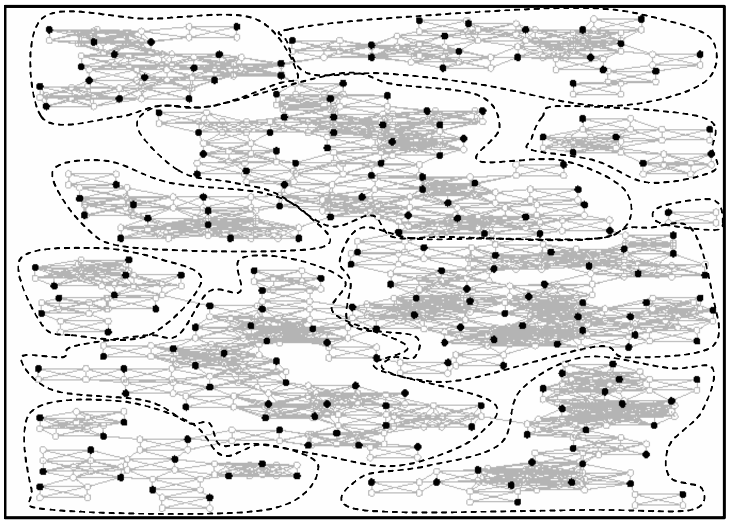


Figure 4 – Clusters provided by a conflict graph of a map labeling problem on 250 points. Adapted from Strijk et al. (2000)

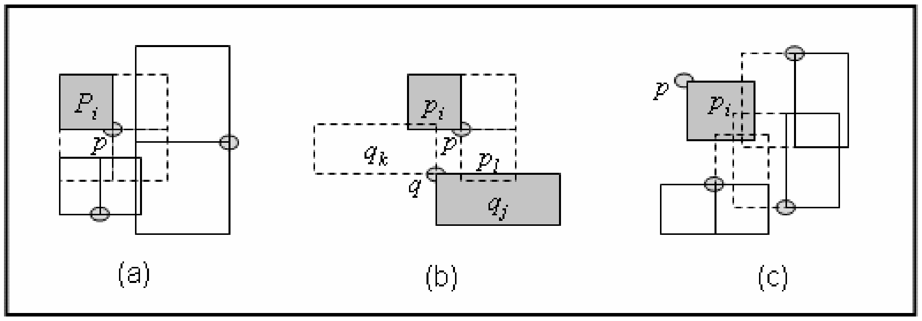


Figure 5 – Rules to reduce the conflict graph (Wagner et al., 2001).

