**EngOpt 2008 - International Conference on Engineering Optimization**
Rio de Janeiro, Brazil, 01 - 05 June 2008.

# An Efficient Evolutionary Algorithm for the Aggregated Weighting Areas Problem

**Gustavo Silva Semaan, Luiz Satoru Ochi**
gsemaan@ic.uff.br, satoru@ic.uff.br
Universidade Federal Fluminense
Niterói, RJ, Brasil

**José André Moura Brito, Flávio Montenegro**
jose.m.brito@ibge.gov.br, flavio.montenegro@ibge.gov.br
IBGE – Instituto Brasileiro de Geografia e Estatística
Rio de Janeiro, RJ, Brasil

**Abstract**

The present work describes an evolutionary algorithm for the Aggregated Weighting Areas (AWAs) problem. This problem consists in to form *K* partitions composed by Weighting Areas (WAs) of the demographic census satisfying contiguity, minimum total population and homogeneity criteria. The computational results show that the proposed hybrid evolutionary algorithm is a suitable alternative to solve the analyzed problem.

**Keywords:** Regionalization; Graph partitioning; Evolutionary Algorithm.

## 1. Introduction

The clustering problem means to group elements from a database in different smaller sets called clusters, aiming to maximize the similarity among elements from the same cluster and reduce as much as possible the similarities among elements from diverse clusters [1]. Considering a given set with *n* elements $X = \{x_1, .., x_n\}$, it must extract partitions from the set *X* in *K* different clusters $C_i$, respecting the following conditions (Figure 1).

$$C_i \neq \varnothing \qquad i = 1 \ldots k$$
$$C_i \cap C_j = \varnothing \qquad i, j = 1, \ldots k$$
$$C_i \cup \ldots \cup C_k = X$$

Figure 1. Conditions of the clustering.

Although a wide field of clusterization approaches use a previously defined quantity of clusters, some real applications can present a lack of previous knowledge about how many clusters should be used. The first case, in which the number of clusters is previously known, is called *k-clusterization* or, simply, clusterization problem (CP) being the number of viable solutions reached by applying the Eq.(1)., where a means Stirling of second type. If the optimum number of clusters is not well-defined or known, it will belong to the called *Automatic Clusterization Problems* (ACP). In this case the number of viable solutions will be given by Eq.(2). Both problems are classified as complete-NP although ACP is very difficult to solve due to its bigger number of alternative solutions.

$$N(x) = \left(\frac{1}{k!}\right) \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} (j)^n \tag{1}$$

$$N(x) = \sum_{k=1}^{n} \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n \tag{2}$$

The analysis of clusters is a fundamental techniques to experimental sciences in which the classification of elements into groups is desirable, as examples of these fields we can cite biology, medicine, economy, psychology, among others. In many applications, CP or ACP can be presented by graphs, considering a problem of partition graphs. This partition consists of grouping the vertex of the graphs in different subsets or clusters, according to their similarities or fitness [2,3,4].

As described by [5], evolutionary and genetic algorithms are widely used in artificial intelligence, inspired on the natural evolution theory and genetics, this field is also known as evolutionary computing. These algorithms try to simulate some aspects from Darwin´s Natural Selection Theory, being widely applied to many problems, previously considered complex.

According to [6] and appreciable quantity of geographical applications such as data analysis of Census, health and social-economical parameters are organized in big groups of spatial objects, being represented by areas. In some cases, it is desirable to group a great quantity of spatial objects in a small number of groups, in which, internally, the elements are homogeneous and take continuous regions of space. This procedure is called regionalization and its application to social-economical data was initially described and done by [7].

This work the objects were presented as vertexes in a graph and the proposed algorithm continuously partitionates the graphs accord to a specified quantity of clusters. The advantage of the graph-based technique is that the adjacent spatial position is essential part of

clustering process. The presented technique is basically divided in two steps: building of a Minimum Spanning Tree (MST) from the graph which represents the problem, and getting of sets of clusters by the algorithm, by partitioning of MST.

