Um algoritmo genético híbrido para o problema quadrático binário irrestrito

Bruno de Athayde Prata (UNIFOR) bprata@unifor.br

Resumo

O problema quadrático binário irrestrito (Unconstrained Quadratic Binary Problem – UQBP) pertence à classe de problemas NP-Difícil, e, dadas as aplicações práticas inerentes ao problema supracitado, diversos trabalhos versam sobre sua resolução. O presente trabalho reporta o desenvolvimento e a aplicação de um Algoritmo Genético Híbrido para o UQBP. A heurística proposta combina mecanismos de recombinação e mutação com uma busca local, aplicados coordenadamente em uma população de soluções, de modo a efetuar uma busca eficaz e eficiente no espaço de soluções possíveis. Foram realizados experimentos computacionais em conjuntos de dados existentes na literatura, para instâncias com 500 variáveis. Os resultados obtidos apontam para a qualidade do algoritmo proposto.

Palavras-chave: Metaheurísticas, Otimização Binária Quadrática, Busca Local.

1. Introdução

O problema quadrático binário irrestrito (em inglês *Unconstrained Quadratic Binary Problem* – UQBP) é um problema de otimização combinatória de grande interesse prático, visto que possui uma série de aplicações reais, nas áreas de gestão de tráfico de mensagens, escalonamento de máquinas e análise orçamentária e financeira (LODI *et al.*, 1999). Além disso, Kochenberger *et al.* (2004) apresentam uma abordagem geral para solução de problemas de otimização combinatória, que consiste na transformação de um problema combinatório com variáveis discretas em um UQBP.

O UQBP é um problema de otimização combinatória de difícil resolução, estando enquadrado na classe de problemas NP - Difícil (GAREY e JOHNSON, 1979). Deste modo, métodos aproximativos, tais como algoritmos heurísticos, são requeridos para resolução do problema supracitado, de modo a se obter soluções de elevada qualidade em tempo computacional admissível.

O presente trabalho tem por objetivo reportar a proposição de um Algoritmo Genético Híbrido para resolução UQBP. A heurística proposta é combinada com procedimentos eficientes de busca local e prima por solucionar, com eficácia e eficiência, o problema supracitado.

O artigo é composto por outras cinco seções, descritas a seguir. Na segunda seção, é apresentado o *Unconstrained Quadratic Binary Problem* (UQBP). Na terceira seção é descrita a heurística proposta. Na quarta seção, são apresentados experimentos computacionais realizados em instâncias disponíveis na literatura. Por fim, na quinta seção, são apresentadas as considerações finais do estudo realizado.

2. Problema quadrático binário irrestrito

O problema quadrático binário irrestrito (*Unconstrained Quadratic Binary Problem* – UQBP) pode ser formulado como segue: seja x um vetor de variáveis de decisão e Q uma matriz simétrica de ordem n, deve-se minimizar uma função objetivo z tal que:

$$\min z = x^t Qx = \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j \, , x_i \in \{0,1\} \, , \vee 1,...,n. \qquad (1)$$

Caso a matriz Q seja positiva-definida, o UQBP pode ser solucionado em tempo polinomial pelo método do elipsóide, pois a função z é convexa. Para os casos em que Q é negativa-definida, o UQBP não pode ser solucionado de forma exata em tempo polinomial. É pertinente ressaltar que as aplicações práticas do UQBP quase sempre requerem elementos negativos na matriz Q, fato que impossibilita a resolução exata do mesmo.

Tendo em vista que as variáveis de decisão são binárias e o conjunto de soluções é irrestrito, existem 2^n soluções possíveis para o UQBP. Para valores de n maiores do que 50, a enumeração explícita de todas as possíveis soluções passa a ser inviável. Beasley (1998) apresenta um levantamento bibliográfico sobre algumas abordagens exatas para o UQBP; entretanto, tais métodos não conseguem ser eficazes para instâncias com centenas ou milhares de variáveis de decisão.

Ao contrário do ocorre com a grande maioria dos problemas de otimização combinatória, no UQBP existem multiplicações entre as variáveis de decisão. Tal característica pode implicar em uma não convexidade do espaço de busca, e, sob o ponto de vista computacional, o cálculo do valor da função objetivo incorre em um razoável custo, visto que são necessárias $2 \times n^2$ multiplicações para sua determinação.