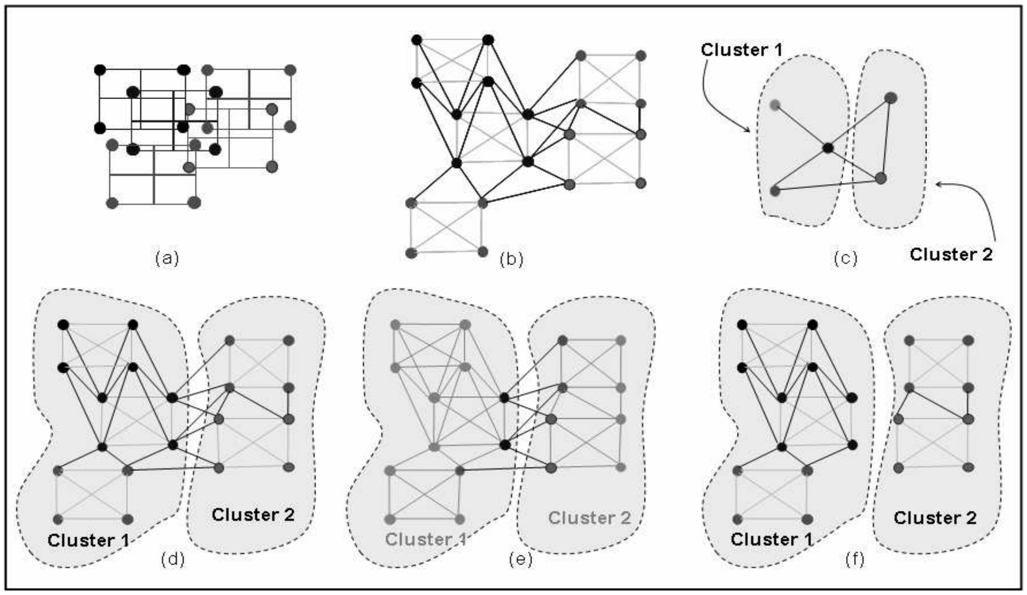
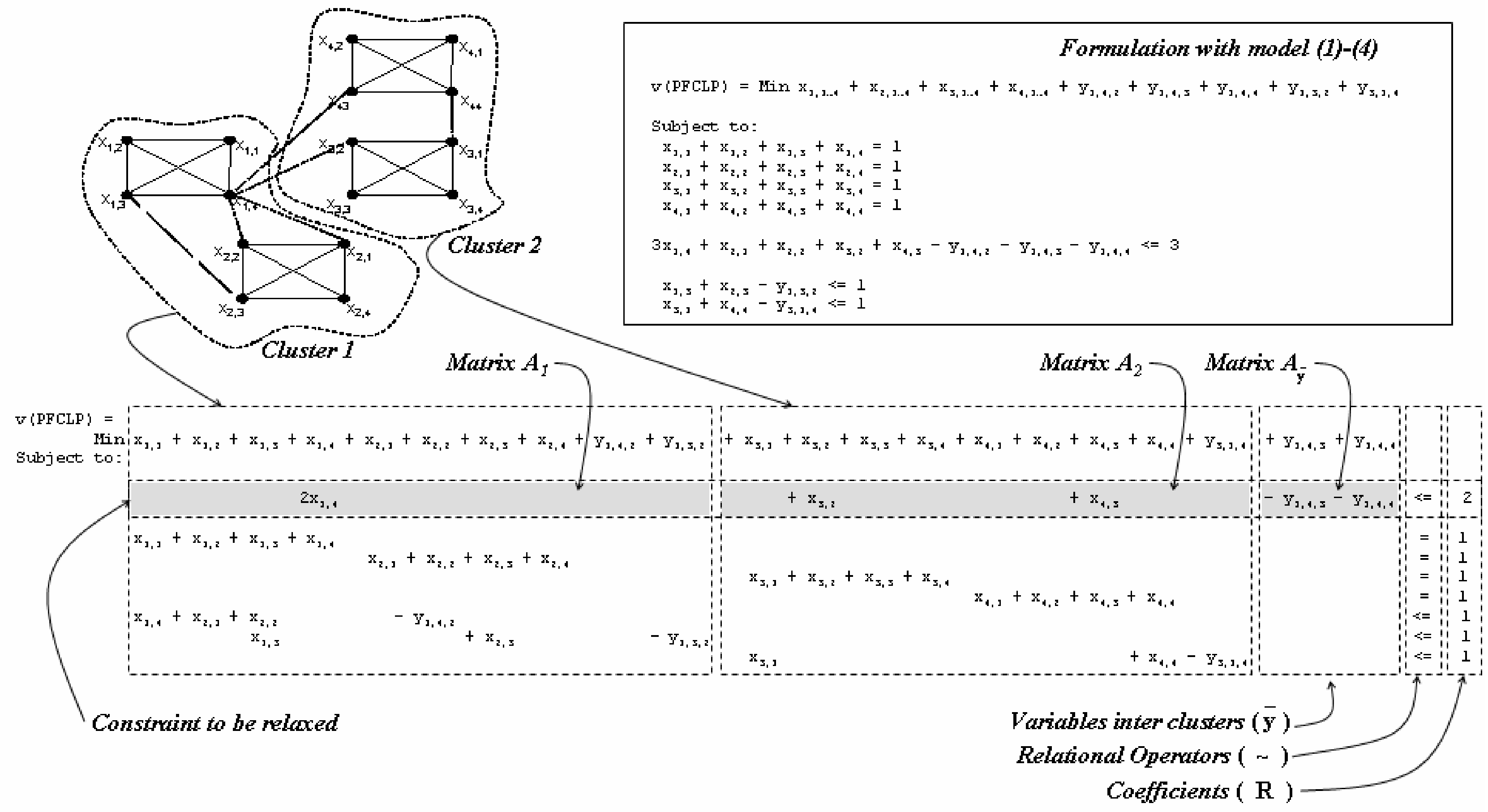


