Um algoritmo genético com chaves aleatórias viciadas aplicado ao problema da clique máxima

Antônio M. Pinto[1], **Dayvson W. F. Almeida** [1], **Christyellen S. C. Lima** [1], **Igor R. B. Estrela** [1], **Glaubos Clímaco** [1]

[1]Departamento de informática – Universidade Federal do Maranhão (UFMA)
Caixa Postal 1966 – Vila Bacanga 65080-805 – São Luís – MA – Brasil

{antoniomorar, dayvsonwilkson, christyellen01}@gmail.com
igor.star.ie@gmail.com, francisco.glaubos@ufma.br

Abstract. *The maximum clique problem consists of finding the largest subgraph within a given graph. This problem is well known in the field of Operational Research and has several applications in our daily life. In this work, we propose solving the maximum clique problem making use a mathematical solver and the well-known BRKGA metaheuristic. To validate our proposals, we used a set of test-problems that are available on-line in the literature. The BRKGA proved to be very efficient in solving the approached problems, and competitive with the mathematical solver used.*

Resumo. *O problema da clique máxima consiste em encontrar o maior subgrafo completo dentro de um dado grafo. Esse problema é bem conhecido no campo da Pesquisa Operacional e tem diversas aplicações em nosso cotidiano. Neste trabalho, propõe-se resolver o problema da clique máxima utilizando um solver matemático e a bem conhecida metaheurística BRKGA. Para validar nossas propostas, utilizamos um conjunto de problemas-teste que estão disponíveis on-line na literatura. O BRKGA provou ser eficiente na resolução dos problemas e competitivo com o resolvedor matemático utilizado.*

1. Introdução

Uma clique de um grafo não direcionado é um subgrafo no qual todos os pares de nós distintos são conectados por uma aresta. Um clique máximo de um grafo é um clique com o número máximo de nós. Computar o clique máximo de um grafo é um problema de otimização combinatória encontrado em muitas aplicações diferentes da vida real, como análise de *cluster*, recuperação de informações, redes móveis e visão computacional [Konc and Janezic 2007].

O problema da clique máxima (PCM) é computacionalmente intratável: é um dos primeiros problemas que se provou ser NP-Difícil [Bomze et al. 1999]. Dada essa prova, muitos pesquisadores têm concentrado seus esforços em desenvolver metaheurísticas eficientes que fornecessem soluções aproximadas em um tempo de CPU razoável. Uma metaheurística que têm sido utilizada por diversos pesquisadores é o algoritmo genético com chaves aleatórias viciadas (do inglês *Biased Random-key Genetic Algorithm* - BRKGA) [Chaves et al. 2016][Martinez et al. 2011][Zudio et al. 2018].

Neste trabalho, é proposto um BRKGA e aplicado um solver matemático para a resolução do PCM. O restante deste artigo é organizado da seguinte maneira. Na Seção

2, o PCM é apresentado com mais detalhes, incluindo uma formulação matemática da literatura. Na Seção 3, é descrito como o método BRKGA foi desenvolvido. Na Seção 4, experimentos computacionais e resultados numéricos são detalhados, e por fim, na Seção 5, é apresentada a conclusão deste trabalho e investigações futuras são propostas.

2. O problema da clique máxima

Um grafo, em linhas gerais é uma tupla $G = (V(G), E(G), w(G))$, no qual $V(G)$ representa o conjunto não vazio de vértices do grafo, e $E(G)$ a ligação entre esses vértices (Arestas) e $w(G)$ uma função de incidência que associa um par de vértices do grafo a uma aresta do mesmo [Goldbarg and Goldbarg 2012]. A clique de um grafo não direcionado, consiste em um sub-grafo completo $G' = (V'(G'), E'(G'))$, onde $V'(G') \subseteq V'(G')$ e $E'(G') \subseteq E'(G')$, no qual para cada par de vértices há uma aresta entre eles.

