CNMAI Congresso Nacional de Matemática Aplicada à Indústria 2014
18 a 21 de novembro de 2014, Caldas Novas - Goiás

APLICANDO O ALGORITMO GENÉTICO DE CHAVES ALEATÓRIAS VICIADAS EM UM PROBLEMA DE CORTE COM ITENS IRREGULARES

Leandro Resende Mundim, mundim@icmc.usp.br [1]
Marina Andretta, andretta@icmc.usp.br[1]
Thiago Alves de Queiroz, taq@ufg.br [2]

[1]Universidade de São Paulo - Instituto de Ciências Matemáticas e de Computação.
Av. Trabalhador São-Carlense, 400, São Carlos - SP.

[2]Universidade Federal de Goiás - Departamento de Matemática.
Av. Dr. Lamartine Pinto de Avelar, 1120, Catalão - GO.

Resumo. *Um dos problemas que ainda tem sido pouco explorado na literatura de corte de itens irregulares (polígonos convexos e não convexos) é o problema da mochila 0-1 (2PMI). No 2PMI é requerido que um subconjunto de itens seja empacotado em um recipiente retangular, com o objetivo de maximizar a ocupação da área. Os itens devem estar totalmente contidos no recipiente e nenhum deles pode se sobrepor com algum outro. Neste trabalho propomos uma abordagem baseada no algoritmo genético de chaves aleatórias viciadas partindo de um framework proposto na literatura. O diferencial está na técnica de alocação, que combina os cantos da mochila, utilizando o canto inferior esquerdo ou o canto superior esquerdo. Para encontrar posições viáveis e evitar a sobreposição, utilizamos uma malha de pontos extraída da técnica de no-fit polygon. Os experimentos computacionais mostraram que o algoritmo proposto é competitivo, conseguindo melhorar resultados da literatura.*

Palavras-chave: *Problemas de corte bidimensional; Problema da mochila 0-1; Itens irregulares; Algoritmo genético; Chaves viciadas; No-fit polygon.*

1. INTRODUÇÃO

Problemas de corte de itens irregulares (polígonos convexos e não convexos), também conhecidos como problemas de *nesting*, são problemas em que um conjunto de itens irregulares deve ser cortado, sem sobreposição, a partir de um recipiente maior, em geral visando minimizar o desperdício do recipiente. Esses problemas estão na área de otimização combinatória e aparecem com muita frequência na indústria têxtil durante o corte de peças de roupa e tecido. Outras aplicações, como cortar placas de metal, peças de couro e corte de espuma, aparecem nas indústrias metalmecânica, calçadista e moveleira. As transportadoras enfrentam este problema ao terem que empacotar produtos sem sobreposição em contêineres e durante o transporte de estátuas e peças de museu.

Apesar de sua aparente simplicidade, problemas de corte, em geral, são NP-difíceis, como é o caso do problema aqui investigado (Garey e Johnson, 1979). Além da importância econômica (redução de custos), este problema tem grande importância ambiental. Ao reduzir o desperdício do material utilizado, estamos melhorando a utilização de recursos naturais, como o alumínio, o couro, a madeira e o vidro. Na intenção de melhorar a utilização desses recursos, diversas técnicas de resolução para problemas de corte de itens irregulares vêm sendo desenvolvidas, em geral heurísticas e metaheurísticas.

Uma boa parcela dos métodos desenvolvidos é para o problema de empacotamento em faixa (*strip packing*) irregular. Neste caso, todos os itens devem ser empacotados na faixa visando minimizar a dimensão (comprimento), que é considerada infinita. No trabalho de Oliveira et al. (2000), os autores apresentaram um algoritmo construtivo para o *strip packing*, definindo cinco critérios para ordenar os itens e vários critérios de alocação, transformando a sequência em um leiaute viável. Heurísticas construtivas, em que é dada uma sequência para empacotar os itens, utilizando a estratégia de empacotar olhando o canto inferior esquerdo (*bottom-left*), podem ser encontradas em Dowsland et al. (2002) e Burke et al. (2006).

Jakobs (1996) apresentou um dos primeiros algoritmos genéticos para o *strip packing* irregular. O autor criou duas instâncias artificiais e aplicou o algoritmo genético para empacotar a envoltória retangular dos itens por meio de uma heurística *bottom-left*, além de ter compactação do leiaute para melhorar a solução. Foi utilizada a ordem em que os itens estão alocados na heurística *bottom-left*, observando a sequência em que eles são dispostos no cromossomo. A estratégia híbrida proposta por Gomes e Oliveira (2006) melhorou vários resultados da literatura. Ela consiste de uma meta-heurística de recozimento simulado que utiliza modelos lineares de separação e compactação. Vale destacar que esta estratégia também utiliza a técnica de *bottom-left* discreta para gerar um leiaute inicial. Elkeran (2013) combinou uma meta-heurística recente, denominada *cuckoo search* de Yang e Deb (2010), com uma busca local direcionada na tentativa para escapar de mínimos locais. Também, foram propostos modelos matemáticos para a compactação do leiaute e algoritmos de separação para retirar as possíveis sobreposições existentes. Com o algoritmo de Elkeran (2013) foi possível melhorar muitos resultados relacionados ao *strip packing* irregular.

