Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXIX, págs. 9499-9509 (artículo completo)
Eduardo Dvorkin, Marcela Goldschmit, Mario Storti (Eds.)
Buenos Aires, Argentina, 15-18 Noviembre 2010

# EFFICIENT ALGORITHMS FOR THE CAPACITY AND CONNECTIVITY GRAPH PARTITION PROBLEM

**Gustavo S. Semaan [a], José André M. Brito[b], e Luiz S. Ochi[c]**

[a]*Instituto de Computação - Universidade Federal Fluminense, Rua Passo da Pátria 156 - Bloco E, 3º andar, São Domingos, Niterói, RJ, Brasil gsemaan@ic.uff.br, http://www.ic.uff.br*

[b]*ENCE - Escola Nacional de Ciências Estatísticas, Rua André Cavalcanti,106.Rio de Janeiro, RJ, jose.m.brito@ibge.gov.br*

[c] *Instituto de Computação - Universidade Federal Fluminense, Rua Passo da Pátria 156 - Bloco E, 3º andar, São Domingos, Niterói, RJ, Brasil satoru@ic.uff.br, http://www.ic.uff.br*

**Keywords:** Minimum Spanning Tree, Graph Partition Problem, Clustering

**Abstract.** This paper proposes some algorithms for the resolution of a real capacity and connectivity graph partition problem. A review of literature showing that the algorithms work only with the edges of the Minimum Spanning Tree is presented. In this case, the algorithms act on the original graph, in order to increase the possibilities of vertex migration. Several experiments over a set of real data were used. The results showed a new and efficient way to solve this problem, in which the algorithms improved both the solution´s quality and the formation of valid solutions.

## 1 INTRODUCTION

Cluster analysis is a method of creating groups of objects, called clusters, in such a way that objects in the same cluster are very similar and objects in different clusters are quite distinct Han and Kamber (2005). Considering a given set with *n* objects X = {$x_1$, .., $x_n$}, it must extract partitions from the set *X* in *K* different clusters $C_i$, respecting the following three conditions:

$$\bigcup_{i=1}^{k} C_i = X \qquad C_i \neq \emptyset, 1 \leq i \leq k \qquad C_i \cap C_j = \emptyset, 1 \leq i, j \leq k, i \neq j$$

Since in this problem the number of necessary clusters is known, it is called k-clusterization or, simply, Clusterization Problem. In this case, the number of possible solutions will be given by Stirling's number of the second kind (Eq. (1)):

$$S(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^i \binom{k}{i}(k-i)^n \tag{1}$$

Aiming to demonstrate the high combinatorial possibilities, some examples concerning to number of solutions where *n* is a number of objects and *k* the number of clusters are presented: S(10,2)=511; S(10,5)=4.2x$10^4$; S(10,9)=45; S(10,10)=1; S(100,2)=6.3x$10^{29}$; S(100,10)=2.7x$10^{93}$.

The cluster analysis is a fundamental technique to experimental sciences in which the classification of elements into groups is desirable. As examples of these fields it is possible to cite: biology, medicine, economy, psychology, market, statistic among others.

## 2 GRAPH PARTITION PROBLEM

In many applications the clustering problem can be presented by graphs, considering a problem of partition graphs. This consists in grouping the vertex of the graphs in different subsets (clusters), according to their similarities, by using a fitness function (Assunção *et al.*, 2006; Dias and Ochi, 2003; Doval *et al.*,1999). Moreover, the addressed problem considers the following restrictions:

- *Connectivity*: the vertexes grouped in each cluster must be connected.
- *Minimum Capacity* (total associated to one of the variables): the total of this variable in each cluster, considering all its respective vertices, must be bigger or, at least, equal to a previously defined value.

The experiments use instances of regionalization statistical field, in which real, instances of Brazilian Demographic Census and Socio-economic Research were mapped into graphs. This way, each vertex *i* corresponds to one area that contains its *p* variables. Moreover, if two areas *i* and *j* are neighbors, it is possible to find an edge and its value is $d_{ij}$, where $d_{ij}$ is a measure of similarity between two areas (e.g. see Eq. (2)):

$$d_{ij} = \sqrt{\sum_{s=1}^{p}(x_i^s - x_j^s)^2} \tag{2}$$

## 3 REVIEW OF LITERATURE

Papers about the general problem of clustering, including graph partition problem and regionalization problem, in which are considered restriction of connectivity and capacity, had been widely reported in literature.