Diante do exposto, diversos autores vêm empregando métodos heurísticos para resolução do UQBP, dentre os quais podem ser destacados:

- *Simulated Annealing*: Beasley (1998), Alkhamis *et al.*, (1998); Katayama e Narihisa (1999).
- Busca Tabu: Beasley (1998), Kochenberger *et al.* (2004), Palubeckis (2004) e Palubeckis (2006).
- Algoritmos Genéticos: Lodi *et al.*, (1999), Merz e Freisleben (1999), Katayama *et al.* (2000), Schütz e Pires (2003).
- GRASP: Palubekckis e Tomkevicius (2002).
- Algoritmos Meméticos: Merz e Katayama (2004).
- Algoritmos gulosos com busca local: Merz e Freisleben (2004).

Beasley (1998) afirma que, no caso do UQBP, um algoritmo que trabalha com uma população de soluções, tal como um Algoritmo Genético (AG), não pode competir computacionalmente com heurísticas que trabalham com uma única solução, como, por exemplo, *Simulatd Annealing* (SA) e Busca Tabu (BT).

Tal consideração se baseia no fato de que heurísticas como SA e BT podem realizar movimentos de busca local eficientes com complexidade $O(n)$, enquanto um AG operaria potencialmente em $O(kn)$ ou $O(n^2)$. Todavia, alguns autores têm demonstrado a competitividade dos algoritmos evolucionários para resolução do UQBP (LODI *et al.*, 1999; MERZ e FREISLEBEN, 1999; MERZ e KATAYAMA, 2004).

3. Algoritmo genético proposto

Em 1798, o economista britânico Thomas Malthus afirmou que as populações humanas mantinham o equilíbrio, apesar do seu crescimento exponencial, devido a limitações naturais como fome e doenças, ou ações humanas como as guerras. Em 1893, o naturalista inglês Charles Darwin adaptou o raciocínio de Malthus aos animais e às plantas e estabeleceu que as espécies competem intensamente pela sobrevivência e tal competição faz com que as características favoráveis dos indivíduos sejam transmitidas para sua prole. Deste modo, os descendentes se adaptarão gradualmente melhor ao ambiente do que seus antecessores e terão uma chance maior de sobrevivência.

Holland (1975), com base na teoria evolucionista de Darwin, propõe um método de busca global denominado Algoritmo Genético, o qual trabalha com uma população de indivíduos. Tais indivíduos são recombinados entre si através de cruzamento e herdam características dos pais que fazem com que eles tendam a ser mais aptos ao longo das gerações. Sob um ponto de vista matemático, os indivíduos podem ser vistos como possíveis valores de uma função e a aptidão pode ser expressa em termos do valor dessa função.

O trabalho de Goldberg (1989) incorreu em uma popularização dos Algoritmos Genéticos, sobretudo na área de Engenharia. Os AG's vêm sendo aplicados com êxito em diversos problemas de otimização, seja de funções não-lineares, seja em problemas combinatórios.

Os Algoritmos Genéticos são uma heurística de busca global que atua com dois tipos de paralelismo: um explícito e outro implícito. O paralelismo explícito consiste na geração e na avaliação de conjuntos de soluções, denominados populações, que permitem a busca em diferentes regiões do espaço de soluções. O paralelismo implícito consiste na geração de bons esquemas. Conforme Linden (2006), um esquema consiste em uma *template* descrevendo um subconjunto dentre o conjunto de todos os indivíduos possíveis. Para cada indivíduo da população, o AG manipula uma grande quantidade de esquemas (dezenas, centenas ou mesmo milhares) e vai selecionando os melhores dentre todos eles.

Os Algoritmos Genéticos combinam mecanismos de intensificação e diversificação. Por trabalhar com populações de soluções, o AG explora diferentes regiões do espaço de busca; e, por causa da pressão seletiva, o AG tende a migrar para regiões promissoras do espaço de soluções e concentrar a busca na recombinação de indivíduos de alta aptidão.

O pseudo-código de um AG genérico é ilustrado no Quadro1.

Quadro 1– Pseudo-código de um Algoritmo Genético padrão.