Existem poucos trabalhos que utilizam métodos exatos para resolver problemas de *nesting*. Dentre eles, destacam-se: um método baseado em programação por restrições em Carravilla et al. (2003) e um modelo de programação inteira mista de Fischetti e Luzzi (2009). Alvarez-Valdes et al. (2013) reformularam o modelo de Fischetti e Luzzi (2009) e propuseram um algoritmo *branch-and-bound* para o *strip packing*, além de estender o modelo de compactação de Gomes e Oliveira (2006) para itens convexos. (Toledo et al., 2013) apresentaram um modelo inteiro misto e resolveram as maiores instâncias da literatura para algoritmo exatos, entretanto a solução é ótima para uma malha de pontos.

No presente trabalho, estudamos um dos problemas que ainda tem sido pouco explorado na literatura de corte de itens irregulares, que é o problema da mochila 0-1 (2PMI). No 2PMI é requerido que um subconjunto de itens irregulares seja empacotado em um recipiente retangular, com altura e largura definidos, com o objetivo de maximizar a utilização de área. Os itens devem estar totalmente contidos no recipiente e nenhum item pode se sobrepor a outro. Trabalhos recentes que envolvem variações do problema da mochila podem ser encontrados em Del Valle (2010), Del Valle et al. (2012), Mundim e Queiroz (2012), Silveira (2013) e Dalalah et al. (2014). Del Valle (2010) propôs um algoritmo híbrido para o problema da mochila irrestrita e uma heurística baseada em GRASP para o 2PMI. Uma solução inicial gulosa é feita empacotando, dentre uma lista aleatória de itens não empacotados, o item que tenha o melhor custo benefício no empacotamento. Este custo depende da ocupação retangular e da área da envoltória retangular do empacotamento corrente. Del Valle et al. (2012) utilizaram as mesmas abordagens de (Del Valle, 2010) e obtiveram resultados para um outro problema, o de corte de estoque bidimensional. Mundim e Queiroz (2012) resolveram o 2PMI com um algoritmo híbrido, envolvendo GRASP com recozimento simulado. Os autores utilizaram a rotina de empacotamento de Del Valle et al. (2012) combinada com o recozimento simulado para melhorar a busca local do GRASP, com o objetivo de escapar de mínimos locais. Silveira (2013) apresentou duas novas heurísticas, baseadas em um algoritmo genético, para resolver o *strip packing* irregular. As heurísticas foram adaptadas para o 2PMI, uma vez que dependem basicamente da sequência em que os itens devem ser empacotados. Como no 2PMI as dimensões do recipiente são fixas, quando o próximo item da sequência não puder ser empacotado, o mesmo é descartado da solução. Dalalah et al. (2014) recentemente apresentaram uma abordagem para lidar com o 2PMI. A estratégia é baseada na união de polígonos, sempre visando minimizar o desperdício. A principal contribuição do algoritmo de Dalalah et al. (2014) é que, além da mochila retangular, os autores propuseram uma estratégia genérica para uma mochila também com forma irregular.

Propomos no presente trabalho uma abordagem baseada no algoritmo genético de chaves aleatórias viciadas, partindo do *framework* de Toso e Resende (2011), para resolver o 2PMI. Inicialmente, um total de P indivíduos são gerados. Cada indivíduo é formado por um cromossomo de N posições e um vetor com N chaves. As posições do cromossomo indicam a ordem em que os itens serão empacotados no recipiente. Atribui-se um valor aleatório no intervalo real [0, 1] para cada chave. Este valor diz se o item é alocado no canto inferior esquerdo (valor maior que 0,7) ou se o item é alocado no canto superior esquerdo da mochila. Em seguida, geramos, para cada indivíduo, um empacotamento viável. Para tanto, os itens são empacotados na ordem com que estão no cromossomo e as chaves informam se o item será empacotado mais próximo do canto superior esquerdo ou no canto inferior esquerdo. Caso o item não possa ser empacotado no recipiente, ele será descartado do empacotamento corrente. Como o objetivo do 2PMI é maximizar a ocupação de área da mochila, a função de aptidão utilizada consiste na área ocupada do recipiente. Após a criação do leiaute dos P indivíduos, criamos um grupo de E indivíduos elites, que são utilizados na geração de novos indivíduos. Além disso, são gerados M novos indivíduos de forma totalmente aleatória, formando o subgrupo dos mutantes. Para completar a população com os P indivíduos, filhos são gerados por meio da operação de cruzamento envolvendo as chaves de um indivíduo elite e outro selecionado do restante da população. O cruzamento consiste em pegar a chave do pai elite, com probabilidade de 70%, ou do outro pai, até completar as N chaves. Esse procedimento é repetido a cada nova geração e continua até atingir um número máximo de iterações. A estratégia utilizada para evitar a sobreposição consiste no método *raster* combinado com o *no-fit polygon*, na qual definimos uma malha de pontos estritamente dentro do *no-fit polygon*.