Some Groups (Scheuerer, 2006; Shieh and May, 2001) had proposed heuristics algorithms for the capacity clustering problem, while others (Assunção *et al.*, 2002; Assunção *et al.*, 2006; Neves, 2003) had suggested algorithms for the regionalization problem, in which the connectivity restriction was considered. Even considering this restriction, the Automatic Zoning Procedure (AZP) was proposed by Assunção *et al.*, (2006).

The problem presented in this paper considers both connectivity and capacity restrictions into partition graph problem. This way, Brito *et al.*, (2004) proposed an integer programming formulation, while other groups (Semaan *et al.*, 2008; Semaan *et al.*, 2009) proposed algorithms with local search procedures.

It is important to underline that, excepting the AZP, all the other work referenced that considered the connectivity restriction were based on Minimum Spanning Tree (MST) Partition Method, which consists, basically, of two steps:

1. Construction of a MST from the graph which represents the problem.
2. Formation of sets of clusters by the algorithm, through of partitioning of MST.

Considering the connectivity restriction of the graph $G$, one natural solution for the problem will consist of building a MST $T$ from $G$, respecting the smaller values of $d_{ij}$. Once provided one tree $T$ and a number $K$ of partitions (cluster to be generated), it is possible to extract $(K - 1)$ edges from $T$, defining, this way, a set of $K$ subtrees $T_j$, $j = 1..K$. Each one of these subtrees will be associated to one cluster.

The connectivity property can be observed in each of the subtrees (clusters). Thus, the solution for the problem will consist of partitioning $T$ in $K$ subtrees $T_j$, $j = 1,.., K$ associated to cluster what satisfies the capacity restriction and results in the lower possible value for a fitness function, where $p$ are the variables and $n$ the vertexes (Eq. (3))

$$f = \sum_{j=1}^{p} \sum_{i=1}^{n} (x_{ij} - \overline{x_j})^2 \tag{3}$$

The case of AZP was based on the spatial object neighbor structure to assure the connectivity restriction and acts, basically, on the migration of the objects in order to minimize a fitness solution.

The minimum capacity can be either submitted as parameter or calculated at the begin of the algorithm, which $\beta$ is the fit factor, $K$ a number of clusters, $n$ a number of vertexes and $x_{il}$ the capacity variable of the vertex $l$ (see (Eq. (4)).

$$Cap_{Min} = (\beta / k).\sum_{j=1}^{n} x_{lj} \tag{4}$$

## 4 PROPOSED ALGORITHMS

Through the review of literature it was possible to observe that the proposed algorithms work only on the edges of the MST. In other way, this work presents new versions of local

search that acts with the original submitted graph of the problem, increasing the possibilities of vertex migration and thus facilitating the formation of not only valid, which the restriction of capacity is respected, but also better quality solutions.

According to Doval *et al.* (1999), a good data structure for the problem is extremely important to the algorithms performance and it can be decisive for a fast convergence and quality of the obtained solutions. This way, the structure used for the representation of the solution was a *group-number*, also used by (Dias and Ochi, 2003; Doval *et al.*, 1999; Semaan *et al.*, 2008; Semaan *et al.*, 2009), in which the index of vector represents the vertex of the graph and its content represents the cluster to which the vertex belongs.

The proposed approach consists in creating solutions using the MST Partition Method through the constructive heuristics, and so, refining its using local search procedures consider the original graph, and not only the MST built. In this sense this paper used concepts of GRASP and Evolutionary Algorithms (Glover and Kochenberger, 2002).

### 4.1 Constructive Heuristics

Three versions of constructive heuristics had been proposed. Assuring the connectivity restriction through MST Partition Method. Moreover, two versions worked aiming to build valid solutions, which the restriction of capacity is respected. The other version acted in order to minimize the fitness solution, independently of the restriction of capacity. All three constructive heuristics versions acted to generate $K$ partitions, removing $(K - 1)$ edges from $T$. Hierarchical division strategy was used, in which, initially, all the vertexes belong to the same cluster.

