UM ALGORITMO EVOLUTIVO HÍBRIDO APLICADO AO PROBLEMA DE CLUSTERIZAÇÃO EM GRAFOS COM RESTRIÇÕES DE CAPACIDADE E CONTIGUIDADE

Gustavo Silva Semaan[1], Luiz Satoru Ochi[1] e José André Moura Brito[2]

[1]Instituto de Computação - Universidade Federal Fluminense.
Rua Passo da Pátria 156, Bloco E – 3° andar, São Domingos. CEP.: 24210-240 Niterói, RJ, Brasil.

[2]Diretoria de Pesquisas - IBGE – Instituto Brasileiro de Geografia e Estatística.
Av. República do Chile 500 - 10º andar, Centro. CEP.: 20031-170 Rio de Janeiro, RJ, Brasil.
gsemaan@ic.uff.br, satoru@ic.uff.br, jose.m.brito@ibge.gov.br

Resumo – Este trabalho descreve uma nova heurística evolutiva híbrida para a resolução de um problema de clusterização em grafos com restrições de capacidade e contiguidade. Neste problema deve-se particionar um grafo G = (V,E) em k subgrafos conexos de forma que a soma dos valores (capacidades) associados aos vértices sejam maior ou igual a um valor pré-fixado e os vértices mais similares, segundo uma função objetivo, estejam em um mesmo cluster. No final do trabalho, é apresentado um conjunto de resultados computacionais obtidos a partir da resolução um problema real, mostrando que o algoritmo proposto representa uma boa alternativa para a resolução do problema analisado.

Palavras-chave – Metaheurística; Algoritmo Evolutivo; Particionamento de Grafos; Regionalização.

1. Introdução

O problema geral de clusterização (PC) está associado a diversas aplicações reais nos campos da biologia, da medicina, da economia e da estatística, dentre outros. Neste problema, deve-se agrupar os *n* elementos de uma base de dados em *k* subconjuntos disjuntos $C_i, i = 1,...,k$ denominados clusters, de forma a maximizar a similaridade entre os elementos de um mesmo cluster e minimizar a similaridade entre elementos de clusters distintos [3], considerando uma particular função objetivo e respeitando as restrições apresentadas abaixo:

$$\bigcup_{i=1}^{k} C_i = X \qquad (1)$$

$$C_i \neq \emptyset, 1 \leq i \leq k \qquad (2)$$

$$C_i \cap C_j = \emptyset, 1 \leq i, j \leq k, i \neq j \qquad (3)$$

Em geral, a obtenção de um ótimo global para tal problema é uma tarefa muito difícil, tendo em vista o seu aspecto combinatório. Na verdade, caso seja aplicado um processo de busca exaustiva para garantir a obtenção da solução global, seria necessário enumerar todas as soluções, ou seja, todas as possibilidades de combinação dos n objetos em k grupos. O número de possibilidades, neste caso, está associado ao número de Stirling de segundo tipo, dado pela Eq.(4).

$$\frac{1}{k!}\sum_{j=0}^{k}(-1)^{k-j}\binom{k}{j}j^n \qquad (4)$$

Caso se tenha, por exemplo, n = 16 objetos a serem alocados em 2 grupos, o número de soluções a serem consideradas é de 32.767 . Para o caso de 3 grupos, este número cresce para 7.141.686 soluções possíveis. Considerando-se um número n maior de objetos, estes valores crescem exponencialmente.

As variantes deste problema, quais sejam: o problema de clusterização automática, o problema de clusterização capacitado e o problema de clusterização com conexidade em grafos [7], apresentam igual ou maior dificuldade de resolução.

O presente trabalho aborda uma aplicação real que está associada a um problema de clusterização em grafos com restrições de capacidade e de conexidade.

Em particular, é possível encontrar soluções de boa qualidade para este problema utilizando-se técnicas de computação intensiva (mas não exaustiva), disponíveis na literatura, as custas de um maior tempo de processamento. Uma possibilidade é aplicação de algoritmos evolucionários e algoritmos genéticos.

De acordo com [9], os algoritmos evolucionários e algoritmos genéticos são muito utilizados na área de inteligência artificial, sendo inspirados na teoria da evolução natural e genética, conhecidas como computação evolucionária. Estes algoritmos tentam simular alguns aspectos da teoria da seleção natural de Darwin e são utilizados em muitos problemas considerados complexos.