Apesar de existir registros sobre o estudo de subgrafos completos em [Erdös and Szekeres 1935], o termo "clique" surgiu em [Luce and Perry 1949]. Esses últimos autores utilizaram subgrafos completos em redes sociais para modelas cliques de pessoas, ou seja, grupos de pessoas nos quais todas se conhecem. As Figuras 1 e 2 apresentam exemplos de cliques e clique máximo em um grafo.

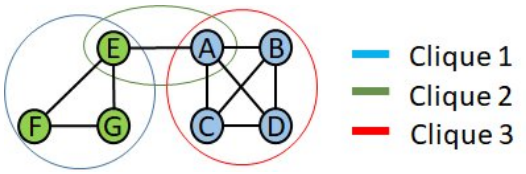


Figura 1. Cliques de um grafo.

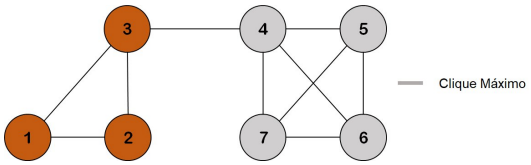


Figura 2. Clique máxima

Para a resolução ótima do PCM, algumas formulações matemáticas foram propostas na literatura. A seguir apresentamos uma formulação conhecida como *edge formulation* [Konc and Janezic 2007].

Considere x_i como sendo uma variável binária, na qual x_i recebe valor 1 se o vértice i está na clique e 0 caso contrário. Assim, o PCM pode ser formulado como um problema de programação linear inteiro da seguinte maneira:

$$max \quad \sum_{i=1}^{n} x_i \tag{1}$$

$$s.a. : x_i + x_j \leq 1 \qquad \forall (i,j) \notin E \tag{2}$$

$$x_i \in \{0,1\} \qquad \forall i \in V \tag{3}$$

Na formulação acima, a função objetivo (1) busca maximizar o número de vértices na clique, e as restrições (2) garantem que se uma aresta (i,j) não existe no grafo, então ou o vértice i ou o vértice j pertence à clique. Por fim, as restrições (3) definem o domínio das variáveis.

3. BRKGA proposto

O algoritmo genético de chave aleatória viciadas (BRKGA) [Martinez et al. 2011] é um aprimoramento do algoritmo genético clássico (AG), e seu principal objetivo é mitigar a dificuldade do operador do AG em lidar com soluções viáveis. AG's [Goldberg and Holland 1988] são métodos que simulam a evolução de uma população ao longo de um certo número de gerações. Esses algoritmos aplicam o conceito de sobrevivência do indivíduo mais apto para encontrar soluções de boa qualidade para problemas de otimização.

Uma população evolui em várias gerações, a fim de explorar o espaço da solução, escapando dos ótimos locais. Cada indivíduo ou cromossomo representa uma solução para o problema de otimização. Normalmente, as soluções são codificadas por uma cadeia finita de bits ou inteiros, chamados genes. Essa codificação permite a representação da reprodução entre dois pais. A função objetivo do problema de otimização é usada como um critério de adequação (*fitness*) na seleção de bons indivíduos.

Em cada geração do AG, uma nova população é gerada a partir da combinação de elementos pertencentes à população atual, realizada por dois operadores principais: cruzamento e mutação. A nova população é obtida da seguinte forma: i) inicialmente, uma pequena porcentagem dos melhores indivíduos da população é copiada diretamente para a próxima população; ii) operadores de crossover, determinísticos ou probabilísticos, são aplicados a pais selecionados aleatoriamente, gerando descendentes para a próxima geração; e iii) com o objetivo de evitar a otimização local, é realizada uma mutação aleatória das posições dos genes.

Algoritmos genéticos de chave aleatória (ou RKGA) foram propostos por [Bean 1994]. Neste método, os cromossomos são representados como cadeias ou vetores de números decimais cujos valores pertencem ao domínio [0,1]. Além disso, um algoritmo determinístico, denominado *decoder*, recebe esse vetor como parâmetro e o associa a uma solução para o problema.