This work is divided in three sections. The section 2 presents, in details, the problem of creating Aggregated Weighting Areas (AWAs), which was the aim of this work. Section 3 describes the evolutionary algorithms proposed for the formation of APAs. Finishing this paper with some final comments the Section 4 presents the results found by applying the proposed algorithm as well as some additional comments.

## 2. Aggregated Weighting Areas Problem

This section contains a detailed description of the problem on AWAs formation. First of all, to make the understanding of the subject easier, some basic concepts on Weighting Areas (WAs) are presented, followed by the description of the main problem. To conclude the section, the modeling of the problem is presented, considering the basic concepts of the theory of graphs.

### 2.1 Definition of the problem

WA can be defined as a geographic unit formed by mutually excludent groups of censitary sectors, being theses censitary groups formed by set of domiciles. These WAs are used in order to estimate informations about the population. The size of the areas, by means of quantity of domiciles and/or population, can not be so small, to avoid lost of precision on the estimatives. Since WAs are defined considering this condition, they represent the more detailed geographic level of the operational database, being developed as a tool to solve the problem of demanding for informations in geographic levels smaller than municipalities [8,9].

For the formation of WAs, contiguity criteria are also considered, in a way that these areas are geographically immediate neighbors, and homogeneity, regarding a set of $p$ variables associated to populational and environmental known characteristics. These variables, which will be represented by $x^s$, $s = 1,..,p$, are also called indicators of pondering areas. Considering these indicators and using the Eq.(3), all distances $d_{ij}$ between $i$ and $j$ neighbors WAs are calculated. The distances $d_{ij}$ represent the homogeneity degree, i.e., the proximity among values from $p$ variables associated to all WAs to be aggregated.

$$d_{ij} = \sqrt{\sum_{s=1}^{p} (x_i^s - x_j^s)^2} \tag{3}$$

The AWAs are formed by mutually-excludent of WAs to create these areas two viability criteria must be respected:

(1) *contiguity*: the WAs aggregated in each one of the AWAs must be neighbors or, at least, must be possible to arrive a new WA $b$ starting from an WA $a$ [9].

(2) *Total associated to one of the variables*: For a given variable, such as the population of each area, the total of this variable in each AWA, considering all its respective WAs, must be bigger or, at least, equal to a previously defined total.

### 2.2 Problem Design

By considering the informations presented in the previous section, it is possible to observe that the problem of formation of aggregated areas should be mapped in a problem of conex and capacitated group formation, in which the restriction (1) implicates in conexity and the restriction (2) implicates in capability.

This way, for the development of every heuristic algoritm ou formulation for this problem is possible to associate relative informations to the contiguity of WAs, the total informations can be be associated to one of the $p$ variables, and the distances $d_{ij}$ can be related to a defined graph $G = (V,E)$. Each vertex $i \in V$ from the graph corresponds to a WA and contains the value associated to the variable which defines the total; moreover, if two WAs $i$ and $j$ are neighbors, it is possible to find an edge that shows the value for the distance $d_{ij}$.

Considering the conexity restriction of the graph G, one natural solution for the problem will consist of building a MST $T = (V, E^* \subset E)$ from $G$, respecting the smaller values of $d_{ij}$. Once one tree $T$ and a number $K$ of partitions (number of AWAs to be generated) are provided, it is possible to extract $K$ - 1 edges from $T$, defining, this way, a set of $K$ subtrees $T_j$, $j = 1..k$, which are also conex. Each one of these subtrees will be associated to one AWA.

The conexity property, observed in each one of the subtrees, allows the imediate execution of the contiguity restriction in each one of the AWAs. Thus, the solution for the problem will consist of partitioning $T$ in $K$ subtrees $T_j$ ($j$ = 1,.., $K$) associated to AWAs which satisfies the total restriction and results in the lower possible value for a fitness function.

Focusing on the same subject and approach we can cite [10,11]. The first described the creation of spatial conglomerates by successive partition of a MST associated to WA, being its procedure of construction similar to the focus of this paper.