O presente trabalho está dividido em 3 seções. Na seção 2 temos uma descrição detalhada sobre o problema de criação de Áreas de Ponderação Agregadas (APAs) [4,10], que foi o estudo de caso deste trabalho. Na seção 3 é feita uma apresentação concisa de algoritmos evolutivos, seguida de uma descrição detalhada do algoritmo evolutivo proposto para a formação de APAs. Concluindo o trabalho, temos na seção 4, um conjunto de resultados computacionais e análises obtidas a partir da aplicação algoritmo proposto.

2. Estudo de Caso: Problema de Formação de APAs

Segundo [1] uma razoável quantidade de aplicações reais, como a regionalização geográfica e a divisão de redes de telecomunicações, podem ser mapeadas em um problema de clusterização com restrições de contiguidade espacial. Neste tipo de problema, deseja-se particionar um conjunto de n objetos que compõe uma base dados em k clusters que sejam homogêneos internamente e de forma que objetos em um mesmo cluster sejam contíguos. Dentre as aplicações reais que incorporam o conceito de regionalização geográfica, podemos citar os censos demográficos e pesquisas socioeconômicas [8]. No censo demográfico os objetos podem estar associados a domicílios, setores censitários, áreas de ponderação (agregados de setores), municípios, etc.

A presente seção contém uma descrição detalhada de uma particular aplicação real de regionalização, conhecida como o problema de formação de APAs. Inicialmente, para facilitar o entendimento do problema, são apresentados conceitos básicos sobre áreas de ponderação, passando-se, em seguida, à descrição do problema em si. Concluindo a seção, descreve-se uma possível modelagem para o problema, considerando alguns conceitos básicos de teoria dos grafos.

2.1. Definição do Problema

A área de ponderação (APOND) corresponde a uma unidade geográfica formada por agrupamentos mutuamente exclusivos de setores censitários, sendo estes, por sua vez, formados por conjuntos de domicílios. Essas áreas são utilizadas para se estimar informações para a população. Desta forma, tamanho destas áreas, em termos de número de domicílios e/ou de população, não pode ser muito reduzido sob pena de perda de precisão das estimativas. As APONDs também representam os níveis geográficos mais detalhados da base operacional, sendo definidas de forma a atender às demandas por informações em níveis geográficos menores que os municípios [10]. Para a formação das APONDs, são considerados ainda critérios de contiguidade, de forma que estas áreas sejam constituídas por conjuntos de setores geograficamente limítrofes, e de homogeneidade, segundo um conjunto de p variáveis associadas às características populacionais e de infra-estrutura conhecidas. Estas variáveis, que representaremos por x^s, $s = 1,..,p$, são chamadas *indicadores de áreas de ponderação*. Considerando esses indicadores, são calculadas todas as distâncias d_{ij} entre APONDs i e j vizinhas, segundo a Eq.(5). As distâncias d_{ij} representam o grau de homogeneidade entre as APONDs que serão agregadas.

As APAs são formadas por agrupamentos mutuamente exclusivos de APONDs, sendo respeitados dois critérios de viabilidade para a sua criação:

- **Contiguidade:** As APONDs agregadas em cada uma das APAs devem ser vizinhas, ou deve ser possível sair de uma APOND ***A*** e chegar em uma APOND ***B***, ambas em uma mesma APA, passando apenas por outras APONDs que também estejam nesta mesma APA [4,10].
- **Total associado a uma das variáveis:** Fornecida uma variável, por exemplo, a população de cada área, o somatório dessa variável em cada APA deve satisfazer um limite inferior pré-estabelecido.

$$d_{ij} = \sqrt{\sum_{s=1}^{p} (x_i^s - x_j^s)^2} \quad (5)$$

2.2. Modelagem do Problema

A partir das informações da seção anterior, pode-se observar que o problema de formação de áreas de ponderação agregadas pode ser mapeado em um problema de agrupamento em grafos com as restrições de conexidade (contiguidade) e capacitado (total de determinada variável).