The Constructive Heuristic 1 (CH1) was proposed by Semaan *et al.* (2008) and consists in, after the selection of the cluster that must be partitioned (what have the high fitness function), to evaluate all the possibilities of edge removal in order to minimize the fitness function. This way, must be removed the edge of high $C_{edge}$ value of Eq.(5).

$$C_{edge} = f(T_i) - (f(T_i^1) + f(T_i^2)) \tag{5}$$

Although it is a greedy procedure which has high operational cost, it was applied on the building of the initial solution for the proposed algorithm. I order to make this algorithm semi-greedy, it was used a Restricted Candidate List (RCL) [15], which the $\alpha$ high edges are selected and, one of them is randomly selected, in order to divide the selected cluster.

The CH2 was presented by Semaan *et al.* (2009), aiming to form valid solution, being composed by four steps:

- The first step consists in the random selection of two vertexes (*a* and *b*) which will be the ends of a principal subtree (Semaan, 2010).
- Selection of the other vertexes on the way between *a* and *b*, forming the principal subtree.
- Partition of the subtree, in order to form valid solutions.
- If necessary, regenerate the solution, which must have K clusters.
- The randomness factor was used aiming to form different solutions, both in the end vertexes selection as the partition of the principal subtree.

The CH3 was based on the CH1 but, in this version, intending to obtain valid solutions. In this case, the selection of the cluster that must be partitioned occurs by capacity criteria, in

which the cluster with higher capacity must be selected. Moreover, the algorithm is also semi-greedy and a RCL was used.

In order to build valid solutions, the CH3 acts dividing the selected cluster $C_w$ in the clusters $C_{w1}$ and $C_{w2}$ and, afterwards, one of them must have its capacity minimized and the capacity criteria respected.

### 4.2 Local Search Procedures

Six versions of local search (LS) were used, among them, three consider only the edges of the MST built (versions LS1, LS4, LS5) and other three were applied using all edges from the original submitted graph (LS2, LS3, LS6), improving the freedom of vertex migration and thus, facilitating the formation of, not only valid, but also better quality solutions.

Afterwards, three of these versions acted aiming to form valid solutions (LS1, LS2, LS5), while the others acted in order to minimize the fitness solution, independent of the restriction of capacity (LS3, LS4, LS6). All six local search versions were based on vertex migration among clusters.

The LS1 was proposed by Semaan *et al.*, (2009) and, as well as the procedures that consider only edges of the MST, uses the edges that were selected during the cluster partition. Basically, the procedure verifies if one and only one cluster associated to vertexes of the edge is penalized. In this case, the vertex is migrated to other cluster, aiming to regenerate the solution.

The LS2 and LS3 versions realize migrations of vertex based on the original submitted graph of the problem, where the first acts in order to regenerate the solutions and the second aiming to minimize the fitness´ solution. It is necessary to verify the restrictions of this problem aiming to form both valid and best solutions.

The LS4 and LS5 versions work joining adjacent clusters in which exists an edge connecting vertexes of this clusters, and after, dividing this cluster using, respectively, the CH1 and CH3 procedures.

The LS6 was based on the known clustering algorithm of the literature, the K-Means (MacQueen, 1967) but, in this case, the restrictions of this problem were considered.

### 4.3 Crossover

The crossover operator (C1) considers only the edges of the MST built and it was applied to combine two solutions, aiming to obtain new best solutions. 1-point type crossover was used and it was necessary to validate if a common removed edge between the solutions exists and if the new solutions have *K* clusters.

### 4.4 Mutation

The Mutation M1 was proposed by Semaan *et al.* (2008), based on MST built. This procedure assures the connectivity criteria and solutions with *K* clusters and acts in the vertexes migration among clusters, aiming to perturb the solution.

### 4.5 Elitism

The elitism technique stores the best found solutions, both valid and invalid. At the beginning of each generation, one of these solutions is inserted to the population in order to

improve quality by using the others procedures. When there is no valid solution, the best penalized solution is inserted into the population. At the final of execution, the algorithm will present the best penalized solution and, if it was found, the best valid solution. In this paper only valid solutions were considered in the computational results.