O presente trabalho está organizado da seguinte forma. A Seção 2 apresenta a estratégia utilizada para evitar a sobreposição de itens. A Seção 3 apresenta as rotinas usadas dentro do *framework*. A Seção 4 discute os experimentos computacionais e a Seção 5 apresenta as conclusões e direções para trabalhos futuros.

2. CONCEITOS GEOMÉTRICOS

A geometria irregular dos itens é um desafio e diversas estratégias vêm sendo desenvolvidas para tratá-la. Um recente tutorial sobre as técnicas mais utilizadas é apresentado em Bennell e Oliveira (2008). As estratégias mais utilizadas consistem no método *raster*, trigonometria direta, *phi function* e *no-fit polygon*.

A idéia do método *raster* é discretizar os itens em uma malha, para verificar, por meio de uma soma de matrizes, a sobreposição. A desvantagem desta abordagem é que a qualidade da representação depende do refinamento da malha, ou seja, quanto menor a discretização dos itens, mais próximo do item real e mais custosa são as operações sobre as suas representações. Em todos as outras abordagens, os itens são polígonos, o que deixa a representação geométrica precisa.

A trigonometria direta utiliza os vértices e as arestas de um item para verificar sua sobreposição com outro item. Para isso, faz até quatro testes, a saber: (1) verifica a sobreposição da menor envoltória retangular dos itens e se não existir sobreposição, os itens estão separados; (2) verifica a sobreposição entre todos os pares de arestas e caso não exista sobreposição, os itens estão separados; (3) verifica se existe interseção entre as arestas dos itens e se existir interseção, os itens estão sobrepostos; (4) verifica se cada um dos vértices de cada item está contido em outro item e caso os vértices não estejam contidos, garantimos que os itens estão separados. A Fig. 1 apresenta dois itens ao lado esquerdo, um polígono convexo e o outro polígono não convexo. Esses polígonos são utilizados sem modificação na trigonometria direta. Por outro lado, no método *raster*, tem-se do lado direito da Fig. 1 como fica a discretização dos dois polígonos.

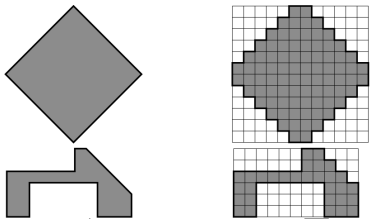


Figura 1. Exemplo de polígonos no método *raster*.

A *phi function* é uma expressão matemática que informa a posição relativa entre um par de itens quaisquer. A principal dificuldade deste método é a falta de um procedimento robusto para construir as expressões. Para interpretar o valor da função existem: valores positivos, indicando itens separados, valores negativos, indicando a sobreposição, e o valor zero, que indica que os itens estão se tocando.

O *no-fit polygon* é utilizado para verificar se pares de itens estão se tocando, separados ou se sobrepõem. Esta técnica mantém todas as posições viáveis da trigonometria direta e reduz os cálculos a partir de uma etapa de pré-processamento, em que se faz o cálculo do *no-fit polygon* para todos os pares de itens e em todas as suas rotações. Em linhas gerais, fixamos um item no plano e deslocamos o outro item, dado um ponto de referência determinado, ao redor do item fixo. O caminho feito pelo ponto de referência forma uma ou mais regiões (polígonos convexos e não convexos) do *no-fit polygon*.

Uma vez calculado o *no-fit polygon* entre dois itens, para verificar se há sobreposição entre estes itens, torna-se suficiente verificar a distância do ponto de referência do item orbital para todas as regiões. Se o ponto de referência estiver no interior de alguma região, os itens estão se sobrepondo. Se o ponto de referência estiver do lado de fora de todas as regiões, os itens estão separados. Se o ponto de referência estiver de fora das regiões e sobre os limites de pelo menos uma região, os itens estão encostados.

A abordagem utilizada neste trabalho é uma matriz baseada nos pontos do *no-fit polygon*. Se for usada apenas uma matriz com o método *raster*, a discretização de cada uma dos itens seria feita em separado e, assim, perderíamos todos os encaixes não ortogonais. Para facilitar o entendimento, veja na Fig. 2 uma solução obtida com o método *raster* da instância proposta por Oliveira et al. (2000) com quatro itens e uma mochila com altura 12 e largura 26.

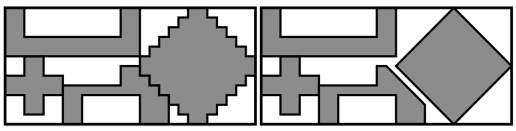


Figura 2. Solução obtida usando o método *raster*.

Notamos na Fig. 2, que utilizando o método *raster*, não se consegue uma solução melhor. Caso a mochila seja uma unidade menor, temos que a solução poderia ser melhorada. O losango poderia ser deslocado um pouco para a esquerda até encostar em algum item, assim todos os itens poderiam ser empacotados na mochila reduzida.