Desta forma é possível associar tanto as informações relativas à contiguidade das APONDs quanto as informações das capacidades a um grafo $G = (V, E)$. Cada vértice $i \in V$ do grafo corresponde a uma APOND e contém o valor associado à variável que determina sua capacidade. Além disso, se duas APONDs *i* e *j* são vizinhas, existe uma aresta *e = (i, j)* tal que esta contém o valor da distância d_{ij}.

Considerando a restrição de conexidade do grafo *G*, uma solução natural para o problema consistirá em construir uma árvore geradora mínima $T = (V, E^* \subset E)$ obtida a partir de *G*, considerando os menores valores de d_{ij}. Fornecida a árvore *T* e um número *k* de *clusters* (número de APAs que devem ser formadas), retira-se $(k - 1)$ arestas de *T*, definindo um conjunto de *k* subárvores T_j, $j = 1,...,k$, que também são conexas. Cada uma destas subárvores corresponderá a uma possível APA.

A propriedade de conexidade, observada para cada uma das subárvores, possibilita o cumprimento imediato da restrição de contiguidade em cada uma das APAs. Desta forma, uma possível abordagem para solução do problema consistirá, então, em particionar *T* em *k* subárvores T_j, $j = 1,.., K$, associadas às APAs, que satisfaçam a restrição de capacidade e que resulte no menor valor possível de uma particular função objetivo que mede o grau de similaridade nas APAs.

Considerando a mesma abordagem, podemos destacar os trabalhos [1,4]. O primeiro trabalho trata da criação de conglomerados espaciais através de uma partição sucessiva de uma árvore geradora mínima associada às áreas de ponderação.

No caso do particionamento, de forma a avaliar a qualidade das APAs obtidas, considerando que os critérios de capacidade e de contiguidade sejam respeitados, os autores propuseram a seguinte função objetivo (6).

$$f = \sum_{j=1}^{m} \sum_{i=1}^{n} (x_{ij} - \overline{x_j})^2 \quad \textit{em que} \quad \overline{x_j} = \sum_{i=1}^{n} x_{ij} / n \quad \textit{para m variáveis, n áreas.} \quad (6)$$

Para a formação de cada dois novos agrupamentos (APAs), avalia-se o valor de *f*, considerando a remoção de cada uma das arestas associadas aos agrupamentos anteriores. Quanto menor o valor de ***f***, melhor será a solução.

Já o trabalho [4], propõe uma formulação de programação inteira que realiza o particionamento da *AGM*, com uma função objetivo que minimiza (por APA) a soma das distâncias d_{ij} entre todas as áreas de ponderação pertencentes a uma mesma APA.

3. Algoritmo Proposto

O algoritmo proposto no presente trabalho constrói a uma árvore geradora mínima *T* (***AGM***), mediante a aplicação do algoritmo de *Kruskal*. Tal árvore é construída a partir de um grafo ***G*** que contém as informações das APONDs. Em seguida, para gerar as *K* APAs (clusters) é necessário remover $(k-1)$ arestas de *T*. Ou seja, em cada iteração j $(j = 1,...,k-1)$, uma aresta e_j é removida de uma subárvore T_{j-1} (obtida na iteração anterior), produzindo duas novas subárvores (dois clusters) T_j^1 e T_j^2 $(j = 1,...,k-1)$. Tal procedimento corresponde a uma estratégia de divisão hierárquica, na qual inicialmente todas as APONDs pertencem a um único *cluster*.

O critério de seleção da subárvore (*cluster*) a ser particionada foi baseado no maior valor obtido para Eq.(6), ou seja, a cada iteração selecionamos a subárvore (c*luster)* que possuir o maior desvio quadrático médio dos nós (APONDs) que compõe o cluster, considerando as p variáveis das APONDs.

Uma vez selecionada a subárvore, a escolha da aresta a ser removida consiste em um procedimento guloso, baseado no princípio da força bruta. Ou seja, todas possibilidades de remoção de arestas no cluster selecionado são executadas, objetivando a minimização da função objetivo f (Eq.(7)).