Passo 1: Gerar uma população de soluções.
Passo 2: Avaliar as soluções geradas.
Enquanto um critério de parada não for satisfeito, faça
    Passo 3: Selecionar um conjunto de k pais.
    Passo 4: Realizar o cruzamento de k pais, com uma dada probabilidade.
    Passo 5: Realizar mutação das w soluções geradas, com uma dada probabilidade.
    Passo 6: Avaliar a aptidão das soluções geradas.
    Passo 5: Atualizar a população.
Fim-do-enquanto
Imprimir a melhor solução obtida

A geração dos indivíduos da população geralmente é aleatória, mas uma heurística construtiva também pode se empregada para tanto. A avaliação de um indivíduo consiste na determinação do valor de sua função objetivo. A seleção prima por selecionar um conjunto de indivíduos aptos para serem recombinados. O cruzamento consiste na recombinação de um conjunto de pais, os quais trocam material genético e geram uma prole, a qual tende a ter uma maior aptidão. A mutação consiste em uma perturbação de uma solução prole obtida, de modo a garantir a diversificação da busca e mitigar uma possível convergência prematura do algoritmo. Por fim, a atualização da população consiste na regra de reposição das soluções, de uma geração para sua geração subseqüente, durante o processo de busca.

A seguir, no Quadro 2, é ilustrado o pseudo-código do Algoritmo Genético proposto para resolução do UQBP.

No AG proposto, é utilizada a codificação clássica (binária), conforme ilustrado na Figura 1. Seja um vetor de dimensão n, em que n é o número de variáveis de decisão, se $x_i = 1$ a i-ésima variável de decisão fará parte da solução; caso contrário, se $x_i = 0$, a i-ésima variável de decisão não fará parte da solução.

Quadro 2 – Pseudo-código do Algoritmo Genético proposto.

Passo 1: Gerar uma população de soluções aleatoriamente.
Passo 2: Avaliar as soluções geradas.
Enquanto ***geração atual ≤ número máximo de gerações*****, faça**
    Passo 3: Efetuar seleção por torneio.
    Passo 4: Realizar o cruzamento uniforme com p_c=100%.
    Passo 5: Realizar mutação das com p_m=1%.
    Passo 6: Realizar busca local 1-opt na solução gerada.
    Passo 6: Avaliar a aptidão da solução gerada.
    Passo 5: Substituir o pior indivíduo da população pela solução gerada.
Fim-do-enquanto
Imprimir a melhor solução obtida

gene	1	2	3	....	*n* - 1	*n*
bit	**1**	**0**	**0**	**...**	**1**	**1**

Figura 1– Codificação binária do cromossomo de um indivíduo.

A solução inicial é gerada de forma puramente aleatória: para cada gene de cada indivíduo da população, os valores 0 e 1 são equiprováveis, isto é, são sorteados com probabilidade igual a 50%.

A avaliação da aptidão de cada indivíduo da população, bem como de cada solução prole gerada, é dada no Quadro 3. Seja *npop* o tamanho da população e n a ordem da matriz Q do UQBP, a aptidão do k-ésimo indivíduo da população é obtida pela equação (1). É oportuno salientar que o cálculo do valor da função objetivo de um indivíduo tem complexidade computacional $O(n^2)$.

Quadro 3 – Procedimento para avaliação da solução.

```
para k←0 to npop-1 faça
    para i←0 to n-1 faça
        para ←0 to n-1 faça
            aptidão[k] ← aptidão[k]+Q[i,j]*pop[k,i]*pop[k,j];
        fim-do-para;
    fim-do-para;
fim-do-para;
```

A seleção dos pais a serem cruzados é dada por torneio: selecionam-se aleatoriamente 4 indivíduos da população e são realizados torneios com dois indivíduos. Os indivíduos mais aptos, isto é, aqueles com menores valores da função objetivo, serão escolhidos com pais.

O cruzamento é efetuado com probabilidade igual a 100%, ou seja, a cada geração, é gerado uma nova solução. O tipo de cruzamento utilizado foi o uniforme, ilustrado na Figura 2. Para cada gene da solução filho a ser gerada, é realizado um experimento, regido por uma distribuição de Bernoulli, no qual uma variável aleatória discreta b recebe um valor 0 ou 1. Se b igual a 0, o filho receberá o valor correspondente ao gene do primeiro pai. Se b igual a 1, o filho receberá o valor correspondente ao gene do segundo pai.

O cruzamento uniforme foi adotado pelo autor por dois motivos: (i) possui fácil implementação computacional; e (ii) é capaz de combinar todo e qualquer esquema existente.

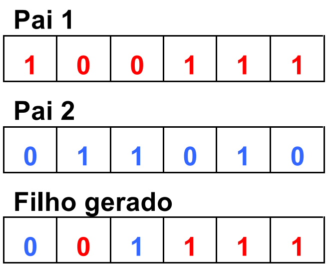

Figura 2 – Ilustração do cruzamento uniforme.