## 5 COMPUTATIONAL RESULTS

For several experiments, twenty six real instances of Brazilian Demographic Census and Socio-economic Research were used. The Table 1 shows details of the used instances, where |V| is the vertex count and |E| is the edge count. Moreover, the algorithms presented were coded in Ansi C, running on a Intel Centrino II 2,4 Ghz processor, 4GB Ram and Windows Vista™.

| Id | \|V\| | \|E\| | Id | \|V\| | \|E\| |
|---|---|---|---|---|---|
| 1 | 21 | 58 | 14 | 178 | 791 |
| 2 | 61 | 286 | 15 | 121 | 567 |
| 3 | 409 | 2020 | 16 | 75 | 359 |
| 4 | 73 | 350 | 17 | 114 | 502 |
| 5 | 14 | 46 | 18 | 133 | 620 |
| 6 | 18 | 59 | 19 | 195 | 868 |
| 7 | 89 | 363 | 20 | 68 | 307 |
| 8 | 16 | 60 | 21 | 181 | 843 |
| 9 | 57 | 236 | 22 | 151 | 560 |
| 10 | 375 | 1769 | 23 | 86 | 388 |
| 11 | 179 | 882 | 24 | 155 | 722 |
| 12 | 74 | 357 | 25 | 461 | 2385 |
| 13 | 231 | 1172 | 26 | 285 | 1451 |

Table 1: used instances.

The algorithms were divided in 2 groups, the group MST considers only edges of the MST and the OG works with the submitted original graph. Moreover, both groups used the three constructive heuristics. All MST versions used the procedures LS1, LS4, LS5, M1 and C1, while the OG versions used LS2, LS3, LS6.

This way, the algorithm version labels were composed by group and the version of the constructive heuristic. The MST3, for example, represented the algorithm that used constructive heuristic 3 and considered only edges of the MST (inclusive the associated procedures LS1, LS4, LS5, M1 and C1). Thus, the experiments used six algorithms: MST1, MST2, MST3, OG1, OG2, OG3.

The parameter values were fitted through the preliminary experiments, and were used: Kgroups = 3, populationSize = 10, generationsCount = 100, crossoverProbability = 80%, mutationProbability = 5% and minCapacity( $\beta$ ) = 30%.

In the first experiment each algorithm executed over the same instance set ten times. In order to analyze the results, three categories were created according to the Gap (Eq. (6)) of the best solution obtained: Best (solution with Gap = 0%), Interesting (Gap <= 5%) and bad (Gap > 70%).

Figure 1 shows the quantity of solutions per categories. It is possible to observe that all OG versions obtain solutions of best category and no solution of bad category. In other way, all MST versions obtain solutions of bad category. Moreover the OG1 and OG2 obtain, in all the runs (instances), solutions with Gap lower than 5%.

$$Gap = 100 * \frac{solution - solution_{best}}{solution_{best}} \tag{6}$$

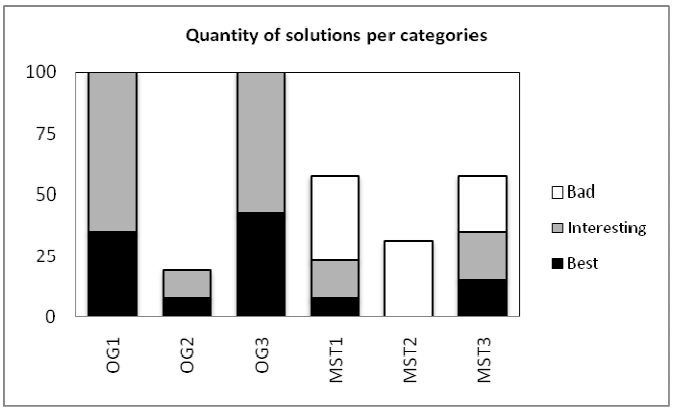


Figure 1: quantity of solutions per categories.

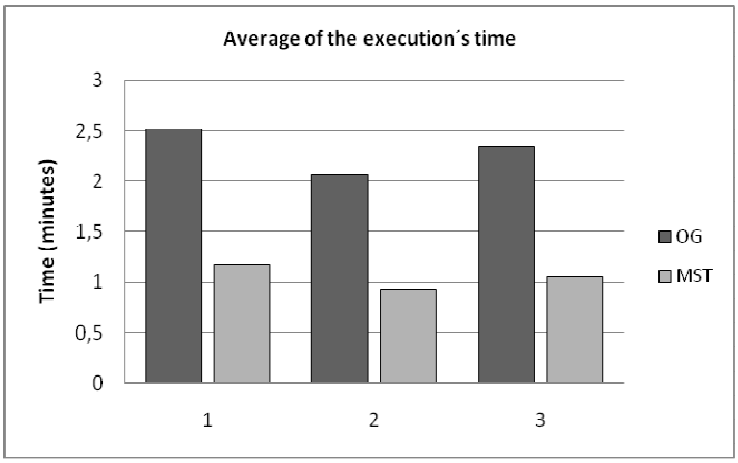


Figure 2: average of the execution´s time.