Com a abordagem adotada neste trabalho, baseada em Toledo et al. (2013), definimos a malha de pontos sobre o *no-fit polygon*. Os itens continuam sendo representados por polígonos, mas o *no-fit polygon* é representado por uma malha. Para gerar os pontos da malha, implementamos um método de triangulação simples (O'Rourke, 1998) e calculamos o *no-fit polygon* parcial entre todos os triângulos utilizando o algoritmo de ordenação de arestas de Cuninghame-Green (1989)

sobre polígonos convexos. A melhora da representação pode ser observada na Fig. 3, em que conseguimos empacotar todos os itens em uma mochila de altura 12 e largura 25, utilizando a mesma configuração apresentada na Fig. 2.

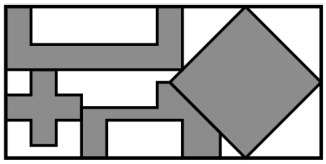


Figura 3. Solução usando o método proposto neste trabalho.

3. ALGORITMO GENÉTICO DE CHAVES ALEATÓRIAS VICIADAS (BRKGA)

Algoritmos genéticos foram introduzidos por Holland (1975) e, desde então, diversos trabalhos e técnicas de otimização foram desenvolvidos a partir deles. Em linhas gerais, o algoritmo genético recebe as características dos pais e as passam para os filhos durante a fase de cruzamento. Gonçalves e Resende (2011) propuseram uma nova versão do algoritmo genético, aquela que inclui chaves aleatórias viciadas, sendo uma melhoria do algoritmo de Bean (1994).

A principal diferença entre o algoritmo genético de Bean (1994) e o algoritmo de chaves aleatórias viciadas de Gonçalves e Resende (2011) está na maneira como o cruzamento é realizado. No algoritmo de chaves viciadas, durante o cruzamento existe um favorecimento dos pais com a melhor função de aptidão, permitindo a geração de indivíduos "viciados" mais promissores. No presente trabalho, usamos uma abordagem baseada no algoritmo genético de chaves aleatórias viciadas partindo do *framework* de Toso e Resende (2011). A Tabela 1 traz todos os parâmetros utilizados pelo algoritmo. Foram feitos experimentos iniciais de calibração dos parâmetros, mas verificamos que os melhores parâmetros foram os já pré-definidos no *framework*.

Tabela 1. Parâmetros utilizados no *bottom-top*-BRKGA

Parâmetro	Descrição	Valores Padrão
N	Número de elementos por cromossomo	quantidade total de itens
P	Tamanho da população	1000
E	Tamanho da população elite	200
M	Número de mutantes em cada geração	100
C	Probabilidade de uma chave ser herdada do pai elite	70%

Jakobs (1996) apresentou um algoritmo genético para o problema de *nesting*, em que cada indivíduo é representado por um cromossomo, com a sequência dos itens a empacotar. Neste trabalho, a ordem em que os itens são alocados pela heurística *bottom-left* é dada pela sequência em que eles estão dispostos no cromossomo de cada indivíduo. Diferente do trabalho de Jakobs (1996) que procurava um empacotamento viável a partir do canto inferior esquerdo, procuramos a partir do canto inferior esquerdo e do canto superior esquerdo.

No algoritmo proposto há um total de P indivíduos, em que cada indivíduo é constituído por um cromossomo (um vetor de chaves), ambos com N posições. No cromossomo é dada a ordem com que os itens serão empacotados (inspirado em Jakobs (1996)). O vetor com as chaves possuem um valor aleatório uniforme no intervalo real $[0, 1]$, para cada chave. O valor da chave diz se o item será empacotado pela heurística *top-left* ou pela *bottom-left*. Em outras palavras, se a chave for maior do que 0,7, o item é alocado no canto superior esquerdo, caso contrário, é alocado no canto inferior esquerdo. O valor 0,7 para alocar o item pela heurística *bottom-left* foi obtido por meio de experimentos iniciais. Vale destacar que este valor é importante para "viciar" a construção da solução.

Para decodificar o indivíduo e gerar um empacotamento viável, empacotamos os itens na mochila respeitando a ordem dada no cromossomo e o valor das chaves associadas, não empacotando os itens que excedem as dimensões da mochila. Retornamos a ocupação da mochila como o valor da função de aptidão do referido indivíduo. Por meio da função de aptidão, criamos um subgrupo de E indivíduos elites. Em seguida, M novos indivíduos são gerados de forma aleatória, formando o subgrupo dos mutantes. Para completar a população de P indivíduos, filhos são gerados por meio da operação de cruzamento entre um indivíduo elite e outro selecionado do restante da população. O cruzamento consiste em pegar a chave do pai elite, com probabilidade de 70%, ou do outro pai, até completar as N chaves. Esses passos são repetidos a cada nova geração e continua até atingir um número máximo de iterações. Neste trabalho, limitamos o algoritmo a 1000 gerações, mas caso o algoritmo encontre a solução ótima, ele é finalizado. Uma solução é considerada ótima quando todos os itens são empacotados no recipiente.

Vamos a um exemplo de como decodificar um indivíduo. Neste exemplo é dada uma mochila de altura 31 e largura 14 e quatro itens, conforme ilustra a Fig. 4. Os números (1, 2, 3, 4) na parte de baixo dos polígonos representam o identificador de cada um.