A operação de mutação ocorre da seguinte forma: é gerado um número aleatório compreendido no intervalo [0,100] e se este é menor ou igual à probabilidade de mutação considerada (no caso em estudo, arbitrou-se, após diversos experimentos, um valor p_m=1%), o operador de mutação é acionado. É gerado um número randômico r entre [0,n] e o bit correspondente ao número aleatório gerado é invertido (por exemplo, se $x_r = 1$ muda-se x_r para 0 e se $x_r = 0$ muda-se x_r para 1).

Sobre cada filho gerado, uma busca local do tipo 1-*opt* é realizada da seguinte forma: para cada gene da solução, inverte-se o valor do bit considerado. Se ocorrer melhoria (trata-se de uma estratégia *first improvement*) o gene é atualizado e a busca local prossegue.

4. Experimentos computacionais

Para os experimentos computacionais, foram utilizadas 5 instâncias de médio porte, as quais são propostas por Glover (http://hces.bus.olemiss.edu/tools.html). A descrição das instâncias é reportada na Tabela 1. As características das instâncias são as seguintes: ordem n da matriz Q, melhor solução conhecida, densidade da matriz Q, amplitude de valores dos elementos da diagonal principal e amplitude de valores dos elementos fora da diagonal principal. Deve-se destacar que o valor da densidade é obtido pela divisão do número de elementos não-nulos da matriz Q pelo número total de elementos da mesma.

Instância	n	Melhor solução conhecida	Densidade (%)	Diagonal principal	Demais elementos
f1a	500	-61194	10	[-75,+75]	[-50,+50]
f1b	500	-100161	25	[-75,+75]	[-50,+50]
f1c	500	-138035	50	[-75,+75]	[-50,+50]
f1d	500	-172771	75	[-75,+75]	[-50,+50]
f1e	500	-190507	100	[-75,+75]	[-50,+50]

Tabela 1 – Características das instâncias.

O Algoritmo Genético Híbrido foi implementado em linguagem Pascal. Os experimentos foram realizados em um Intel Celeron Dual-Core Inside 1.86 GHz, 1GB RAM. Foram utilizados os seguintes parâmetros: tamanho da população igual à 100 indivíduos, número de gerações igual à 1500 e probabilidade de mutação igual à 1%.

Para cada instância, foram feitas 5 execuções do algoritmo, sendo que tais resultados são expostos na Tabela 2. A primeira coluna apresenta a instância. A segunda coluna apresenta a replicação do algoritmo para cada instância. Na terceira coluna é apresentada a solução obtida pelo algoritmo. Na quarta coluna é apresentado o desvio percentual entre a solução obtida pela heurística e a melhor solução conhecida. Nas quinta e sexta colunas são expostos, respectivamente, a média e o desvio padrão dos desvios percentuais dos resultados do algoritmo. Por fim, na sétima coluna, são apresentados os tempos computacionais para cada replicação do algoritmo.

Instância	**Replicação**	**Solução**	**Desvio (%)**	**Média (%)**	**Desvio padrão (%)**	**tempo(s)**
f1a	1	-61083	-0,18	0,14	0,08	9,78
	2	-61194	0,00			10,02
	3	-61083	-0,18			9,78
	4	-61072	-0,20			10,59
	5	-61109	-0,14			9,34
f1b	1	-100005	-0,25	0,08	0,10	10,88
	2	-100128	-0,05			10,50
	3	-100158	0,00			10,67
	4	-100122	-0,06			10,03
	5	-100161	0,00			10,50
f1c	1	-137823	-0,35	0,41	0,22	10,44
	2	-137805	-0,38			10,05
	3	-137884	-0,25			10,23
	4	-137851	-0,30			10,09
	5	-137545	-0,80			10,24
f1d	1	-172216	-0,91	0,87	0,11	11,03
	2	-172232	-0,88			10,64
	3	-172137	-1,04			9,81
	4	-172313	-0,75			9,95
	5	-172290	-0,79			9,81
f1e	1	-190502	-0,01	0,09	0,12	10,91
	2	-190327	-0,29			10,84
	3	-190498	-0,01			10,36
	4	-190457	-0,08			10,63
	5	-190489	-0,03			9,92

Tabela 2 – Experimentos computacionais.