In the case of partitioning, since the criteria of capability and contiguity had been satisfied, the authors proposed the following objective function Eq.(4), aiming to evaluate the quality of obtained AWAs.

$$f = \sum_{j=1}^{m}\sum_{i=1}^{n}(x_{ij} - \overline{x_j})^2 \quad \text{where } \overline{x_j} = \frac{\sum_{i=1}^{n} x_{ij}}{n} \quad (m \text{ variables, } n \text{ areas}) \tag{4}$$

This function represents the sum of squares of deviations in space of variables, related to the average of all the areas of the tree. For the formation of each pair of new groups, it is necessary to assess the value of Eq.(4), considering the removal of each one of the edge associated to previous groups. More smaller this sum becomes, better will be the solution.

Another report [11] proposed a complete formulation of integer programming which realizes the partitioning of the MST by the minimization of the sums of distances among all WAs which belong to the same AWA.

## 3. Proposed Algorithm

This section presents the algorithm proposed for the extractions of AWAs. As it was described in the last section, such problem can be designed by partition graph problem. The first step builds a MST from Kruskal's algorithm. The second step is to partition the MST with the proposed algorithm.

### 3.1 Evolutionary Algorithms

Evolutionary algorithms (EAs) are search methods based on biological theories of evolution, like the Darwinian Theory, where the concepts of survival of the best fitted individual and hereditary transference of genetic characteristics are exploited. EAs usually involve simulating mechanisms of selection of the best fitted individuals in a population, reproduction and crossover between pairs of their chromosomes, and mutation over a little part of the overall genetic charge, in order to create a new population. Along successive iterations, each new population, or generation, is expected to be best fitted as a whole than the previous one.

### 3.2 Proposed Algorithm

The proposed algorithm is related to evolutionary heuristics, which has the objective of partitioning the MST extracted from a graph submitted under *k* clusters. This way, aiming to generate *K* partitions it become necessary to remove *K* - 1 edges from *T*, considering that for each removal, $T_k$ will be divided in the subtrees $T_k^1$ e $T_k^2$.

Hierarchical division strategy was carried out, in which, initially, all the WAs belong to the same cluster. In each one of the iterations, one edge is removed from a cluster, being divided in two ones.

The selection of the cluster to be partitionate will follow the higher value of Eq.(4). On the other hand, the selection of the edge to be removed is a greedy procedure, it means that all the possibilities of edge removal from the selected cluster must be executed in order to minimize Eq.(4) of the solution. Considering that $S_l^T$ owns *l*, the removed edge from *T*, the Eq.(5) shows how to evaluate the division of the selected cluster.

$$f1(S_l^T) = fT_k - (f T_k^1 + f T_k^2) \tag{5}$$

The objective is to reach the best partial solution for each interaction. Figure 2 presents the steps of the algorithm that create solutions through by successive removal of edges.

(1) Start graph $G^* = T_0$ where $T_0$=MST.
(2) Search edge with highest Eq.(5) in $T_0$.
(3) While #($G^*$) < *K*.
(4) Select cluster with highest Eq.(4).
(5) Select the edge with highest Eq.(5).
(6) To divide selected cluster in two new subtrees and update $G^*$.
(7) End while

Figure 2. Greedy algorithm for to create solutions.

Although the greedy procedure search presented by the previous algorithm has high operational cost, it was applied on the building of the initial solution for the proposed algorithm. The purpose is to turn this building a semi-greedy process with randomness α given

as parameter. Moreover, a local search procedure and the perturbation caused by mutation will help on search best solutions. Figure 3 presents the steps of the proposed algorithm:

```
(1)  Extract MST from G* using Kruskal's algorithm.
(2)  Build initial population.
(3)  Calculate fitness function for each solution.
(4)  While not ( StopCriteria() )
(5)     Elitism()
(6)     LocalSearch()
(7)     Mutation_Migrate()
(8)     Mutation_Union()
(9)     Recalculate fitness function for each solution that was modified.
(10) End while.
```

Figure 3. Proposed evolutionary algorithm.