Em seguida, considere o cromossomo de um indivíduo dado por [4 2 1 3]. Este cromossomo informa que o primeiro

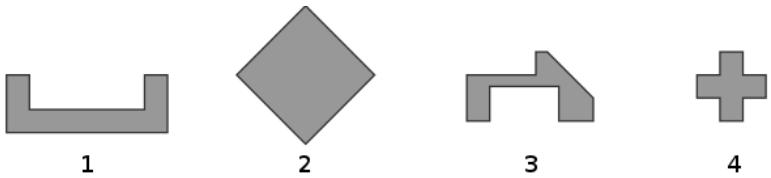


Figura 4. Itens da instância *SHAPES0*.

item a ser empacotado é o item 4, seguido pelos itens 2, 1 e 3 respectivamente. Além do cromossomo, todo indivíduo possui um vetor de chaves. Neste exemplo, o vetor dado corresponde a [0.8 0.6 0.8 0.9]. Observe que apenas o item 2 foi empacotado na parte inferior, enquanto os outros três itens foram empacotados pela heurística *top-left*. A Fig. 5 ilustra a solução do nosso exemplo.

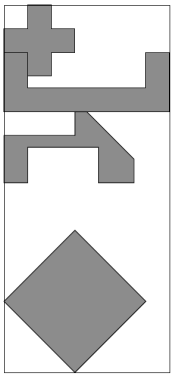


Figura 5. Solução obtida com o *framework* usando a estratégia apresentada neste trabalho.

4. EXPERIMENTOS COMPUTACIONAIS

Os algoritmos utilizados no presente trabalho foram implementados na linguagem C++, a mesma linguagem do *framework* de Toso e Resende (2011), viabilizando a comunicação entre as nossas rotinas e o que já estava no *framework*. Todos os experimentos foram feitos em um computador com processador Intel Core 2 Duo 2.66 GHz e 4 GB de memória RAM, no sistema Linux Ubuntu 14.04.

Para verificar o desempenho do algoritmo desenvolvido, os experimentos foram separados em dois grupos. No primeiro, utilizamos onze instâncias da literatura e os resultados foram comparados com os trabalhos de Del Valle et al. (2012) e Silveira (2013). O segundo apresenta os resultados que são comparados com os experimentos de Dalalah et al. (2014). Vale destacar que, para simplificar a análise e facilitar a reprodução dos resultados, utilizamos apenas a semente padrão já definida no *framework* de Toso e Resende (2011). A Tabela 2 apresenta os dados das instâncias utilizadas. Esta tabela foi baseada nos trabalhos de Del Valle et al. (2012) e Dalalah et al. (2014).

Tabela 2. Informações sobre as instâncias utilizadas.

Instância	Altura	Largura	Rotações Permitidas
FU	38,00	34,00	0º; 90º; 180º
JACKOBS1	40,00	13,00	0º; 90º; 180º
JACKOBS2	70,00	28,20	0º; 90º; 180º
SHAPES0	40,00	63,00	0º
SHAPES1	40,00	59,00	0º; 180º
SHAPES2	15,00	27,30	0º; 180º
DIGHE1	100,00	138,13	0º
DIGHE2	100,00	134,05	0º
MARQUES	104,00	83,60	0º; 90º; 180º
SHIRTS	40,00	63,13	0º; 180º
TROUSERS	79,00	245,75	0º; 180º
EXEMPLO 2	10,00	6,00	0º; 90º; 180º; 270º
EXEMPLO 3	17,00	17,00	0º

A Tabela 3 apresenta os resultados obtidos por Del Valle et al. (2012), pelas duas heurísticas propostas em Silveira

(2013) e pelo algoritmo aqui proposto, que denotamos por *bottom-top*-BRKGA. Nesta tabela, apresentamos, para cada algoritmo, a ocupação da mochila e o tempo computacional gasto. As melhores taxas de ocupação para cada instância estão em negrito. O asterisco em frente ao nome das instâncias significa que o nosso algoritmo conseguiu encontrar a solução ótima para o 2PMI, ou seja, foi possível empacotar todos os itens na mochila.

Tabela 3. Resultados para o 2PMI e comparação com a literatura.