Diante dos experimentos computacionais realizados, os seguintes comentários podem ser salientados:

- Os desvios foram compreendidos em um intervalo [0,00;1,04], apontando para uma pequena discrepância entre os resultados obtidos pela heurística e as melhores soluções conhecidas.

- Em 3 das 25 replicações, a heurística conseguiu obter a melhor solução conhecida (vide instâncias f1a e f1b).

- As instâncias f1a, f1b e f1b mostram-se mais fáceis de serem solucionadas que as instâncias f1c e f1d.

- As médias dos desvios e dos tempos computacionais, para as 25 replicações realizadas, foram de 0,32% e 10,28s, respectivamente.

- Tendo em vista que as instâncias correspondem a matrizes de ordem igual a 500, tais resultados são considerados satisfatórios.

5. Conclusões

O presente artigo teve como objetivo reportar a aplicação de um Algoritmo Genético Híbrido para o problema quadrático binário irrestrito. A abordagem proposta foi testada em um conjunto de 5 instâncias médio porte presentes na literatura.

Os resultados reportados no presente artigo são apenas parciais. O autor está ampliando os testes, realizando experimentos em outros conjuntos de intâncias. Para uma validação do algorito proposto, este terá seu desempenho comparado com outras abordagens presentes na literatura.

O autor também está desenvolvendo aplicações do Algoritmo Genético Híbrido proposto para aplicações, em casos reais, do problema quadrático binário irrestrito.

Referências

ALKHAMIS, T. M., HASAN, M., AHMED, M. A. *Simulated annealing for the unconstrained quadratic pseudo-Boolean function.* European Journal of Operational Research, v.108, p. 641–652, 1998.

BEASLEY, J.B. *Heuristic algorithms for the unconstrained binary quadratic problem*. The Management School Imperial College of London: Technical Report, 1998.

GAREY, M. R., JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NPCompleteness*. New York: Freeman, 1979.

GOLDBERG, D. E. *Genetic Algorithms in search, optimization and machine learning*. Reading: Addison Wesley, 1989.

HOLLAND, J. H. *Adaptation in natural artificial systems*. Michigan: University of Michigan Press, 1975.

KATAYAMA, K., NARIHISA, H. *Performance of simulated annealing-based heuristic for the unconstrained binary quadratic programming problem*. European Journal of Operational Research, v.119, p. 662–670, 1999.

KATAYAMA, K., TANI, M.; NARIHISA, H. *Solving large binary quadratic programming problems by effective genetic local search algorithm.* In GECCO-200: Proceedings of the Genetic and Evolutionary Computation Conference, Las Vegas, 2000.

KOCHENBERGER, G. A., GLOVER, F., ALIDAEE, B., REGO, C. *A unified modeling and solution framework for combinatorial optimization problems.* OR Spectrum, v.26, p. 237–250, 2004.

LINDEN, R. *Algoritmos Genéticos – uma importante ferramenta da inteligência computacional.* Rio de Janeiro: Brasport, 2006.

LODI, A., ALLEMAND, K., LIEBELING, T. M. *An evolutionary heuristic for quadratic 0-1 programming.* European Journal of Operational Research, v.119, p. 662–670, 1999.

MERZ, P., FREISLEBEN, N. *Genetic algorithms for binary quadratic programming.* In GECCO-1999: Proceedings of the Genetic and Evolutionary Computation Conference, Florida,1999.

MERZ, P., FREISLEBEN, N. *Greedy and local search heuristics for the unconstrained binary quadratic programming problem.* Journal of heuristics, v.8, p. 197–213, 2004.

MERZ, P., KATAYAMA, K. *Memetic algorithms for the unconstrained binary quadratic programming problem.* BioSystems, v.78, p. 99–118, 2004.

PALUBECKIS, G.; TOMKEVICIUS, A. *GRASP implementations for the unconstrained binary quadratic optimization problem.* Information Technology and Control, v.24, p. 14–20, 2002.

PALUBECKIS, G. *Multistart Tabu Search for the unconstrained binary quadratic optimization problem.* Annals of Operations Research, v.131, p. 259–282, 2004.

PALUBECKIS, G. *Iterated Tabu Search for the unconstrained binary quadratic optimization problem.* Informatica, v.17, p. 279–296, 2006.

SCHÜTZ, G., PIRES, F. *A genetic based algorithm for the quadratic 0-1 problem.* Investigação Operacional, v.23, p. 71–84, 2003.