3.3 Representation of the Solution

According to [12], a good presentation for the problem is extremely important to the performance of the algorithm and it must be decisive for a fast convergence and quality of the obtained solutions.

The structure used for the representation of the adopted solution was a *group-number* in which, according to [13], the index of vector represents the vertex of the graph and its content represents the cluster to which the vertex belongs. The solution presented by Figure 4 indicates that the vertex of the graph were divided in three clusters, in which the cluster number 1 owns the vertexes 1, 3, 5 and 7, cluster 2 owns vertexes 2 and 6, and vertexes 4 and 8 belong to cluster 3.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 3 | 1 | 2 | 1 | 3 |

Figure 4. Representation of the one solution

3.4 Construction of the Initial Solution

For the construction of initial solutions the algorithm presented in Figure 2 was used, but, in this case, one random factor, submitted as α factor (positive integer values) to the algorithm, was also applied. The difference is that in the presented algorithm the edge with bigger Eq.(5) is removed while in the version implemented to the constructive procedure of the algorithm the biggest edges are related and randomly one of them is excluded. The vertexes of the removed edges are stored in order to allow the execution of other procedures of search to best solutions or perturbations in the solution of the mutation procedure, for example.

Although the restriction of minimum capability exists, i. e., a inferior limit for the sum of determined variable submitted to the problem, the Constructor Procedure only signalize the clusters with penalties. The Local Search Procedure will work in possible migrations of vertexes among clusters, aiming to correct eventual penalties.

3.5 Local Search Procedure

The local search procedure acts on the attempts of eliminating penalties, independently of the improvement of the solution. For this, it uses the relation of the vertexes of removed edges and the relation of the clusters with penalties. Figure 5 presents the steps of the local search procedure:

```
(1)  For each RemovedEdge E(S,T)
(2)     If ( Clusters(S) xor Clusters(T) ) Penalized
(3)        If (Vertex can be migrate)
(4)           MigrateVertex()
(5)        End If
(6)     End If
(7)  End for
```

Figure 5. Local search procedure.

For each removed edge it is necessary to identify to which cluster its vertexes belong. The next step is the verification of the necessity of migration of vertex among the clusters, what must occur only if one of them present penalties. Moreover, it is necessary to check if the vertex can be migrate, what happens depending on its connectivity degree and location. The possibilities are:

- *Degree 1*: Since the cluster become empty, migration can not occur.
- *Degree 2*: Migration can occur.
- *Degree 3 or higher*: If connected to more than one vertex that belong to the same cluster, migration can not occur.

In the case of confirmation of the possibility of migration, the vertex of the removed edge which belongs to the non-penalized cluster will change to the penalized one. This operation is not a warranty of cancelling penalties, neither that the cluster from which the vertex has left will be penalized, nor the improvement of fitness of the solution. The reapplication, on the other hand, can acts generating one cascade effect in the correction of cluster penalties. Figure 6 presents an example of vertex migration, considering the minimum capacity of 6 units.

| Vertex | Variable |
|---|---|
| A | 5 |
| B | 2 |
| C | 3 |
| D | 4 |
| E | 5 |
| F | 6 |
| G | 7 |

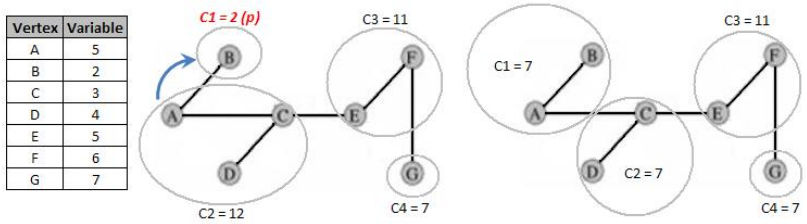


Figure 5. Example of migrate vertex among clusters. Cluster *C1* receive vertex *A* from cluster *C2*. *C1* penalty was cancelled.

