Desenvolvimento e Análise Experimental de Algoritmos Evolutivos para o Problema de Clusterização Automática em Grafos Orientados

C. Rodrigo Dias e Luiz S. Ochi
Instituto de Computação, Universidade Federal Fluminense, Niterói, RJ, Brasil
{cdias, satoru}@ic.uff.br

Resumo

Os algoritmos evolutivos (AEs) e, em particular, o seu representante mais popular, o algoritmo genético (AG), vêm sendo utilizados cada vez mais na solução de problemas NP-Completo e NP-Difícil. Particularmente no caso dos AGs, tem ocorrido uma grande evolução a partir das primeiras versões encontradas na literatura. Nessa evolução, os AGs vêm recebendo diversas contribuições para melhorar seus desempenhos, tanto na qualidade das soluções geradas como no tempo exigido na sua convergência para uma solução alvo. O objetivo deste trabalho é realizar uma análise experimental de alguns aspectos que interferem diretamente no desempenho destes algoritmos quando aplicados à solução de problemas de clusterização automática de vértices em um grafo. Para isso foram desenvolvidos diferentes procedimentos que, quando acoplados a um AG, podem melhorar sensivelmente o seu desempenho.

1. Introdução

As técnicas de clusterização vêm sendo tratadas com freqüência na literatura para a solução de vários problemas de aplicação prática em diferentes áreas do conhecimento. Em trabalhos recentes as técnicas de clusterização têm sido utilizadas para a mineração de dados, onde o objetivo é identificar grupos de itens relacionados em uma base de dados [1] [7], em aplicações de Bioinformática, principalmente no que tange à descoberta de padrões de expressão gênica em microarrays [2] [8] [11], em Engenharia de Software, para particionar a estrutura modular de um sistema [5], dentre outras aplicações.

Dado um conjunto com n objetos $X = \{X_1, X_2, ..., X_n\}$, o problema de clusterização consiste na obtenção de um conjunto de k *clusters*, ($k \leq n$), $C = \{C_1, C_2, ..., C_k\}$, tal que os objetos contidos em um *cluster* C_i possuam uma maior similaridade entre si do que com os objetos de qualquer um dos demais *clusters* do conjunto C. O conjunto C é considerado uma *clusterização* com k *clusters* caso as seguintes condições ocorram:

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

$$C_i \neq \varnothing,\ 1 \leq i \leq k \quad (2)$$

$$C_i \cap C_j = \varnothing,\ 1 \leq i,j \leq k \text{ e } i \neq j \quad (3)$$

O valor de k pode ser conhecido *a priori* ou não. Caso o valor de k seja fornecido como parâmetro de entrada para a solução do problema de clusterização, este é referenciado na literatura como "*problema de k-clusterização"* [7]. Caso contrário, isto é, caso o k seja desconhecido, o problema é referenciado como "*problema de clusterização automática*". Neste último caso, a obtenção do valor de k fará parte do processo de solução do problema.

O problema de clusterização em ambos os casos, é classificado como um problema NP-Difícil, limitando com isso o uso exclusivo de métodos exatos [5]. Dessa forma, a maioria absoluta dos métodos existentes na literatura são métodos aproximados ou heurísticas. Dentre os diferentes métodos aproximados que podem ser utilizadas para a solução de problemas de clusterização, podemos citar os algoritmos evolutivos (AEs), com destaque para o seu representante mais popular, o algoritmo genético (AG).

Devido à grande heterogeneidade das aplicações de problemas de clusterização, as heurísticas utilizadas são normalmente desenvolvidas para determinadas classes de problemas, ou seja, não existe uma heurística genérica que seja a melhor em todas as aplicações de clusterização.

As heurísticas existentes podem ser classificadas, de modo geral, em métodos hierárquicos e métodos de particionamento [7]. Nos algoritmos de clusterização hierárquica tradicionais os *clusters* vão sendo formados gradativamente através de aglomerações ou divisões, formando uma hierarquia de *clusters* representada através de uma estrutura em árvore. Nos algoritmos de

clusterização que utilizam algum método de particionamento, o conjunto de objetos é dividido em k subconjuntos, podendo k ser conhecido ou não, a configuração (solução) obtida é avaliada através de uma função-objetivo e novas configurações vão sendo obtidas através da migração de objetos entre os *clusters*, até que algum critério de parada seja alcançado.