Figure 6 - Partitioning the conflict graph (Ribeiro and Lorena, 2006a).



Formulation with model (1)-(4)

$$
\begin{aligned}
&v(PFCLP) = \text{Min } x_{1,1..4} + x_{2,1..4} + x_{3,1..4} + x_{4,1..4} + y_{1,4,2} + y_{1,4,3} + y_{1,4,4} + y_{1,3,2} + y_{3,1,4}\\
&\text{Subject to:}\\
&\quad x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} = 1\\
&\quad x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} = 1\\
&\quad x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} = 1\\
&\quad x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} = 1\\
&3x_{1,4} + x_{2,1} + x_{2,2} + x_{3,2} + x_{4,3} - y_{1,4,2} - y_{1,4,3} - y_{1,4,4} <= 3\\
&\quad x_{1,3} + x_{2,3} - y_{1,3,2} <= 1\\
&\quad x_{3,1} + x_{4,4} - y_{3,1,4} <= 1
\end{aligned}
$$

$$
\begin{array}{l|l|l|l|c|c}
\text{v(PFCLP) = Min} & x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} + x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} + y_{1,4,2} + y_{1,3,2} & + x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} + x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} + y_{3,1,4} & + y_{1,4,3} + y_{1,4,4} & & \\
\text{Subject to:} & 2x_{1,4} & + x_{3,2} \quad + x_{4,3} & - y_{1,4,3} - y_{1,4,4} & <= & 2\\
 & x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} & & & = & 1\\
 & x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} & & & = & 1\\
 & & x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} & & = & 1\\
 & & x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} & & = & 1\\
 & x_{1,4} + x_{2,1} + x_{2,2} \quad - y_{1,4,2} & & & <= & 1\\
 & x_{1,3} \quad + x_{2,3} \quad - y_{1,3,2} & & & <= & 1\\
 & & x_{3,1} \quad + x_{4,4} - y_{3,1,4} & & <= & 1
\end{array}
$$

Figure 7 – An example to illustrate variables and matrices in model (5)-(7)