3.6 Mutation

This work had exploited two different versions of mutation. The first version, an algorithm for vertex migration among clusters similar to the local search algorithm was implemented but, in this case, it was done without the purposed of penalties elimination. This way, since one probability is submitted as parameter to the algorithm, one of the vertexes of each removed edge can migrated to the neighbor cluster. For the execution of this procedure it becomes necessary to check if the vertex should be moved, respecting the conditions presented on the local search procedure section.

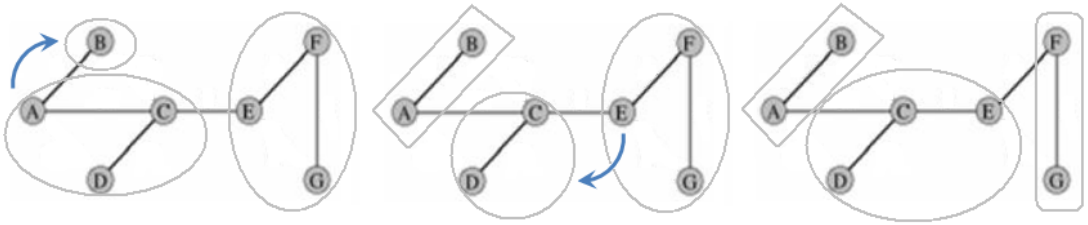

Figure 7. Example of vertex migration among clusters based on the first version of mutation procedure.

The other version, also dependent on the submitted probability, works with removed edges in cluster generation, but, in this situation, the clusters to which the vertexes of the removed edges belong can become one unique cluster , being , after this, partitionated by using a semi-greedy constructive procedure.

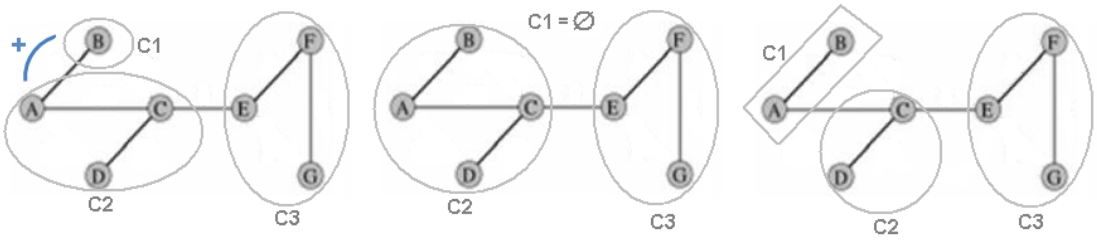


Figure 8. Example of fusion and division, according to the described for the second version.

3.7 Elitism

The algorithm applies the elitism technique and stores the best found solutions, with and without penalties. At the beginning of each iteraction, one of these solutions is inserted to the population in order to improve quality by using the procedure of local search and mutations. When there is no solution without penality, 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 is possible, the best solution without penalties.

**4. Computational Results**

4.1 Applied Instances

Aiming to evaluate the proposed algorithms a set of instances was used, these instances were obtained from a sample of Brazilian Demographic Census 2000. Actually, for these Federation Units it was applied the defined WA from the Census.

This way, for each unit, two files were used: the first file contains the information of neighbors among the WA associated to the UFs, the second contains in each line the identification of WA and some associated variables as total of houses, total of domiciles, total of person, sum of total salaries, sum of time of instruction or study, sum of salary per-capita, average time of instruction or study of the responsible.

The first file is used to determine the relationship among vertexes of graph *G* and its neighbors, from which the MST will be constructed, i.e.. Being these variables, after suitable normalization, applied to the determination of distances $d_{ij}$ in edges of this graph.

A variable *total of person* was noticed and also used on the formation process of groups as capacity variable, i.e.. It was previously defined that the sum of values for the variable *total of person* associated to the WAs which compose each AWA must not be smaller than a defined value *P*.