Para medir o quanto um objeto é similar a outro e, dessa forma, identificar se ambos devem estar contidos em um mesmo *cluster* ou não, deve ser utilizada uma "*medida de similaridade*", que é específica para cada problema de clusterização a ser tratado. Um importante critério utilizado para identificar a similaridade entre dois objetos é a *distância* entre eles. Para o problema tratado neste trabalho será utilizada uma medida de similaridade que considera as conexões entre os vértices de um grafo orientado, e que será tratada adiante.

2. Clusterização em Grafos Orientados Utilizando Algoritmos Evolutivos.

O problema de clusterização aplicado a grafos, também referenciado na literatura como *problema de particionamento de grafos,* consiste em, dado um grafo $G = (V, E)$, com V sendo o conjunto de vértices e E o conjunto de arestas, dividir os vértices de V em subconjuntos disjuntos, ou *clusters*, otimizando alguma função-objetivo. A clusterização em grafos é também um problema NP-Difícil e inicialmente era tratado através de métodos estatísticos ou por métodos determinísticos usando heurísticas de construção e/ou busca local. A partir dos anos 80, o problema passou a ser resolvido predominantemente via heurísticas genéricas conhecidas como metaheurísticas ou heurísticas inteligentes. Dentre as metaheurísticas utilizadas, os algoritmos genéticos (AGs) merecem destaque [3] [4] [6] [9] [10] [12] [13].

Em 1999 Doval et. al. [5] propuseram um AG tradicional para obter, de forma automática, um bom particionamento de um grafo de dependências de módulos (MDG - *Module Dependency Graph*). Um MDG é uma estrutura utilizada pelos projetistas de *software* para tornarem sistemas complexos mais compreensíveis, e corresponde a um grafo orientado em que os módulos de um sistema são representados pelos vértices e as dependências estáticas entre os módulos são representadas pelos arcos do grafo. Embora o problema de particionamento de grafos já seja um tema bastante explorado, isso não se observa em relação ao modelo proposto em [5]. Para este modelo, o trabalho visto em [5] nos parece o mais relevante encontrado na literatura utilizando metaheurísticas, justificando desta forma, a sua descrição a seguir.

No AG proposto por Doval et. al., referenciado neste trabalho como AGT (Algoritmo Genético Tradicional), dado um grafo orientado, é realizada uma clusterização automática. Cada indivíduo do AGT corresponde a uma solução viável, podendo cada solução possuir um número de *clusters* diferente das demais. Para avaliar a qualidade de uma clusterização, os autores apresentam uma função objetivo que leva em consideração as conexões entre os vértices do grafo.

A função objetivo, denominada *MQ* (*Modularity Quality* – qualidade de modularização) é usada como função de aptidão do AGT. O objetivo é encontrar um bom (possivelmente ótimo) particionamento através da maximização de *MQ* , que é definido como sendo

$$MQ = \begin{cases} \dfrac{\sum_{i=1}^{k} A_i}{k} - \dfrac{\sum_{i,j=1}^{k} B_{i,j}}{\frac{k(k-1)}{2}} & \forall k > 1 \\ A_i & k = 1 \end{cases} \quad (4)$$

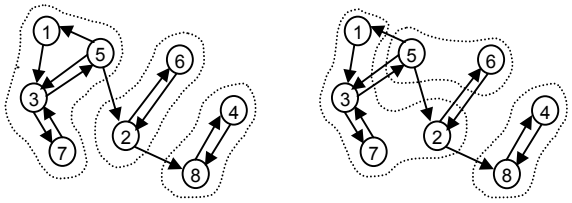
onde A_i é a intra-conectividade do *cluster* i, $B_{i,j}$ é a inter-conectividade entre os *clusters* i e j, e k é o número total de *clusters* da solução. A intra-conectividade A_i de um *cluster* i é uma medida que considera o número total de arcos dentro de cada um dos *clusters* da solução (densidade do *cluster*), sendo definida como

$$A_i = \frac{\mu_i}{N_i^2} \quad (5)$$

onde μ_i é o número total de arcos internos ao *cluster* i e N_i é o seu total de vértices. A inter-conectividade $B_{i,j}$ entre os *clusters* i e j é uma medida que considera o número total de arcos entre os pares de *clusters* de uma solução. Considerando um par de *clusters* i and j, ε_{ij} sendo o total de arcos do *cluster* i ao *cluster* j, N_i e N_j sendo o total de vértices dos *clusters* i e j, respectivamente, a medida de inter-conectividade $B_{i,j}$ do par de *clusters* é dada por

$$B_{i,j} = \begin{cases} 0 & if \quad i = j \\ \dfrac{\varepsilon_{ij}}{2N_iN_j} & if \quad i \neq j \end{cases} \quad (6)$$