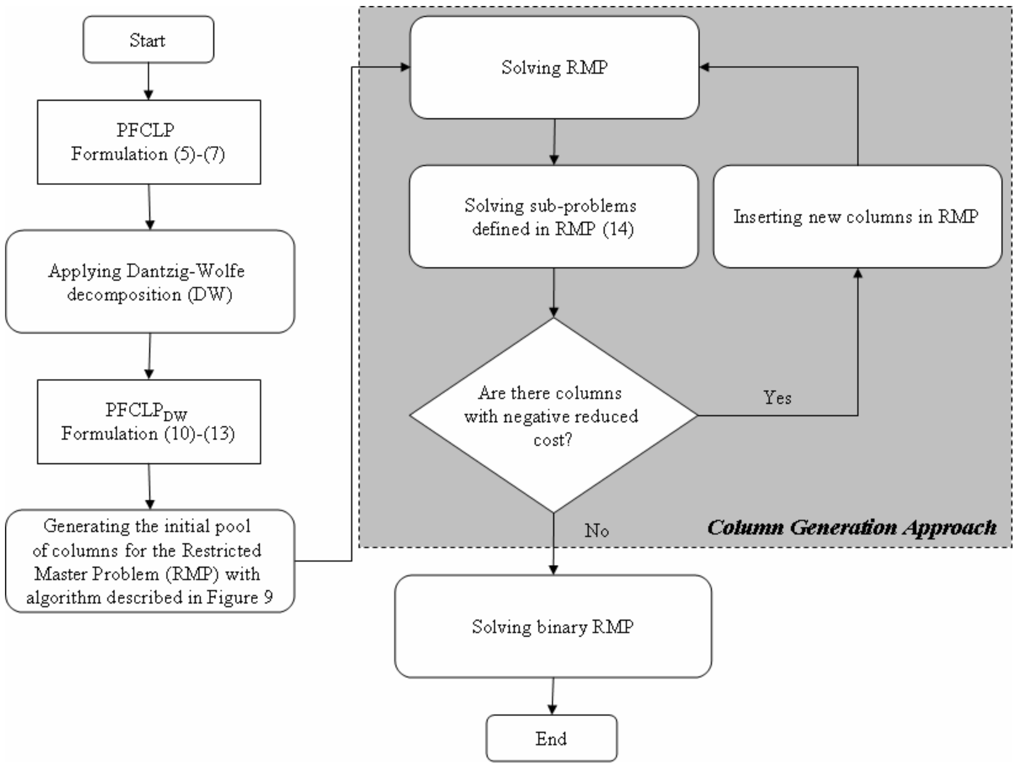

Start
PFCLP Formulation (5)-(7)
Applying Dantzig-Wolfe decomposition (DW)
PFCLP$_{DW}$ Formulation (10)-(13)
Generating the initial pool of columns for the Restricted Master Problem (RMP) with algorithm described in Figure 9
Solving RMP
Solving sub-problems defined in RMP (14)
Inserting new columns in RMP
Are there columns with negative reduced cost?
Yes
No
Column Generation Approach
Solving binary RMP
End

Figure 8 – A diagram of the column generation approach.

```
Algorithm Insert_solutions_in_RMP(Initial_Solutions)

// Let:
// - Random_Generation_Solution(N) be a function that generates
//     a random solution for N points
// - Object_Function(Sol) be the objective function described in
//     Equation (1)
// - Insert_in_RMP(Sol) be a function that insert Sol in
//     restricted master problem

Best_Solution ← ∞
For i_sol=1 To Initial_Solutions Do
  Sol ← Random_Generation_Solution(N)

  IMPROVEMENT HEURISTIC
  Bkp_Sol ← Sol
  Found_New_Solution ← true
  fCurrent ← Object_Function(Bkp_Sol)
  Bkp_Current ← fCurrent
  While Found_New_Solution Do
    Found_New_Solution ← false
    For i = 1 To N Do
      Current_Candidate_Position = Bkp_Sol [i]
      For $\forall j \in P_i$ Do
              If j== Bkp_Sol [i] Then Continue
        Bkp_Sol [i] ← j
        If (Object_Function(Bkp_Sol)<fCurrent) Then
          fCurrent ← Object_Function(Bkp_Sol)
          Found_New_Solution ← true
          Best_Neighbor ← j
          Changed_Point ← i
         End If
      End For
      Bkp_Sol [i] ← Current_Candidate_Position
    End For
    If Found_New_Solution Then
      Bkp_Sol [Changed_Point] ← Best_Neighbor
    End If
  End While
  If (fCurrent<Bkp_Current)And(fCurrent!=Best_Solution) Then
    Sol ← Bkp_Sol
    If (fCurrent < Best_Solution) Then
      Best_Solution ← fCurrent
    End
  End

  Insert_in_RMP(Sol)
End For
```

Figure 9 – Algorithm used to create and insert initial columns in RMP.

Tables

Table 1 – Results using CPLEX 10 with reduced graphs.

Reduced Conflict Graph							
Problem	**Lower Bound**	**Upper Bound**	**GAP (%)**	**# of instances solved**	**Time (s)**	**Labels in conflicts**	**Proportion of free labels (%)**
100	100.00	100.00	0.00	25	0.00	0.00	100.00
250	250.00	250.00	0.00	25	0.01	0.00	100.00
500	500.84	500.84	0.00	25	0.14	1.68	99.67
750	757.10	758.92	0.24	14	5517.12	17.60	97.65
1000	1004.87	1042.88	3.64	0	4504.52	83.12	91.69

Table 2 – Average results using column generation approach.