4.2 Computational Results

This section presents a set of computational results obtained from the application of the evolutionary algorithm described. To evaluate the algorithm performance, it was used three instances related to Federation Units data from the Demographic Brazilian Census 2000. For each of these instances, there is a set of WAs and their respective attributes.

In Table 1, there is some information related to each of the three considered instances. The first column in this table identifies the instance number, while the second one corresponds to the quantity of WAs in each instance, which means the number of vertexes in the graph. Finally, the third column presents the quantity of neighborhoods among weighting areas, which corresponds to the number of edges in the graph associated to the instance.

Table 1. Instances (graphs).

| Instance | Vertexes | Edges |
|---|---|---|
| 1 | 21 | 58 |
| 2 | 14 | 46 |
| 3 | 73 | 350 |

Table 2 describes and enumerates all the considered variations for the evolutionary algorithm, taking into account all the eight possible combinations of the Local Search and the two Mutation Procedures presented in Section 3.

Table 2. Configurations.

| | Configuration | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Local Search | X | | | X | X | | X | |
| Mutation Migrate | | X | | X | | X | X | |
| Mutation Union | | | X | | X | X | X | |

Finally, Table 3 presents the values of the fitness function (Eq. (4)) obtained for the three instances, according to the application of the eight configurations (variations) of the algorithm and taking into account the number of clusters varying from two to four.

Table 3. Values of the fitness function.

| | | **Instance 1** | | | **Instance 2** | | | **Instance 3** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | **Clusters** | | | **Clusters** | | | **Clusters** | | |
| | | **2** | **3** | **4** | **2** | **3** | **4** | **2** | **3** | **4** |
| **Configuration** | **1** | 31.82 | 30.84 | 29.44 | 4.76 | 4.51 | 4.24 | 703.81 | 694.09 | 693.38 |
| | **2** | 31.59 | 30.53 | 29.44 | 4.65 | 4.25 | 3.93 | 703.81 | 693.67 | 693.38 |
| | **3** | 31.59 | 29.66 | 29.16 | 4.56 | 4.51 | 4.18 | 701.88 | 694.09 | 693.38 |
| | **4** | 31.59 | 30.53 | 29.44 | 4.65 | 4.25 | 3.93 | 703.81 | 693.67 | 693.38 |
| | **5** | 31.59 | 29.66 | 29.16 | 4.56 | 4.51 | 4.18 | 701.88 | 694.09 | 693.38 |
| | **6** | 31.59 | 29.56 | 29.41 | 4.76 | 4.42 | 3.93 | 702.51 | 693.67 | 693.38 |
| | **7** | 31.11 | 29.56 | 29.41 | 4.76 | 4.42 | 3.93 | 702.51 | 693.67 | 693.38 |
| | **8** | 31.82 | 30.84 | 29.44 | 4.76 | 4.51 | 4.24 | 703.81 | 694.09 | 693.38 |

From the analysis of these tables, it was possible to make the following considerations:

- For the Instance 1, and considering the overall analysis of the clusters, the best solutions were obtained by applying the configurations 3, 5, 6 and 7. Particularly, for two clusters, the best solution (31.11) was obtained by using the configuration 7 (that is, the complete configuration, which combines all the three procedures: the local search and the two mutations routines). For three clusters, the algorithm performed better for the configurations 6 and 7 (29.56). Finally, for four clusters, the best solution (29.16) was reached using the configurations 3 and 5.
- Still considering Instance 1, the average reduction (taking into account the reductions for all the eight configurations) of the objective function when increasing the number of clusters from two to three was 4.8%. When increasing that number from three to four, the objective function reduced itself, in average, 2.67%.
- For the Instance 2, and considering the overall analysis of the clusters, the best solutions were obtained by applying the configurations 2, 3, 4 and 5. Particularly, for two clusters, the best solutions (4.56) were provided by using configurations 3 and 5. For three and four clusters, the configurations 2 and 4 performed better, providing the values 4.25 and 3.93.
- The average reduction of the objective function, for the Instance 2, when increasing the number of clusters from two to three was 5.94%. When increasing that number from three to four clusters, the objective function reduced itself, in average, 8.72%. That is, for this instance, the more significant reduction was obtained when using four clusters.
- For the Instance 3, considering again the overall analysis of the clusters, the best solutions were provided by applying the configurations 3, 5, 6 and 7. For two clusters, the best solution (701.9) was reached using the configurations 3 and 5. For three clusters, the algorithm performed better for the configurations 6 and 7 (693.7), while, for four clusters, all the configurations performed equally (resulting in the value 693.4).
- The reduction, for Instance 3, of the objective function when increasing the number of clusters from two to three was in average 1.31%. When increasing that number from three to four clusters, the average reduction of the objective function was very small (0.07%).
- Considering the average reduction of the objective function, in terms of percent, it could be seen that the more significant reductions were obtained for the Instance 2.