Seguindo as ideias da RKGA, o BRKGA foi desenvolvido por [Gonçalves and Resende 2011]. A principal diferença entre BRKGA e RKGA é a maneira tendenciosa de como os pais são escolhidos no operador de reprodução. De uma população de p indivíduos, cada novo indivíduo é obtido pela combinação de um indivíduo selecionado aleatoriamente de um grupo de elite de pe indivíduos, e outro de um conjunto não elite de $(p - p_e)$ indivíduos, no qual $p_e < p - p_e$. Um único indivíduo pode ser selecionado mais de uma vez e, portanto, pode gerar mais de um filho..

Como o BRKGA requer $p_e < p - p_e$, a probabilidade de um indivíduo de elite ser selecionado para reprodução $(\frac{1}{p_e})$ é maior que a de um indivíduo não elite $(\frac{1}{p-p_e})$ e, portanto, um indivíduo de elite tem uma probabilidade maior de transmitir suas características às futuras gerações. Além disso, o mecanismo implementado no BRKGA para acasalamento entre pais é o *crossover* uniforme parametrizado [Spears and De Jong 1995]

com $\rho > 0.5$. Este último parâmetro corresponde à probabilidade de que um filho herde o gene de seu pai de elite.

Uma vez que o crossover é realizado, a adequação dos novos indivíduos é calculada e uma nova população surge. Esta população é composta por um grupo de elite contendo os melhores indivíduos desta nova população, e um número de indivíduos mutantes gerados aleatoriamente. O resto da população é composto por indivíduos gerados na fase de cruzamento. Um esquema de funcionamento do BRKGA pode ser visto na Figura 3.

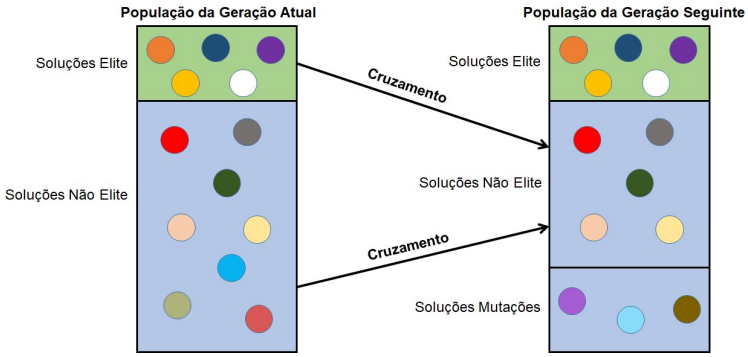


Figura 3. Esquema do BRKGA.

3.1. Decoder

Neste trabalho, foi utilizado um framework BRKGA fornecido por [Toso and Resende 2015]. Este framework gerencia automaticamente módulos que são independentes do problema, incluindo população e seu processo de evolução. Assim, o usuário deve implementar apenas um procedimento dependente de problema (chamado *decoder*), que é responsável por converter um vetor de chaves aleatórias em uma solução para o problema.

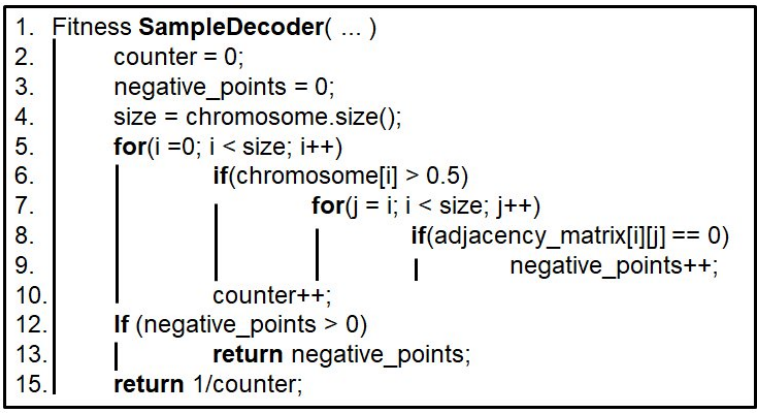
Nesse mesmo *decoder*, a solução recebe um valor de *fitness* ou uma penalidade se for uma solução inviável para o problema. Para o PCM, cada chave aleatória ($chromossome[i]$) corresponde a um vértice i do grafo, e se $chromossome[i] > 0.5$ então considerou-se que o vértice i está na clique. Um pseudo-código do *decoder* implementado é apresentado na Figura 4.