Os valores das funções de intra-conectividade e inter-conectividade (5) e (6) estão no intervalo [0,1] e a melhor clusterização será aquela que maximizar o somatório da intra-conectividade de todos os *clusters* e minimizar o somatório da inter-conectividade de todos os pares de *clusters* possíveis. Com base no exposto, a função *MQ* (4) encontra-se no intervalo [-1,1] e deve ser maximizada. Para exemplificar a utilização da função *MQ*, a Figura 1 apresenta um grafo com 8 vértices e duas clusterizações possíveis (os *clusters* estão delimitados pelas linhas tracejadas). Os valores das respectivas funções *MQ* são indicados na parte inferior de cada clusterização. Observe que quanto maior for o valor de *MQ*, melhor será a qualidade da clusterização, ou seja, na Figura 1, a solução (a) é superior à solução (b).



(a): $MQ = 0{,}3958$ (b): $MQ = 0{,}0833$

Figura 1. Diferentes clusterizações de um grafo

No AGT, o esquema de codificação utilizado para os indivíduos é o esquema denominado *group-number* [4], no qual a clusterização de um grafo de n vértices é representado por um vetor de n inteiros onde o i-ésimo inteiro indica o número do *cluster* onde o i-ésimo vértice encontra-se naquela clusterização. Neste esquema, todos os indivíduos correspondem a soluções válidas. A Figura 2 apresenta um exemplo de indivíduo durante o processo de clusterização de um grafo com 8 vértices através do AGT.

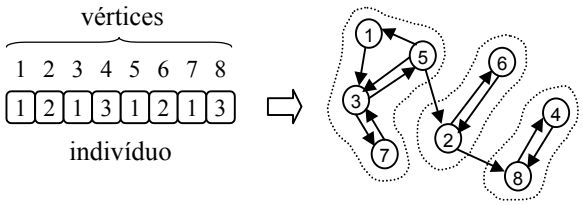


Figura 2. Exemplo de codificação de um indivíduo

O número de *clusters* indicado por cada indivíduo pode variar entre 1 e n. O dois casos extremos ocorrem quando todos os elementos do indivíduo possuírem um mesmo valor e quando cada elemento do indivíduo possuir um valor diferente dos demais, respectivamente.

Sobre as populações do AGT, Doval et. al. [5] propõem que a população inicial seja gerada aleatoriamente e o número de indivíduos mantido em cada geração seja fixo em um valor igual a $10 \times n$, onde n corresponde ao total de vértices do grafo. A partir da população inicial, o processo evolutivo consiste na avaliação da população (através da computação da função de aptidão de cada indivíduo), seleção dos indivíduos para a próxima geração, seguida da aplicação dos operadores genéticos de cruzamento e mutação sobre os indivíduos da população. O processo evolutivo é executado novamente e se repete até que algum critério de parada seja alcançado. No AGT proposto, o critério de parada é o total de iterações, que corresponde a $200 \times n$, sendo n o total de vértices do grafo.

Na etapa de seleção de indivíduos para a próxima geração é utilizada a política conhecida como *roleta com elitismo*, em que os indivíduos são selecionados com probabilidade diretamente proporcional ao valor computado para a sua aptidão. O elitismo garante que o melhor indivíduo de uma geração irá fazer parte da geração seguinte.

O operador de cruzamento utilizado para combinar pares de indivíduos para obter novos indivíduos corresponde ao *cruzamento de um ponto*, em que, cada par de indivíduos da população, de acordo com uma taxa de aplicação, é dividido em duas partes e estas partes são permutadas entre os indivíduos do par, conforme exemplificado na Figura 3 (a). Não é necessário realizar ajustes após a aplicação do operador, tendo em vista que para o problema de clusterização automática ele irá não gerar indivíduos inválidos. A taxa de aplicação do operador de cruzamento utilizada no AGT equivale a 0,8.