Considerando S_e^T como o conjunto formado por duas novas subárvores T_j^1 e T_j^2, obtidas a partir da remoção de uma aresta e_j de uma subárvore T_j, definida na iteração anterior, *j = 1, .., k – 1*, temos que $f(S_e^T)$ representa o custo definido pela diferença entre o cluster definido por T_j e a soma dos custos dos dois novos clusters.

$$f(S_e^T) = f_{T_j} - (f_{T_j^1} + f_{T_j^2}) \qquad (7)$$

O objetivo é produzir a melhor solução parcial a cada iteração. Os passos para o algoritmo básico de construção da solução inicial, através da remoção de arestas, são apresentados na Figura 1:

(1) Inicializar grafo $G^* = (T_0)$ onde T_0=AGM.
(2) Defina o número de arestas selecionadas j = 0
(3) Remova de ***G**** a aresta ***e*** que produz o menor valor de Eq.(7),definindo duas subárvores e faça j=1
(4) **Enquanto** (j < *k-1*) **Faça**
    Remova de ***Tj*** a aresta com o menor valor (Eq. (7)).
(5) Atualize *G** e faça j=j+1.
(6) **Fim-Enquanto**

Figura 1 - Algoritmo para construção de uma solução.

Tendo em vista que as partições produzidas a partir da AGM têm qualidade apenas razoável (valor da função objetivo), foi proposto um algoritmo evolutivo que incorpora os procedimentos de busca local, cruzamento e a mutação. Com a implementação destes procedimentos é possível melhorar a qualidade das soluções (*clusters*). A Figura 2 apresenta os passos do algoritmo proposto.

(1) Extração da AGM de *G** (Algoritmo de Kruskal).
(2) Construção da população inicial.
(3) Calcular função de Aptidão para as soluções.
(4) **Enquanto** (não atende ao Critério de Parada) **Faça**
(5) Elitismo
(6) Busca Local
(7) Mutação
(8) Cruzamento
(9) Recalcular Aptidão das soluções.
(10) **Fim-Enquanto.**

Figura 2 - Algoritmo Evolutivo Proposto.

3.2. Representação da Solução

Antes da aplicação do algoritmo é necessária a definição de uma estrutura para representação das soluções. Segundo [6], uma boa representação da solução é extremamente importante, podendo ser um fator determinante na performance do algoritmo, ou seja, para a rápida convergência e qualidade das soluções obtidas pelo mesmo.

Neste problema, cada solução do algoritmo corresponde a um vetor de n posições, no qual cada índice i corresponde a um vértice v_i do grafo e cada posição deste vetor pode assumir um valor entre 1 e k, ou seja, o cluster (APA) ao qual o vértice pertence (APOND será alocada).

3.3. Construção da Solução Inicial

Foram utilizadas duas versões para construção das soluções iniciais. A primeira versão consiste no algoritmo ilustrado na Figura 2, considerando, porém, que para a construção de duas novas subárvores T_j^1 e T_j^2 (obtidas de T_j), são selecionadas inicialmente as α maiores arestas de uma subárvore T_j, que ao serem removidas produzem o maior ganho em relação a função da eq(7). Observando que em cada iteração o parâmetro α assume um valor inteiro entre $\alpha_{\min} = 1$ e $\alpha_{\max} = \min(6, |e_{T_j}|)$ (e_{T_j} é número de arestas de T_j). Após a seleção das α arestas, a escolha da aresta a ser removida é feita de forma aleatória. Tal estratégia, permite a construção de soluções (partições) diferentes.

Os vértices de cada aresta removida são armazenados, visando que a aplicação de outros procedimentos possam trabalhar em busca de soluções melhores ou mesmo para perturbar a solução através da aplicação do procedimento de mutação, por exemplo.

Embora exista a restrição de capacidade mínima, ou seja, um limite inferior para a soma de determinada variável associada às APONDs, o procedimento construtivo apenas sinaliza os clusters que estão com penalidades (onde a restrição não é satisfeita). O procedimento de busca local trabalhará com possíveis inclusões ou remoções (migrações) de vértices em cada um dos clusters (subárvores), de forma a sanar as eventuais penalidades e tentar reduzir o valor da função objetivo.