The CH3, however, had obtained more solutions of the best and interesting categories than others versions. Thus, the OG obtained best results, the execution´s time were higher than the MST algorithms, according the Figure 2.

In order to obtain more detailed results, the algorithms were submitted to a new experiment: The empirical probability distributions of the random variable time to target solutions (Aiex *et al.*, 2002), where the bests solutions found were used as target (hard target). This way, the algorithms were executed one hundred times over the selected instances. This paper presents the results of this experiment using the instances 4 and 13, selected by its quantity both vertexes and edges.

Figures 3, 4 and 5 shows that the OG3 was the best algorithm, where it reached the target in all the runs and in few seconds.

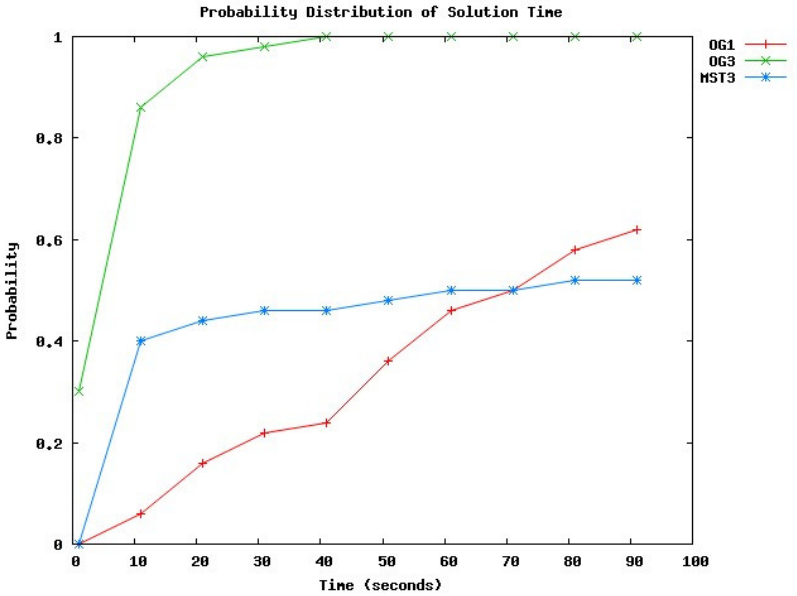


Figure 3: Empirical Probability- instance 4.

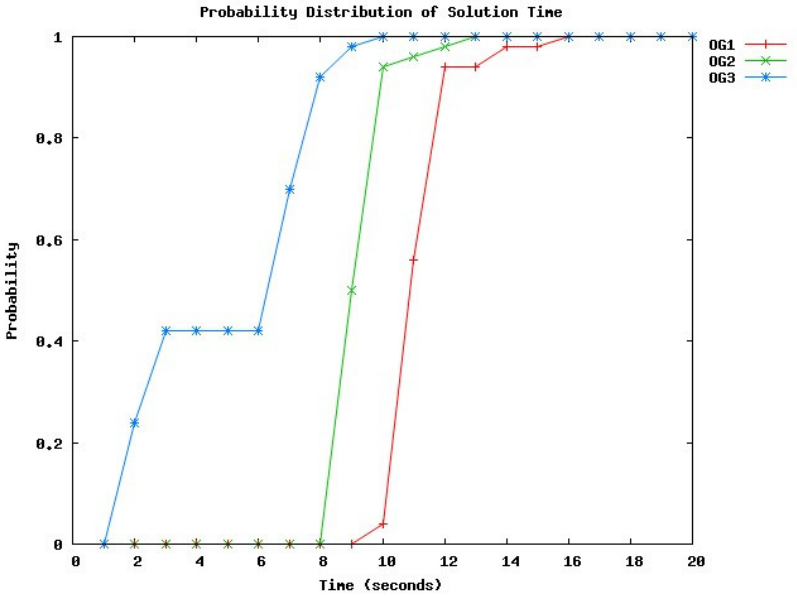


Figure 4: Empirical Probability OGs- instance 13.