Após a aplicação do operador de cruzamento é aplicado operador mutação, em que o valor de cada elemento de cada indivíduo (que indica o número do *cluster* do elemento) possui um chance de ser trocado por um valor do intervalo [1, n], ou seja, o número máximo de *clusters* da solução será igual a n, sendo n o total de vértices do grafo. A taxa de aplicação do operador mutação no AGT é definida como sendo $0{,}004 \times \log_2(n)$. Um exemplo de aplicação do operador mutação é apresentado na Figura 3 (b).

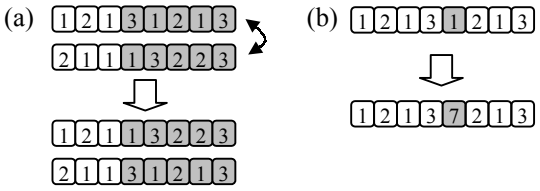


Figura 3. Exemplos de aplicação dos operadores de cruzamento (a) e mutação (b)

3 . Algoritmos Híbridos Propostos

Neste trabalho enfocamos o modelo de particionamento de grafos proposto por Doval et. al. em [5]. Os resultados experimentais obtidos através da execução do AGT mostraram-se, entretanto, insatisfatórios mesmo para grafos a partir de algumas dezenas de vértices. Entretanto, um ponto positivo do AGT, é a sua função *MQ* que permitiu avaliações de clusterizações de forma correta e bastante simples. Com base na análise do comportamento do AGT foi possível identificar seus pontos mais sensíveis (gargalos) e, portanto, sugerir alternativas para tentar melhor a qualidade das soluções obtidas.

As propostas apresentadas neste trabalho correspondem à otimização de parâmetros do AGT, alteração na etapa de seleção e inclusão de um módulo de busca local. A seguir são descritas cada uma das

adaptações propostas, com o respectiva sigla utilizada para indicar a sua utilização no AGT.

3.1. Taxa do Operador Mutação – M(*t*)

Corresponde à utilização de valores distintos de *t* para a taxa de mutação, diferentes da proposta original do AGT. Esta adaptação tem por objetivo avaliar a influência da taxa de aplicação do operador mutação na obtenção de boas soluções pelo AG.

3.2. Seleção por Torneio – T

Corresponde a uma alteração no processo de seleção dos indivíduos para a próxima geração. A seleção de cada indivíduo por torneio, utilizada neste trabalho, corresponde a escolher aleatoriamente quatro indivíduos da população corrente e selecionar o melhor dos quatro indivíduos para participar da próxima geração.

3.3. Inserção de Busca Local – B

O procedimento de busca local (B) é introduzido no intuito de sanar uma das deficiências tradicionais dos AGs incluindo o AGT, que é a dificuldade deste em refinar de forma eficiente as melhores soluções obtidas a cada geração. A busca local irá analisar a vizinhança do melhor indivíduo obtido em cada geração, com o objetivo de tentar melhora-lo. Ou seja, deseja-se obter a cada geração do AG, um ótimo local através da busca local.

Neste procedimento, cada vértice migra do seu *cluster* atual para o *cluster* com o qual ele compartilha o maior número de arcos. Caso a sua migração acarrete uma melhora na avaliação do indivíduo, ele é mantido no novo *cluster* ou, caso contrário, o vértice retornará para o seu *cluster* original.

4. Análise Experimental

Diferentemente do trabalho de Doval et. al., em que foram utilizadas apenas instâncias pequenas (menores ou iguais a 20 vértices), neste trabalho, cada população das versões do AGT utilizadas nos experimentos consistirá de um total de *máx*(100, *n*) indivíduos e o total de iterações será 20 × *n*. Estes valores foram redefinidos tendo em vista que permitiram resultados comparáveis ao AGT original, com uma redução significativa no tempo total de execução para grafos com dimensões elevadas.