Na implementação realizada, uma solução é viável quando os vértice escolhidos pelas chaves aleatórias formam um subgrafo completo. Quando isso não ocorre, tem-se uma solução inviável com um grau de inviabilidade definido pela variável $negative\_points$.

4. Experimentos computacionais

Nesta seção, são apresentados os resultados da formulação matemática utilizando o solver Gurobi (versão 6.8) e do BRKGA proposto. Para validar os métodos propostos, foram utilizadas 14 instâncias disponíveis em `http://iridia.ulb.ac.be/~fmascia/maximum_clique/DIMACS-benchmark`.

Para a execução do solver, todos os seus parâmetros de pré-processamento foram mantidos, e a capacidade de paralelização do processador não foi utilizada. Para a



```
1.  Fitness SampleDecoder( ... )
2.      counter = 0;
3.      negative_points = 0;
4.      size = chromosome.size();
5.      for(i =0; i < size; i++)
6.          if(chromosome[i] > 0.5)
7.              for(j = i; i < size; j++)
8.                  if(adjacency_matrix[i][j] == 0)
9.                      negative_points++;
10.         counter++;
12.     If (negative_points > 0)
13.         return negative_points;
15.     return 1/counter;
```

Figura 4. Pseudo-código do *decoder*.

execução da metaheurística, o número de gerações foi definido como 1000 e o restante dos parâmetros foram mantidos como em [Toso and Resende 2015].

Os resultados dos experimentos são apresentados nas Tabelas 1 e 2. Nessas tabelas, as colunas indicam, respectivamente, o nome instância testada, o número de vértices da instância, o número de arestas da instância, a solução obtida e o tempo despendido. Para o BRKGA, foram realizadas dez execuções com sementes diferentes; a coluna "Média T.(s)" representa o tempo médio, enquanto que a coluna "Sol." indica a melhor solução obtida dessas execuções.

Tabela 1. Resultados do solver Gurobi.

Instância	$\|V\|$	$\|E\|$	Sol.	T. (s)
queen6-6.txt	36	580	6	0,01
myciel5.txt	47	236	2	0,18
miles750.txt	128	4226	31	0,12
1-FullIns_5.txt	282	3247	3	1,12
frb30-15-4.clq	450	83194	30	157,55
frb30-15-1.clq	450	83198	30	411,13
frb30-15-2.clq	450	83151	30	135,29
frb30-15-3.clq	450	83216	30	265,22
frb30-15-5.clq	450	83231	30	45,00
frb45-21-3.clq	945	387795	41	1800,00
frb45-21-5.clq	945	387461	42	1800,00
frb45-21-4.clq	945	387491	42	1800,00
frb45-21-1.clq	945	386854	41	1800,00
frb45-21-2.clq	945	387416	41	1800,00

Ao analisar os resultados contidos na Tabela 1, percebe-se que o Gurobi é um *solver* robusto, capaz de resolver problemas pequenos e médios, em tempo consideravelmente rápido (menos de 1,13 segundos para $|V| < 283$ .Porém devido à complexidade do problema, a solução ótima não é garantida em grafos com mais de 450 vértices, no tempo limite de 1800 segundos.

Tabela 2. Resultados da metaheurística BRKGA.