In the experiment with the instance 4, the algorithms OG2, MST1 and MST2 did not reach the target (0%). Besides this, with the instance 13, only the algorithms MST1 and MST2 did not obtain 100%, with the probability of, 12% and 6%, respectively.

Moreover, these graphs can be analyzed in other way, such as: in the Figure 3 the OG3 reached the target more than 100% of runs before the 50 seconds of the execution time. The MST3 reached the target less than 60% of the runs in the end of the established time.

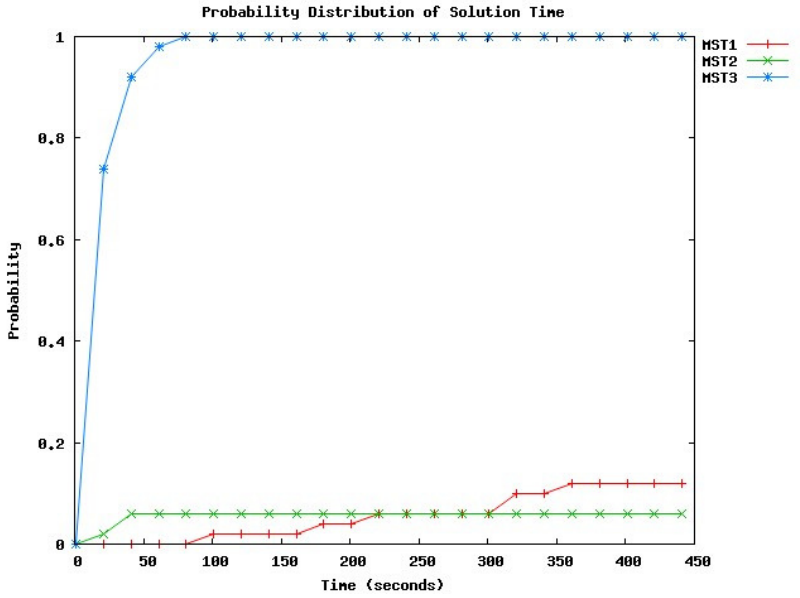

Figure 5: Empirical Probability MSTs- instance 13.

## 6 CONCLUSIONS AND FUTURE STEPS

Constructive heuristics were presented considering only the MST and the construction of initial solutions. Afterwards, those solutions were then refined through other procedures (some acts only over the MST and others act over the original graph). Therefore, two groups of algorithms were presented: OGs (based on the Original Graphs) and MST (based only on edges of MST).

In the first experiment, each algorithm was executed over the same set of instances ten times and the results showed that the OG algorithms obtained more expressive results, but with a higher processing time. In order to evaluate and obtain more detailed information, the second experiment was the empirical probability analyses.

Despite the algorithm had obeyed the stipulated processing time, the MSTs continued limited in the best local solution, while the OGs obtained new different solutions, that could not be formed through the only MST method.

The procedures that acted with the original submitted graph increase the possibilities of vertex migration and thus facilitated the formation of both valid as better quality solutions.

The computational results showed that the proposed OG algorithms are an interesting alternative to solve this problem, in which all this versions improve both the solution´s quality as the quantity of formation of valid solutions. Besides this, the OG3 represents the best algorithm version.

Based on the experiments and the searches, this paper proposes as future work new ways that can help to solve the problem, such as:

- *Penalization methods:* use a new fitness function with a penalization factor, which will increase the solutions value according of the level of the penalization.
- *Integer programming:* use the mathematical formulation proposed by Brito *et al.*, (2004), to analyze and to compare the results.
- *Path relinking:* propose a new local search procedure in order to integrate intensification and diversification in search for new best solutions. It consists in to explore trajectories between high quality solutions and was proposed by (Glover,

1996; Glover *et al.,* 2000).

- *Use others metaheuristics:* to developer and analyze the use of other metaheuristics, such as: ILS Iterated Local Search (ILS), Variable Neighborhood Search (VNS), Tabu Search, Greedy Randomized Adaptive Search Procedure (GRASP) or a hybrid heuristic version (Glover and Kochenberger, 2002).

## 7 ACKNOWLEDGMENTS

To all the teachers and students of the Computer Institute at UFF (http://www.ic.uff.br) and CAPES (http://www.capes.gov.br) for the financial support.