Com relação às demais características do AGT, os experimentos foram realizados utilizando seus valores originais e acrescentando as adaptações sugeridas na Seção 3. Os grafos utilizados nos experimentos foram gerados de forma artificial (devido a inexistência de instâncias na literatura para este modelo de problemas de particionamento) com base nas definições do total de vértices, total de *clusters* da melhor solução, e valores desejados para o somatório da intra-conectividade e inter-conectividade (o que determinará o total de arcos internos e entre os *clusters da solução*). Desta forma, foi obtido, durante a geração dos grafos, um valor para *MQ* que, caso não seja o ótimo, encontra-se próximo a ele. A clusterização com este valor para *MQ* será referenciada aqui como *melhor solução conhecida*. Uma observação a ser feita, é que mesmo não havendo disponíveis instâncias do problema, torna-se posssível avaliar a qualidade das soluções geradas pelo fato do *MQ* máximo ser sempre igual ou menor que 1, ou em outras palavras, o valor *MQ* = 1 neste caso é um *upper bound* (*limite superior*) do valor ótimo viável de *MQ*. Ou seja, se obtivermos uma solução *MQ* = 0.85 podemos garantir que no pior caso, ela estará a uma distância no máximo a 0.15 do valor ótimo global.

A Tabela 1 apresenta os grafos utilizados nos experimentos com suas respectivas características e a melhor solução conhecida. Pela Tabela 1, observa-se que foram testadas instâncias de 10 até 500 vértices, com o número de arcos variando de 26 até 11.747.

Tabela 1. Grafos utilizados nos experimentos

Nome do Grafo	Total de Vértices	Total de Arcos	Melhor Solução Conhecida	
			Valor de *MQ*	Número de *Clusters*
A10	10	26	0,7083	3
A20	20	98	0,7371	4
A40	40	171	0,7087	5
A60	60	969	0,7402	3
A80	80	539	0,7780	10
A100	100	524	0,8092	7
A160	160	1.335	0,7368	16
A500	500	11.747	0,8759	20

A Figura 4 compara as soluções obtidas pelo AGT em relação à melhor solução encontrada. Para estes experimentos o AGT foi executado 10 vezes para cada grafo.

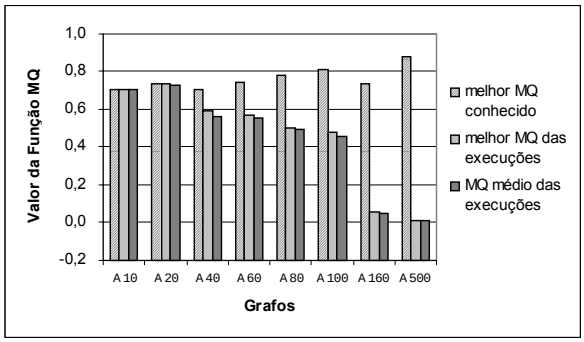


Figura 4. Experimentos utilizando-se o AGT

A Figura 5 apresenta um gráfico de distribuições percentuais cumulativas para diferentes versões do AGT. Neste tipo de gráfico, também utilizado na Figura 6, cada polígono refere-se a uma bateria de execuções do AGT para uma mesma instância, com as características indicadas na legenda. Cada polígono indica, dado um valor alvo arbitrado para *MQ*, qual o percentual das execuções (eixo y) alcançou o valor alvo ao longo de um intervalo de tempo de execução (eixo x).

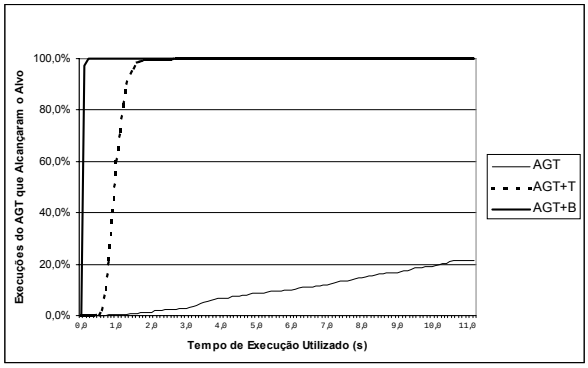


Figura 5. Análise das diferentes versões do AGT

Nos experimentos descritos na Figura 5, foram realizadas, para uma instância específica, 500 execuções para o AGT e para cada uma das suas variações com a seleção por torneio e a inclusão de procedimento de busca local. Foi utilizado o grafo A60 e o valor da *MQ* alvo arbitrado foi 0,5. O AGT forneceu o pior resultado (curva mais a direita), obtendo uma convergência muito inferior (mais lenta) às demais versões em todas as execuções. O melhor resultado foi obtido acrescentando-se o módulo de busca local (B) descrito na Seção 3.3 (curva mais a esquerda). Vale observar também que a simples troca do método de seleção, de roleta para torneio (T), permitiu uma melhora significativa nos resultados obtidos.