A segunda versão de procedimento de construção trabalha na tentativa de formar *clusters* que atendam à restrição de capacidade mínima. O primeiro passo é a definição de dois vértices como sendo extremidades de uma árvore principal (vértices A e B). Em seguida, determina-se os vértices que pertencerão a essa árvore principal, ou seja, os vértices que estão no caminho existente entre A e B na *AGM* , mediante a aplicação da Eq.(8). Todos os vértices ***X*** que satisfazem a essa equação pertencem a árvore principal. A função *Dist(s,t)* ,utilizada na Eq.(8), retorna a distância entre os vértices *s* e *t* conforme o caminho existente na AGM. Os demais vértices ficarão agrupados ao vértice pertencente a árvore principal mais próximos a eles. Desta forma, o grafo ficará semelhante a um "cordão" e os vértices pertencentes a árvore principal possuirão o somatório das capacidades dos vértices a eles agrupados.

$$\text{Dist}(A,B) - (\text{Dist}(A, x) + \text{Dist}(B, x)) = 0 \qquad (8)$$

O próximo passo é percorrer a árvore principal obtida anteriormente formando clusters sem penalidade. Para isso, a partir de um dos vértices das extremidades, o procedimento adiciona o vértice da árvore principal mais próximo e verifica a restrição de capacidade. Quando a restrição de capacidade é satisfeita, o vértice mais próximo ao último vértice adicionado ao cluster inicia um novo cluster. É importante ressaltar que cada vértice da árvore principal representa todos os vértices não pertencentes a essa árvore a ele agrupados. Este algoritmo é executado enquanto a quantidade de clusters informada ao algoritmo proposto não for obtida e existir algum vértice da árvore principal ainda não adicionado a um *cluster*.

Ao final do procedimento, pode-se produzir uma solução cujo número de *clusters* é inferior ao determinado previamente. Neste caso, torna-se necessário regenerar a solução mediante a seleção do *cluster* com maior total populacional e aplicar do procedimento construtivo (Figura 2) para particioná-lo. Esta operação ocorre até que a quantidade de *clusters* definida previamente seja atingida.

3.4. Busca Local

O procedimento de busca local tem por objetivo principal, eliminar as penalidades, independente da melhoria da solução. Para isto, ele utiliza a relação das arestas removidas na construção da solução, durante o particionamento da AGM e a relação de clusters que estão penalizados. A Figura 3 apresenta de forma simplificada os passos do algoritmo de busca local:

```
(1) Para cada (aresta e de uma subárvore Tj ) Faça
(2)   Remova a aresta e de Tj obtendo clusters Tj1 e Tj2
(3)   Se (Cluster Tj1) ou (exclusivo) (Cluster Tj2) Penalizado
        Então
(4)       Se (Vertice pode ser migrado)  Então  MigraVertice()
(5)       Fim-Se
(6)   Fim-Se
(7) Fim-Para
```

Figura 3 - Algoritmo de busca local.

Para cada aresta removida, identificam-se a quais clusters seus vértices pertencem, verificando-se, em seguida, a necessidade de migração de vértices entre estes clusters. O que deve ocorrer somente se apenas um desses clusters estiver penalizado. Considerando que a migração de um vértice estará condicionada ao seu nível de conectividade e da localização (*clusters* a que pertencem) dos vértices a ele

conectados. As possibilidades são: Nível 1 - Não pode ser migrado pois o cluster ao qual ele pertence ficaria vazio; Nível - 2: Pode ser migrado; Nível 3 ou superior - Não pode ser migrado se estiver conectado a mais de um vértice que pertença ao mesmo cluster.

Caso seja confirmada a possibilidade de migração, o vértice da aresta removida pertencente a um cluster não penalizado migrará para o cluster penalizado. Esta operação não garante a remoção da penalidade, que o cluster de origem do vértice migrado seja penalizado e nem mesmo a melhoria da qualidade da solução. Contudo, a sua sucessiva reaplicação pode atuar causando um efeito cascata na correção das penalidades dos clusters.

3.5. Mutação

O procedimento de mutação consiste unir os clusters adjacentes, ou seja, aqueles cujos os vértices das arestas removidas pertencem (se um valor aleatório for superior a uma probabilidade definida como parâmetro do algoritmo). Após a união, particiona-se o novo cluster utilizando a primeira versão do procedimento construtivo, o algoritmo semi-guloso.

3.6. Cruzamento

O cruzamento de soluções realiza trocas de partes dos indivíduos previamente selecionadas, objetivando a obtenção de novas soluções de melhor qualidade. A execução deste procedimento também está condicionada a uma probabilidade submetida como parâmetro ao algoritmo. O cruzamento utilizado no algoritmo proposto é o de um ponto em que, implicando na troca de genes entre duas soluções.