## REFERÊNCIAS

Aiex, R. M.; Resende, M. G. C.; Ribeiro, C. C. TTTPLOTS: A Perl program to create time-to-target plots, *Optimization Letters 1*, 2007.

Assunção , R. M; Lage, J. P.;Reis , A. E.. Análise de Conglomerados Espaciais Via Árvore Geradora Mínima. *Revista Brasileira de Estatística*, vol. 63, n. 220, 2002, 7-24.

Assunção, R. M.; Neves , M. C.; Câmara, G., Freitas , C. Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. *International Journal of Geographical Information Science 20*(7): 797-811, 2006.

Berkhin, P. Survey of Clustering Data Mining Techniques. Accrue Software, 2002.

Brito, J. A. M.; Montenegro , F. M. T.; Brito L. R.; Passini , M. M.. Uma formulação de programação inteira para o problema de criação de áreas de ponderação agregadas, *Anais do SOBRAPO*, 2004.

Dias, C. R.; & Ochi, L. S.. Efficient Evolutionary Algorithms for the Clustering Problems in Directed Graphs. Proc. of the IEEE Congress on Evolutionary Computation (IEEE-CEC), 983-988. Canberra, Austrália, 2003.

Dias, C. R. Algoritmos Evolutivos para o Problema de Clusterização de Grafos Orientados: desenvolvimento e análise experimental. 2004. 129 f. *Dissertação de Mestrado em Computação, Universidade Federal Fluminense*, Niterói, 2004.

Doval, D., Mancoridis, S. and Mitchell, B. S. Automatic Clustering of Software Systems using a Genetic Algorithm. *Proc. of the Int. Conf. on Software Tools and Engineering Practice*, pp. 73-81, 1999.

Freitas F. G.; Maia, C. L. B; Coutinho, D. P.; Campos, G. A. L.; Souza, J. T., Aplicação de Metaheurísticas em Problemas da Engenharia de Software: Revisão de Literatura, *Anais do II Congresso Tecnológico InfoBrasil* (InfoBrasil´2009), 2009.

Glover, F. Tabu search and adaptive memory programing: advances, applications and challenges. *Interfaces in Computer Science and Operations Research*, pp. 1-75, 1996

Glover, F. ; Kochenberger, G. A. Handbook of Metaheuristics. Kluwer Academic Publishers, 2002.

Glover, F.; Laguna, M.; Mart, R. Fundamentals of scatter search and pathrelinking. *Control Cybernetics*, pp. 653-684, 2000.

Goldberg, D. E. *Genetic Algorithms in search, optimization and machine learning*. Tuscaloosa: Addison-Wesley, 1989.

Han, J., e Kamber, M., *Data Mining: Concepts and Techniques*, 2 ed., Morgan Kaufmann, USA, 2005.

Holland, J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press,

Ann Arbor, 1975.

MacQueen, J. Some Methods for classification and Analysis of Multivariate Observations. *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*. 1. University of California Press, 1967.

Santos, H. G., Ochi, L. S., Marinho, E. H., Drummond, L. M. A. Combining an Evolutionary Algorithm with Data Mining to solve a Vehicle Routing Problem. *NEUROCOMPUTING Journal - ELSEVIER*, volume 70 (1-3), pp. 70-77, 2006.

Scheuerer, S. W. , R. A scatter search heuristic for the capacitated clustering problem. European *Journal of Operational Research vol. 169*, 2006.

Semaan, G.S. *Algoritmos Heurísticos para o Problema de Particionamento de Grafos com Restrições de Capacidade e Conexidade*. Dissertação de mestrado, UFF, 2010.

Semaan, G. S., Ochi, L.S., Brito, J. A. M. Um Algoritmo Evolutivo Híbrido Aplicado ao Problema de Clusterização em Grafos com Restrições de Capacidade e Conexidade. IX *Congresso Brasileiro de Redes Neurais / Inteligência Computacional (IX CBRN)*, Ouro Preto, 2009.

Semaan, G. S., Ochi, L. S., Brito, J. A. M.; Montenegro, F. An Efficient Evolutionary Algorithm for the Aggregated Weighting Areas Problem. *International Conference on Engineering Optimization*, 2008.

Shieh, H.M., May, M.D. Solving the Capacitated Clustering Problem with Genetic Algorithms. *Journal of the Chinese Institute of Industrial Engineers*, Vol. 18, 2001.