Com base no desempenho ruim do AGT, foram realizados novos experimentos com diferentes valores para a taxa de mutação. Cada experimento foi realizado utilizando-se o grafo A60 e executando o AGT 500 vezes. Para um grafo de 60 vértices a taxa de mutação definida para o AGT original é 0,0240. Entretanto, pode-se observar que os melhores resultados foram obtidos para valores próximos a 0,004.

Análises sobre convergência para um alvo fixado, foram efetuadas para todas as instâncias analisadas neste trabalho, e os resultados foram similares ao caso ilustrado (A60). Com estes experimentos, conclui-se que o desempenho do AGT é muito sensível ao valor definido para a taxa de mutação e, portanto, esta deve ser definida com muito critério.

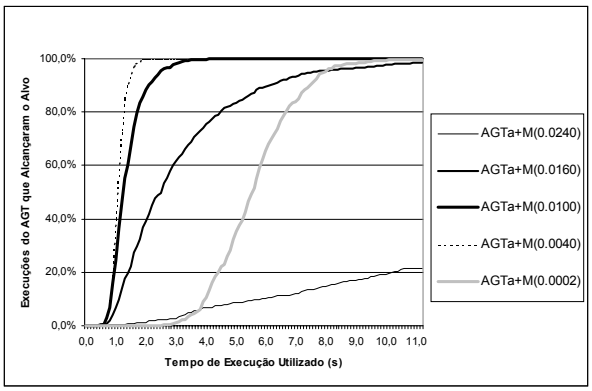


Figura 6. Variações na taxa de mutação

Outros experimentos foram realizados com o AGT para todos os grafos da Tabela 1, com 10 execuções por experimento, utilizando a melhor taxa de mutação obtida para o grafo A60, que corresponde a 0,004. As melhores soluções obtidas pelo AGT original e pelo AGT com variação na taxa de mutação, são apresentadas, de forma comparativa na Figura 7.

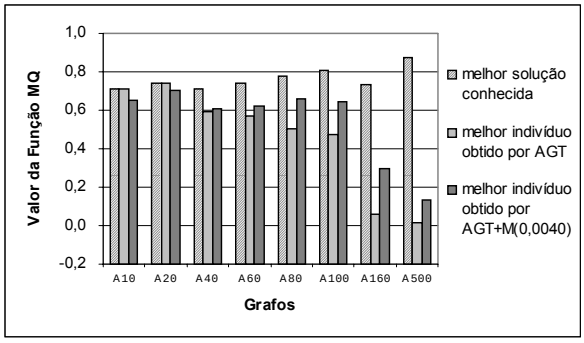


Figura 7. Comparação entre versões do AGT

Na Figura 7 pode-se observar que, apesar do AGT que utiliza o valor 0,004 para a taxa de mutação obter melhores resultados para os grafos com número de vértices igual ou superior a 40, o AGT com a taxa de mutação original obteve os melhores resultados para os grafos menores. Para os grafos maiores, as diferenças nos desempenhos das versões são mais significativas.

Os resultados apresentados na Figura 8 mostram que, alterando-se a seleção originalmente realizada pelo método da roleta para o método de torneio, os resultados melhoram em relação ao AGT original. Entretanto, somente após o ajuste na taxa de mutação a melhora das soluções obtidas é realmente significativa. Para os grafos apresentados, a combinação destas variações no AGT apresentou melhores resultados do que as versões que utilizam cada variação isoladamente.

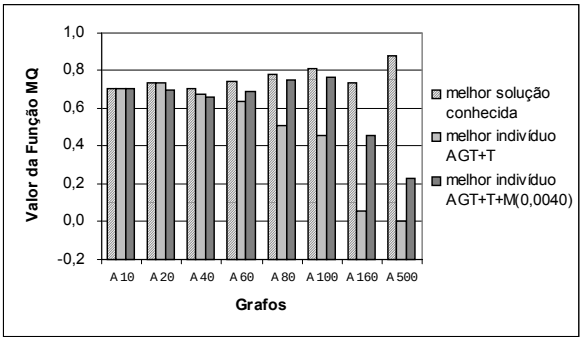


Figura 8. Versões do AGT com torneio

Finalmente, o gráfico da Figura 9 apresenta os resultados obtidos inserindo-se o procedimento de busca local, em comparação com a melhor solução conhecida e a melhor versão com torneio. A inserção do procedimento de busca local permitiu que todos os experimentos, exceto para o grafo com 500 vértices, conseguissem obter a melhor solução (para o grafo com 500 vértices, entretanto, o resultado com a busca local foi muito superior à melhor versão com torneio). A média dos melhores indivíduos obtidos nas 10 execuções do AGT+B também foi superior à melhor versão com torneio para a maioria dos grafos utilizados.