Neste trabalho um ponto de corte corresponde a uma das arestas removidas na construção dos *clusters*. Para isto é necessária a identificação de um mesmo arco removido nas soluções candidatas ao cruzamento e, além disso, que o número de grupos existentes entre as partes identificadas pelo ponto de corte sejam equivalentes. Sem esta verificação é possível gerar soluções não viáveis com o número de clusters diferente do número de clusters fixado previamente.

3.7. Elitismo

O algoritmo utiliza a técnica de elitismo e armazena as melhores soluções obtidas com e sem penalidade. E a cada iteração, uma dessas soluções é inserida na população objetivando, através dos procedimentos de busca local e mutação, a minimização da função objetivo (Eq.(6)). Quando não há uma solução elite sem penalidade, a melhor solução penalizada é inserida na população. Ao final da execução, o algoritmo terá a melhor solução viável que satisfaz às restrições de capacidade e contiguidade.

4. Resultados Computacionais

4.1. Instâncias Utilizadas

Com a finalidade de efetuar uma primeira avaliação dos algoritmos propostos neste trabalho, utilizou-se um conjunto de instâncias, geradas a partir dos dados do Censo Demográfico de 2000 (dados de uso público), considerando as informações das áreas de ponderação das seguintes unidades da federação (UFs): Amazonas, Rondônia, Acre, Espírito Santo e Bahia. Na tabela 1 temos o total de áreas de ponderação em cada uma das UFs e o total de vizinhanças entre estas áreas.

Tabela 1 - Instâncias (UFs)

	ES	RO	AC	AM	BA
Vértices (APONDs)	73	14	21	61	409
Arestas	350	46	58	286	2020

As informações disponibilizadas para cada uma destas unidades estão separadas em dois arquivos, quais sejam: (1) um arquivo que contém a informação de vizinhança entre as áreas de ponderação associadas a cada UF e (2) um arquivo que contém a identificação da área de ponderação e as variáveis utilizadas no cálculo das distâncias: Total de casas, de domicílios, de Pessoas, Soma dos Rendimentos Totais dos Domicílios da APOND, dos Anos de Estudo do Responsável pelos Domicílios, dos

Rendimentos perCapita dos Domicílios e Número médio de Anos de Estudo do Responsável pelos Domicílios. Sendo utilizada como variável de capacidade o total de pessoas.

4.2. Resultados Obtidos

Esta seção apresenta os resultados computacionais obtidos a partir da aplicação do algoritmo proposto. Tal algoritmo foi executado em um computador com 1GB de RAM e dotado de um processador AMD Turion x2 1.6 GHz.

O algoritmo foi executado 10 vezes para cada uma das configurações apresentadas pela Tabela 2 em todas as UFs citadas na seção anterior. A busca local foi utilizada em todas as configurações para eliminação de penalidades. Os percentuais de cruzamento e de mutação fixados no procedimento construtivo foram respectivamente de 80% e 50% e o total de gerações foi de 5. Foi utilizada como capacidade mínima do *cluster* 75% da média entre o total populacional (todas as APONDs) e a quantidade *clusters*. Observando-se que os percentuais de mutação e cruzamento, bem como o total de gerações, foram determinados a partir de um conjunto de experimentos realizados previamente.

A Tabela 3 indica as configurações que obtiveram os melhores resultados conhecidos ou distantes apenas 5% desses valores, considerando a definição de 3 e 4 *clusters.* Os melhores resultados conhecidos foram obtidos na execução sistemática do algoritmo utilizando todas as combinações de técnicas (configurações) do algoritmo proposto.

É possível observar que a primeira versão do algoritmo construtor produz melhores resultados na maioria dos experimentos. A configuração 8, entretanto, mesmo utilizando a segunda versão de algoritmo construtor obteve melhor resultado em três configurações e de maneira isolada, ou seja, as demais configurações obtiveram valor da função aptidão no mínimo 5% superior.

As Tabelas 4 e 5 apresentam os melhores valores da função aptidão e tempo de execução (em segundos) em cada configuração para 3 e 4 *clusters* respectivamente.