Instância	Del Valle et al. (2012)		H1-Silveira (2013)		H2-(Silveira, 2013)		*bottom-top*-BRKGA	
	Ocupação	Tempo (s)	Ocupação	Tempo (s)	Ocupação	Tempo (s)	Ocupação	Tempo (s)
FU*	**0.8382**	21.79	**0.8382**	13.20	**0.8382**	69.70	**0.8382**	3.03
JACKOBS1*	**0.7538**	8.30	**0.7538**	13.60	**0.7538**	10.04	**0.7538**	3.86
JACKOBS2*	**0.6844**	565.71	**0.6844**	38.80	**0.6844**	32.90	**0.6844**	14.52
SHAPES0*	0.6016	1552.46	0.6175	1200.40	0.6111	1201.60	**0.6332**	15.23
SHAPES1*	0.6424	3891.60	0.6559	1202.00	0.6559	1203.10	**0.6763**	6.12
SHAPES2	0.7289	1048.12	0.7643	1200.20	**0.7766**	1201.00	0.7375	503.00
DIGHE1*	**0.7240**	10.68	**0.7240**	31.60	**0.7240**	18.00	**0.7240**	17.01
DIGHE2*	**0.7460**	0.07	**0.7460**	1.70	**0.7460**	7.80	**0.7460**	8.61
MARQUES*	**0.8274**	217.77	**0.8274**	58.20	**0.8274**	63.20	**0.8274**	52.05
SHIRTS*	0.7702	14317.13	0.8471	1207.50	0.8389	1220.10	**0.8553**	2750.33
TROUSERS	0.7866	5796.59	0.8745	1202.00	**0.8830**	1206.40	0.8201	4152.60

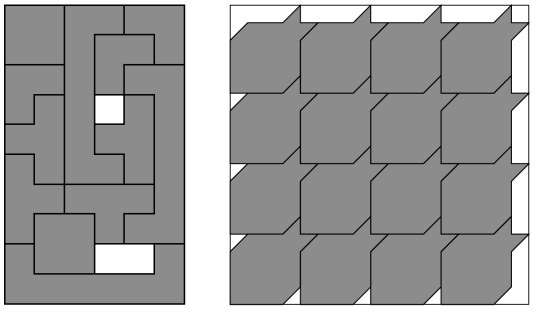
Observando a Tabela 3, não conseguimos a solução ótima apenas para as instâncias SHAPES2 e TROUSERS. Para todas as demais, o *bottom-top*-BRKGA conseguiu resultados iguais ou melhores. Além disso, computou a solução ótima para as instâncias SHAPES0, SHAPES1 e SHIRTS. Vale mencionar que utilizamos uma máquina diferente de Del Valle et al. (2012) e Silveira (2013). Por isso, a comparação do tempo computacional foi evitada.

Podemos analisar que o tempo do nosso algoritmo é pequeno para as instâncias em que alcançou a otimalidade, exceto para SHIRTS. Além disso, vale destacar que as instâncias com dimensões pequenas, como FU, SHAPES0 e SHAPES1, foram resolvidas em um tempo menor. Isso aconteceu pelo fato dos polígonos pequenos serem vantajosos para o *bottom-top*-BRKGA, devido a estratégia de não sobreposição adotada. Por outro lado, instâncias com polígonos grandes e mais complexos, como a TROUSERS, necessitaram de mais tempo de processamento.

O trabalho de Dalalah et al. (2014) apresenta seis experimentos computacionais para o 2PMI. Porém, apenas os experimentos número dois e três utilizam uma mochila retangular, para fins de comparação de resultados. O resultado para as duas instâncias estão sumarizados na Tabela 4. O desempenho do *bottom-top*-BRKGA, comparado com o algoritmo de (Dalalah et al., 2014), foi superior para o EXEMPLO 2 e igual para o EXEMPLO 3. As melhores taxas de ocupação estão em negrito. As soluções do *bottom-top*-BRKGA são apresentadas nas Figs. 6 e 7.

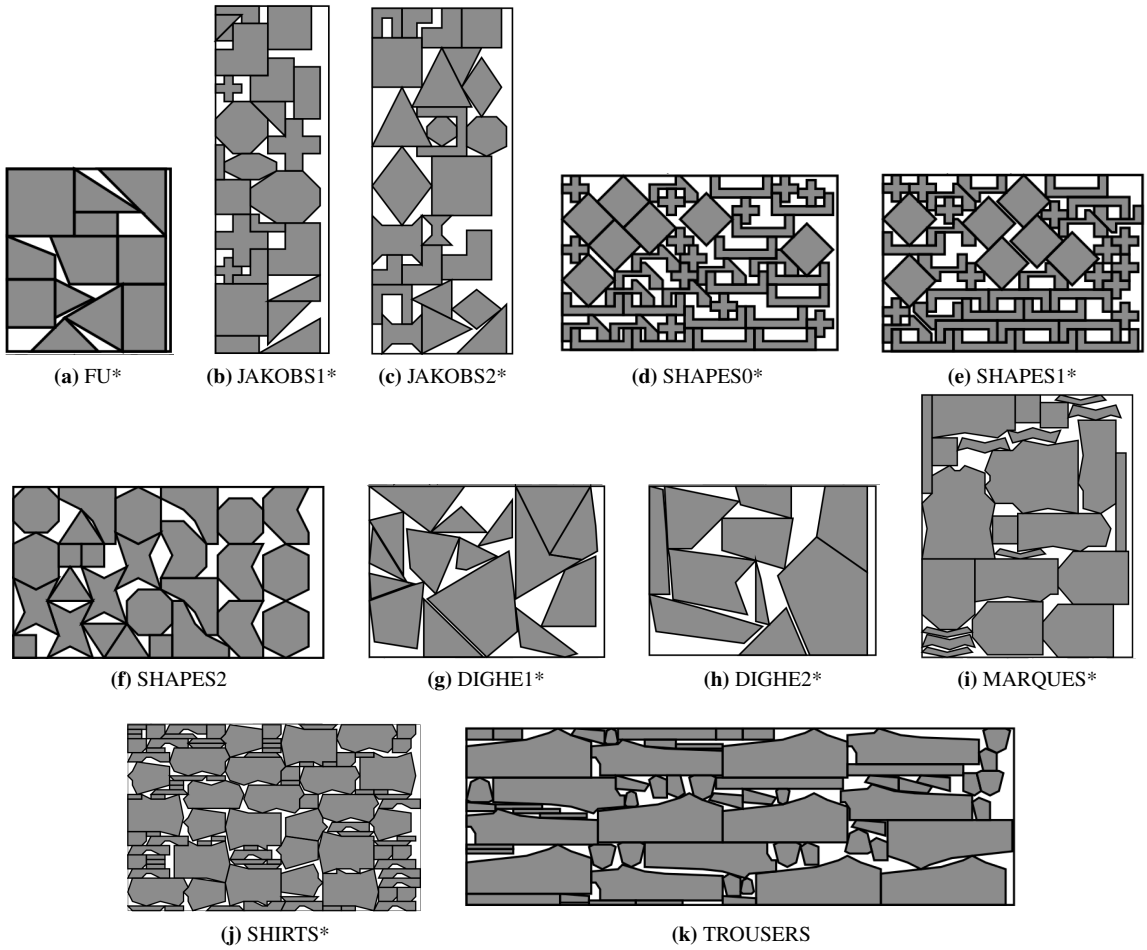
Tabela 4. Resultados comparativos para os experimentos em Dalalah et al. (2014).