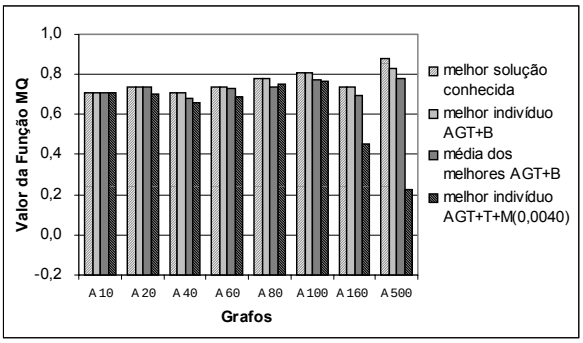


Figura 9. Versões com busca local e com torneio

5 . Conclusão

Neste trabalho foi tratado o problema de clusterização de grafos orientados através de AGs e foram apresentadas propostas de adaptações em um AG da literatura, com o intuito de melhorar o seu desempenho.

A principal contribuição deste trabalho é a análise experimental comparativa das diferentes versões do AGT original, que permitiu verificar o quanto um AG pode ser sensível à mudança do parâmetro "taxa de mutação" e ao tipo de seleção utilizada para a reprodução dos indivíduos ao longo das gerações. A inclusão do procedimento de busca local moustrou-se fundamental para que o AG conseguisse um bom desempenho para grafos com dimensões elevadas.

6 . Referências

[1] Berkhin, P., "Survey of Clustering Data Mining Techniques". Accrue Software, 2002.

[2] Ben-Dor, A., Shamir, R. e Yakhini, Z., "Clustering gene expression patterns". J. Comput. Biol., vol. 6, pp. 281-297, 1998.

[3] Chiun, Y. e Lan, L. W., "Genetic Clustering Algorithms". European Journal of Operational Research (135) 2, pp. 413-427, 2001.

[4] Cole, R. M., "Clustering with Genetic Algorithms". Tese de Mestrado, Dept. of Computer Science, Univ. of Western Australia, 1998.

[5] Doval, D., Mancoridis, S. e 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.

[6] Drummond, L. M. A., Vianna, D. S. e Ochi, L. S., "Genetic Algorithm for the Vehicle Routing Problem". Future Generations on Computer Systems, Elsevier, vol. 14(5-6), pp. 285-292, 1998.

[7] D. Fasulo, "An Analysis of Recent Work on Clustering Algorithms". Relatório técnico, Dept. of Computer Science and Engineering, Univ. of Washington, 1999.

[8] Hartuv, E., Schmitt, A., Lang, J. et al., "An Algorithm for Clustering for Gene Expression Analysis". Proc. of Third Annual Int. Conf. on Computational Molecular Biology , 1999.

[9] Lorena, L. N. e Furtado, J. C., "Constructive Genetic Algorithm for Clustering Problems". Evolutionary Computation, vol. 9, no. 3, pp. 309-327, 2001.

[10] Maini, H. S., Mehrotra, K. G., Mohan, C. K. e Ranka, S., "Genetic Algorithms for Graph Partitioning and Incremental Graph Partitioning". Proc. of the Conference on Supercomputing, pp. 449-457, 1994.

[11] Oja, M., Nikkila, J., Toronen, P., Wong, G., Castren, E. e Kaski, S., "Exploratory clustering of gene expression profiles of mutated yeast strains". In Comp. and Statistical Approaches to Genomics. Kluwer, pp. 64-78, 2002.

[12] Ochi, L. S. e Rocha, M. L., "A new hybrid evolutionary algorithm for the vehicle routing and scheduling problems". Proc. of the Ninth International Conference on Intelligence Systems: Artificial Intelligence Applications for the New Millennium, pp 135-140, 2000.

[13] Ochi, L. S., Drummond, L. M. A., Vianna, D. S., " An asynchronous parallel metaheuristic for the period vehicle routing problem", Future Generations on Comp. Systems, Elsevier, pp. 379-386, vol 17(4), 2001.