Tabela 2 – Configurações (X = Indica que o operador foi utilizado)

	Configurações							
	1	2	3	4	5	6	7	8
Construtor (versões 1, 2)	v1	v1	v1	v1	v2	v2	v2	v2
Cruzamento (utilizado)		X		X		X		X
Mutação			X	X			X	X

Tabela 3 - Configurações dos melhores resultados obtidos

		Instâncias				
		RO	AC	AM	BA	ES
Clusters	3	1,2,3,4	Exceto 2	8	Todas	1,2,3,4
	4	1,3,4	8	2,4,7,8	Exceto 5,6	8

Em trabalhos futuros, pretende-se implementar novos procedimentos de construção que gerem componentes conexas (*clusters*) e novos procedimentos de busca local que trabalhem diretamente sobre estas componentes, ao invés de particionamento da AGM.

Tabela 4 - Resultados obtidos considerando 3 *clusters* em que F é o valor da função aptidão e T o tempo de execução em segundos.

		RO		*AC*		*AM*		*BA*		*ES*	
		F	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*
Configuração	**1**	104,9	1	23,9	1	98,0	1	1569,6	16	410,7	1
	2	104,9	1	26,7	1	96,0	1	1528,1	17	409,9	1
	3	104,9	1	23,9	1	95,2	1	1566,2	40	409,9	1
	4	103,7	1	23,9	1	89,7	1	1498,0	35	401,9	1
	5	171,0	1	23,8	1	93,9	1	1577,8	1	675,1	1
	6	171,0	1	23,8	1	93,9	1	1577,8	2	675,1	1
	7	155,1	1	23,8	1	89,5	1	1558,7	27	650,1	1
	8	149,0	1	23,3	1	76,2	1	1532,4	18	472,9	1

Tabela 5 - Resultados obtidos considerando 4 *clusters* em que F é o valor da função aptidão e T o tempo de execução em segundos.

		RO		*AC*		*AM*		*BA*		*ES*	
		F	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*
Configuração	*1*	93,8	1	10,8	1	93,4	1	1554,6	24	403,2	2
	2	103,6	1	10,3	1	84,0	2	1540,8	24	410,6	1
	3	93,8	2	22,7	1	94,2	1	1485,0	60	405,6	2
	4	93,8	1	10,5	1	82,4	2	1528,2	50	548,4	2
	5	158,9	1	22,3	1	88,8	1	1571,7	1	547,5	1
	6	152,6	1	22,3	1	88,8	2	1571,7	2	547,5	1
	7	135,9	1	22,2	1	84,8	1	1523,0	42	493,8	1
	8	134,8	1	8,8	1	86,1	1	1534,4	27	364,7	2

Referências:

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

[2] Assunção, R. M., Lage J. P. e Reis A. E. (2002) Análise de Conglomerados Espaciais Via Árvore Geradora Mínima. **Revista Brasileira de Estatística**, v. 63, n. 220, 7-24.

[3] Berkhin, P. (2002). Survey of Clustering Data Mining Techniques. **Accrue Software**.

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

[5] Dias, C. R.; Ochi, L. S.(2003) Efficient Evolutionary Algorithms for the Clustering Problem in Directed Graphs, **Proceedings of the 2003 IEEE Congress on Evolutionary Computation**. v.1, pp.983–988.

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

[7] Hartuv, E.; Shamir, R. (1999). A Clustering Algorithm based on Graph Connectivity. **Technical Report**, Tel Aviv University, Dept. of Computer Science.

[8] Openshaw, S., (1977), A geographical solution to scale and aggregation problems in regionbuilding, partitioning and spatial modelling. **Transactions of the Institute of British Geographers** (New Series), 2, pp. 459–472.

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

[10] Silva, A. N., Cortez, B. F. e Matzenbacher L. A. (2004) Processamento das Áreas de Expansão e Disseminação da **Amostra no Censo Demográfico 2000**, Textos para Discussão, número 17, IBGE/DPE/COMEQ.

[11] Trindade, A.R.; Ochi, L.S. (2006) Um algoritmo evolutivo híbrido para a formação de células de manufatura em sistemas de produção. **Pesquisa Operacional, vol. 26(2)**, pp. 255-294.