Instância	Dalalah et al. (2014)		*bottom-top*-BRKGA	
	Ocupação	Tempo (s)	Ocupação	Tempo (s)
EXEMPLO 2	0.90	11.50	**0.95**	65.00
EXEMPLO 3	**0.89**	12.40	**0.89**	0.02



(a) EXEMPLO 2 **(b)** EXEMPLO 3

Figura 6. Soluções encontradas pelo *bottom-top*-BRKGA.



(a) FU* **(b)** JAKOBS1* **(c)** JAKOBS2* **(d)** SHAPES0* **(e)** SHAPES1*

(f) SHAPES2 **(g)** DIGHE1* **(h)** DIGHE2* **(i)** MARQUES*

(j) SHIRTS* **(k)** TROUSERS

Figura 7. Soluções encontradas pelo *bottom-top*-BRKGA.

5. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho apresenta uma abordagem baseada no algoritmo genético de chaves aleatórias viciadas para o problema da mochila 0-1 com itens irregulares. Implementamos a técnica de verificação de não sobreposição de Toledo et al. (2013), na qual o *no-fit polygon* é discretizado em uma malha de pontos. Por outro lado, o algoritmo genético parte do *framework* de Toso e Resende (2011).

O *framework* é muito dependente do algoritmo de decodificação utilizado. Neste trabalho, propomos um algoritmo simples para decodificar os indivíduos. Cada indivíduo é formado por um cromossomo que fornece a sequência de itens a empacotar e um vetor de chaves viciadas. A sequência dos itens no cromossomo representa a sequência com que os itens serão empacotados na mochila, enquanto que o vetor com as chaves viciadas referem a estratégia utilizada para empacotar cada item. Utilizamos as estratégias do canto inferior esquerdo (*bottom-left*) e canto superior esquerdo (*top-left*).

O algoritmo proposto foi testado e comparado com outros de três trabalhos da literatura. E, dividimos os testes em dois grupos, de forma que, das onze instâncias testadas no primeiro, conseguimos resultados melhores ou iguais para nove delas, incluindo soluções ótimas. Para o segundo grupo, que considerava apenas duas instâncias, foi possível obter uma solução igual e outra melhor. Com isso, acreditamos que estes experimentos são ponto de partida para investigações futuras, bem como para validar o bom funcionamento da estratégia quando resolvendo o 2PMI.

Como trabalhos futuros, pretendemos estender o algoritmo para recipientes irregulares, testar novos algoritmos de decodificação dentro do *framework* e considerar outros problemas, como o *strip packing* irregular.

Agradecimentos

Este trabalho contou com o apoio financeiro da CAPES, CNPq (processos 476792/2013-4 e 471351/2012-1), FAPESP (processos 2010/10133-0 e 2013/07375-0) e FAPEG.

REFERÊNCIAS

Alvarez-Valdes, R., Martinez, A. e Tamarit, J. 2013. A branch and bound algorithm for cutting and packing irregularly shaped pieces. *International Journal of Production Economics*, *145*(2), 463–477.

Bean, J. C. 1994. Genetic algorithms and random keys for sequencing and optimization. *ORSA journal on computing*, *6*(2), 154-160.

Bennell, J. A. e Oliveira, J. F. 2008. The geometry of nesting problems: A tutorial. *European Journal of Operational Research*, *184*, 397–415.