4.3 Future Steps

Since the graph submitted to the proposed algorithm must be conex and with no circuits, it is possible to work with any graph that has these properties, without the need of using the Kruskal's algorithm to build a MST. In this case diversity population techniques can be applied to search best solutions on the several graphs submitted to the problem.
The paper [14] presented the SKATER Algorithm (Spatial 'K'luster Analysis by Tree Edge Removal) and proposed an heuristic procedure that speeds up MST partition. This heuristic can be an alternative to the constructive procedure of the proposed algorithm in this paper.

## References

1. Berkhin, P. Survey of Clustering Data Mining Techniques. Accrue Software, 2002
2. Hartuv, E. and Shamir, R. A Clustering Algorithm based on Graph Connectivity. Technical Report, Tel Aviv University, Dept. of Computer Science, 1999
3. Trindade, A.R. and Ochi, L.S. Um algoritmo evolutivo híbrido para a formação de células de manufatura em sistemas de produção. Abstracts in Operational Research / Statistical Theory and Methods Abstracts) vol. 26(2), 2006, 255-294.
4. Dias, C. R. and Ochi, L. S. Efficient Evolutionary Algorithms for the Clustering Problem in Directed Graphs, Proceedings of the 2003 IEEE Congress on Evolutionary Computation, v.1, 2003, 983–988.
5. Santos, H. G. et al. Combining an Evolutionary Algorithm with Data Mining to solve a Vehicle Routing Problem. Neurocomputing Journal - Elsevier, volume 70 (1-3), 2006, 70-77.
6. Assunção R. M. et al. Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. IJGIS (7): 797-811.
7. Openshaw, S. A geographical solution to scale and aggregation problems in regionbuilding, partitioning and spatial modelling. Transactions of the Institute of British Geographers (New Series), 2, 1977, 459–472.
8. Censo Demográfico 2000. Primeiros Resultados da Amostra, Parte I, 2001, IBGE/CDDI.
9. Silva A. N. et al. Processamento das Áreas de Expansão e Disseminação da Amostra no Censo Demográfico 2000, Textos para Discussão, número 17, 2004, IBGE/DPE/COMEQ.
10. Assunção, R. M. et al. Análise de Conglomerados Espaciais Via Árvore Geradora Mínima. Revista Brasileira de Estatística, vol. 63, n. 220, 2002, 7-24.
11. Brito, J. A. M. et al. Uma formulação de programação inteira para o problema de criação de áreas de ponderação agregadas, Anais do SOBRAPO, 2004.
12. Doval, D. et al. Automatic Clustering of Software Systems using a Genetic Algorithm. Proc. of the Int. Conf. on Software Tools and Engineering Practice, 1999, 73-81.
13. Cole, R. M. Clustering with Genetic Algorithms. Master's thesis, Department of Computer Science, University of Western Australia, 1998.
14. Assunção, R. M.; Neves, M. C.; Câmara, G.; Freitas, C. C. Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. International Journal of Geographical Information Science, v. 20, 2006, 797-811.