Problem	**Column Generation**						**Integer RMP**		**# of instances solved**	**Labels in conflicts**	**Proportion of free labels (%)**
	Best Solution Inserted in RMP	**Initial Number of Columns**	**Initial RMP**	**Final Number of Columns**	**Final RMP**	**Time$_1$ (s)**	**ILP**	**Time$_2$ (s)**			
100	100.04	2000	100.00	2002	100.00	0.00	100.00	0.08	25	0.00	100.00
250	250.84	2000	250.00	2002	250.00	0.00	250	0.12	25	0.00	100.00
500	504.00	2000	502.84	2003.04	500.84	0.16	500.84	0.00	25	1.64	99.67
750	785.52	10000	765.16	10013.40	758.92	14.48	758.92	0.48	25	17.50	97.67
1000	1123.24	25000	1048.97	25052.44	1037.56	84.04	1039.04	2.64	10	76.04	92.40

Table 3 – Main results found with CPLEX and column generation approach

Problem	**Approach**	**Lower Bound**	**Upper Bound**	**GAP (%)**	**# of instances solved**	**Time (s)**	**Labels in conflicts**	**Proportion of free labels (%)**
100	CPLEX	100.00	100.00	0.00	25	0.00	0.00	100.00
	Column Generation	100.00	100.00	0.00	25	0.08	0.00	100.00
250	CPLEX	250.00	250.00	0.00	25	0.01	0.00	100.00
	Column Generation	250.00	250.00	0.00	25	0.12	0.00	100.00
500	CPLEX	500.84	500.84	0.00	25	0.14	1.68	99.67
	Column Generation	500.84	500.84	0.00	25	0.16	1.64	99.67
750	CPLEX	757.10	758.92	0.24	14	5517.12	17.60	97.65
	Column Generation	**758.92**	**758.92**	**0.00**	**25**	**14.96**	**17.50**	**97.67**
1000	CPLEX	1004.87	1042.88	3.64	0	4504.52	83.12	91.69
	Column Generation	**1037.56**	**1039.04**	**0.14**	**10**	**86.68**	**76.04**	**92.40**

Table 4 – Comparison with the literature

Algorithm	**Proportion of free labels (%)**				
	Problems				
	100	**250**	**500**	**750**	**1000**
Column Generation Approach	100.00	100.00	99.67	97.67	92.40
LagClus (Ribeiro and Lorena, 2006a)	100.00	100.00	99.67	97.65	91.42
CGA_{Best} (Yamamoto and Lorena, 2005)	100.00	100.00	99.60	97.10	90.70
$\text{CGA}_{\text{Average}}$ (Yamamoto and Lorena, 2005)	100.00	100.00	99.60	96.80	90.40
Tabu Search (Yamamoto et al., 2002)	100.00	100.00	99.30	96.80	90.00
GA with masking (Verner et al., 1997)	100.00	99.98	98.79	95.99	88.96
GA (Verner et al., 1997)	100.00	98.40	92.59	82.38	65.70
Simulated Annealing (Christensen et al., 1995)	100.00	99.90	98.30	92.30	82.09
Zoraster (Zoraster, 1990)	100.00	99.79	96.21	79.78	53.06
Hirsh (Hirsh, 1982)	100.00	99.58	95.70	82.04	60.24
3-opt Gradient Descent (Christensen et al., 1995)	100.00	99.76	97.34	89.44	77.83
2-opt Gradient Descent (Christensen et al., 1995)	100.00	99.36	95.62	85.60	73.37
Gradient Descent (Christensen et al., 1995)	98.64	95.47	86.46	72.40	58.29
Greedy Algorithm (Christensen et al., 1995)	95.12	88.82	75.15	58.57	43.41