Burke, E., Hellier, R., Kendall, G. e Whitwell, G. 2006. A new bottom-left-fill heuristic algorithm for the two-dimensional irregular packing problem. *Operations research*, *54*(3), 587–601.

Carravilla, M. A., Ribeiro, C. e Oliveira, J. F. 2003. Solving nesting problems with non-convex polygons by constraint logic programming. *International Transactions in Operational Research*, *10*(6), 651–663.

Cuninghame-Green, R. 1989. Geometry, shoemaking and the milk tray problem. *New Scientist*, *123*, 6–20.

Dalalah, D., Khrais, S. e Bataineh, K. 2014. Waste minimization in irregular stock cutting. *Journal of Manufacturing Systems*, *33*, 27–40.

Del Valle, A. M. 2010. Problema da mochila com itens irregulares. *Dissertação de Mestrado*, Universidade Estadual de Campinas (Unicamp) - SP.

Del Valle, A. M., Queiroz, T. A., Miyazawa, F. K. e Xavier, E. C. 2012. Heuristics for two-dimensional knapsack and cutting stock problems with items of irregular shape. *Expert Systems with Applications*, *39*, 12589–12598.

Dowsland, K. A., Vaid, S. e Dowsland, W. B. 2002. An algorithm for polygon placement using a bottom-left strategy. *European Journal of Operational Research*, *141*, 371–381.

Elkeran, A. 2013. A new approach for sheet nesting problem using guided cuckoo search and pairwise clustering. *European Journal of Operational Research*, *231*, 757–769.

Fischetti, M. e Luzzi, I. 2009. Mixed-integer programming models for nesting problems. *Journal of Heuristics*, *15*(3), 201–226.

Garey, M. R. e Johnson, D. S. 1979. *Computers and intractability: a guide to the theory of np-hardness*. Freeman, San Francisco. 124–127.

Gomes, A. M. e Oliveira, J. F. 2006. Solving irregular strip packing problems by hybridising simulated annealing and linear programming. *Journal of the Operational Research Society*, *171*, 811–829.

Gonçalves, J. F. e Resende, M. G. 2011. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, *17*(5), 487–525.

Holland, J.H. 1975. *Adaptation in natural and artificial systems*. University of Michigan Press.

Jakobs, S. 1996. On genetic algorithms for the packing of polygons. *European Journal of Operational Research*, *88*, 165–181.

Mundim, L. R. e Queiroz, T. A. 2012. A hybrid heuristic for the 0-1 knapsack problem with items of irregular shape. *In Informatica (CLEI), 2012 XXXVIII Conferência Latino-americana.* , 1–6.

Oliveira, J. F., Gomes, A. M. e Ferreira, J. S. 2000. Topos - A new constructive algorithm for nesting problems. *OR Spectrum*, *22/*, 263–284.

O'Rourke, J. 1998 *Computational Geometry in C*. Cambridge. 33–43.

Silveira, T. 2013. Problemas de empacotamento com itens irregulares: Heurísticas e avaliação de construtores de NFP. *Dissertação de Mestrado*, Universidade Estadual de Campinas (Unicamp) - SP.

Toledo, F. M., Carravilla, M. A., Ribeiro, C., Oliveira, J. F. e Gomes, A. M. 2013. The dotted-board model: A new MIP model for nesting irregular shapes. *International Journal of Production Economics*, *145*(2), 478–487.

Toso, R. F. e Resende M. G. C. A C++ application programming interface for bised random-key genetic algorithms. *ATandT Labs Research Technical Report*.

Yang, X. S. e Deb, S. 2010. Cuckoo search via Lévy flights *World congress on nature and biologically inspired computing (NaBIC)*, 210–214.

APLLYING THE BIASED RANDOM-KEY GENETIC ALGORITHM IN A CUTTING PROBLEM WITH IRREGULAR SHAPED ITEMS

Leandro Resende Mundim, mundim@icmc.usp.br [1]
Marina Andretta, andretta@icmc.usp.br[1]
Thiago Alves de Queiroz, taq@ufg.br [2]

[2]Universidade de São Paulo - Instituto de Ciências Matemáticas e de Computação.
Av. Trabalhador São-Carlense, 400, São Carlos - SP.

[3]Universidade Federal de Goiás - Departamento de Matemática.
Av. Dr. Lamartine Pinto de Avelar, 1120, Catalão - GO.

Abstract. *A problem with few references in the literature of nesting problems is the 0-1 knapsack problem with irregular shaped items (2PMI). In the 2PMI, it is required that a subset of items has to be packed in a rectangular bin aiming to maximize the occupied area. In this paper we use a framework proposed in the literature, which implements a biased random-key genetic algorithm. The main contribution of this paper is the allocation technique that considers the bottom-left or top-left corner of the bin. In order to find feasible positions and avoid overlapping between items, we use a mesh over a grid of no-fit polygons. The computational experiments validate the proposed algorithm showing that it is competitive, since it improved results from the recent literature.*

Keywords: *Two-dimensional cutting problems; 0-1 Knapsack Problem; Irregular shaped items; Genetic algorithm; Biased-key; No-fit polygon.*