Instância	$\|V\|$	$\|E\|$	Sol.	Média T. (s)
queen6-6.txt	36	580	6	3,22
myciel5.txt	47	236	2	3,77
miles750.txt	128	4226	24	10,36
1-FullIns_5.txt	282	3247	3	20,06
frb30-15-4.clq	450	83194	21	44,44
frb30-15-1.clq	450	83198	20	46,40
frb30-15-2.clq	450	83151	20	45,42
frb30-15-3.clq	450	83216	21	46,06
frb30-15-5.clq	450	83231	23	44,24
frb45-21-3.clq	945	387795	31	145,95
frb45-21-5.clq	945	387461	29	146,30
frb45-21-4.clq	945	387491	30	147,67
frb45-21-1.clq	945	386854	31	141,10
frb45-21-2.clq	945	387416	29	139,67

Na Tabela 2 são apresentados os resultados obtidos com a metaheurística BRKGA. Todas as instâncias foram executadas em 1000 gerações, e somente nas instâncias menores foram encontradas a solução ótima. Observa-se também que das 14 instâncias testadas, o Gurobi obteve a maior clique máxima em 11, e empatou nas restantes. Apesar de estourar o tempo limite de 1800 segundos para as instâncias de maior porte, o solver ainda atinge soluções melhores que o BRKGA, que não despende mais de 148 segundos em suas execuções.

5. Conclusões e trabalhos futuros

O problema da clique máxima é NP-Difícil e portanto, para a resolução de problemas de grande porte, faz-se necessária a utilização de metaheurísticas. Neste trabalho, foi proposto um BRKGA para a resolução desse problema, e foi observado que o mesmo possui desempenho satisfatório. A metaheurística implementada foi capaz de encontrar soluções ótimas para instâncias pequenas, e soluções aproximadas para problemas de maior porte, em tempo de CPU razoável.

Para verificar a complexidade do problema, foi utilizado um solver matemático (Gurobi) para tentar resolver as instâncias em sua otimalidade. Como esperado, devido à complexidade do problema, verificou-se que o solver é incapaz de resolver problemas grandes, sendo necessário a definição de um tempo limite de execução, para que o computador não ficasse sem memória RAM e interrompesse os experimentos.

A partir dos resultados obtidos, pretende-se investigar o seguinte. Em relação ao BRKGA, há vários relatos na literatura sobre a geração "manual" da população inicial, portanto, um trabalho futuro será criar parte dos indivíduos da população em vez deixá-la totalmente aleatória. Em relação ao Gurobi, será realizada uma pesquisa em torno de métodos de pré-processamento, que visem a redução do grafo de entrada, a fim de tornar viável a resolução de instâncias maiores.

Referências

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA journal on computing*, 6:154–160.

Bomze, I. M., Budinich, M., Pardalos, P. M., and Pelillo, M. (1999). The maximum clique problem. In *Handbook of combinatorial optimization*, pages 1–74. Springer.

Chaves, A. A., Lorena, L. A. N., Senne, E. L. F., and Resende, M. G. (2016). Hybrid method with cs and brkga applied to the minimization of tool switches problem. *Computers & Operations Research*, 67:174–183.

Erdös, P. and Szekeres, G. (1935). A combinatorial problem in geometry. *Compositio mathematica*, 2:463–470.

Goldbarg, M. and Goldbarg, E. (2012). *Grafos: Conceitos, algoritmos e aplicações*. Elsevier.

Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3:95–99.

Gonçalves, J. F. and Resende, M. G. C. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17:487–525.

Konc, J. and Janezic, D. (2007). An improved branch and bound algorithm for the maximum clique problem. *proteins*, 4(5).

Luce, R. D. and Perry, A. D. (1949). A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116.

Martinez, C., Loiseau, I., Resende, M. G. C., and Rodriguez, S. (2011). Brkga algorithm for the capacitated arc routing problem. *Electronic Notes in Theoretical Computer Science*, 281:69–83.

Spears, W. M. and De Jong, K. D. (1995). On the virtues of parameterized uniform crossover. Technical report, DTIC Document.

Toso, R. F. and Resende, M. G. C. (2015). A c++ application programming interface for biased random-key genetic algorithms. *Optimization Methods and Software*, 30:81–93.

Zudio, A., da Silva Costa, D. H., Masquio, B. P., Coelho, I. M., and Pinto, P. E. D. (2018). Brkga/vnd hybrid algorithm for the classic three-dimensional bin packing problem. *Electronic Notes in Discrete Mathematics*, 66:175–182.