

Um Híbrido BRKGA-MP aplicado a um Problema da Mochila Não Linear

Wagner Pimentel

CEFET-RJ

Maracanã, Rio de Janeiro

wagner.pimentel@cefet-rj.br

Clovis Jose da Silva

CEFET-RJ

Adrianópolis, Nova Iguaçu

clovis.silva@cefet-rj.br

Matheus Pimentel Cardoso

CEFET-RJ

Maracanã, Rio de Janeiro

matheus.pimentel@aluno.cefet-rj.br

RESUMO

Neste artigo, propomos a aplicação de *Multi-Parent Biased Random-Key Genetic Algorithms* na solução de um interessante problema da mochila não linear e não convexo, onde a função objetivo e as restrições são funções separáveis. Um conjunto de instâncias da literatura foi selecionado para a apresentação dos resultados computacionais. Apresentamos uma estratégia eficiente de busca local em associação com a meta-heurística adotada. Novos limites inferiores foram obtidos para a maior parte das instâncias, demonstrando o potencial da combinação utilizada.

PALAVRAS CHAVE. Pesquisa Operacional, Meta-heurística, Problema da Mochila Não Linear.

ABSTRACT

In this article, we propose the application of *Multi-Parent Biased Random-Key Genetic Algorithms* to solve an interesting nonlinear and non-convex knapsack problem, where the objective function and constraints are separable. A set of instances from the literature was selected to present the computational results. We present an efficient local search strategy in association with the adopted metaheuristic. New lower bounds were obtained for most instances, demonstrating the potential of the proposed approach.

KEYWORDS. Operations Research, Metaheuristics, Nonlinear Knapsack Problem.

1. Introdução

O problema da mochila é um clássico em otimização combinatória, no qual um conjunto de itens deve ser selecionado de forma a maximizar o valor total associado, sem exceder a capacidade da mochila. Este problema tem sido amplamente estudado na literatura. Trabalhos recentes, como Cacchiani et al. [2022a] e Cacchiani et al. [2022b], apresentam uma visão geral dos avanços no problema da mochila e algumas de suas variações, como o problema da mochila quadrática. Isso indica que o problema da mochila é um tema de pesquisa muito ativo em otimização combinatória. O foco deste artigo é uma versão não linear do problema da mochila, que é relativamente pouco estudada em comparação com o problema clássico. O problema da mochila não linear (PMNL), também conhecido como problema não linear de alocação de recursos, é abordado em Katoh e Ibaraki [1998]. Esse trabalho considera uma função objetivo não linear sujeita a uma restrição, onde as variáveis de decisão podem ser discretas ou contínuas, limitadas ou ilimitadas. Variantes do PMNL estão relacionadas às propriedades das funções envolvidas, como convexidade ou não convexidade, separabilidade ou não separabilidade e, eventualmente, aos limites e tipos das variáveis de decisão. Neste trabalho, o PMNL apresenta funções não convexas, separáveis, e as variáveis de decisão são limitadas.

1.1. Trabalhos Relacionados ao PMNL

Em D'Ambrosio e Martello [2010], os autores desenvolveram um algoritmo heurístico e um procedimento de pós-otimização de busca local para resolver o PMNL aqui tratado. Na mesma linha de pesquisa, Mencarelli et al. [2016] propuseram algoritmos heurísticos com busca local para o problema da mochila não linear múltipla, com função não convexa separável e funções de peso. O problema da mochila não linear múltipla com funções separáveis e não convexas também foi abordado em D'Ambrosio et al. [2018], por meio da aplicação de técnicas construtivas heurísticas baseadas em relaxações substitutas e melhorias por busca local na solução do problema. Em Hoto et al. [2020], foi proposto um algoritmo de gradiente projetado e penalização para a solução do problema da mochila separável e convexo. Em Fampa et al. [2020], uma relaxação do problema da mochila quadrática convexa foi resolvida utilizando um algoritmo de plano de corte.

1.2. Proposta de Solução Relacionada ao PMNL

Algoritmos genéticos têm sido amplamente usados na solução de problemas de otimização combinatória e são teórica e empiricamente comprovados como fornecendo uma estratégia de busca robusta Goldberg [1989]. Neste artigo, propomos a utilização de um híbrido, o *Multi-Parent Biased Random-Key Genetic Algorithm* (BRKGA-MP) Andrade et al. [2021], como proposta de solução para os PMNL abordados. Analisaremos a qualidade da solução proposta comparando o BRKGA-MP com a melhor solução disponível em D'Ambrosio [2010]. O BRKGA-MP é uma extensão do BRKGA, *Biased Random-Key Genetic Algorithm*, proposto em Goncalves e Resende [2011], e adiciona a este um novo mecanismo de geração de descendentes, o cruzamento *Multi-Parent*. Trabalhos recentes utilizando BRKGA e BRKGA-MP em problemas de otimização combinatória têm sido apresentados na literatura. Chaves et al. [2016] aplicaram o BRKGA ao problema de alocação de recursos. O problema de empacotamento foi tratado em Zudio et al. [2018], aliado a uma busca por vizinhança. Pinto et al. [2018] utilizaram o BRKGA na solução de um problema clássico de grafos. Com relação ao BRKGA-MP, destacamos, entre os trabalhos recentes, a aplicação da meta-heurística na solução de um problema de alocação de recursos na área hospitalar Kummer et al. [2024].

Este artigo está organizado da seguinte forma: as formulações estudadas do PMNL são apresentadas na Seção 2. Na Seção 3, descrevemos os princípios gerais da meta-heurística BRKGA-MP, apresentamos o módulo decodificador desenvolvido e o procedimento de busca local. Na Seção 4, as instâncias, o ajuste de parâmetros e os resultados numéricos são apresentados. Por fim, na Seção 5, apresentamos uma análise e conclusão do trabalho.

2. Formulação dos Problemas

Seja $N = \{1, \dots, n\}$ o conjunto de itens do problema e $x = (x_1, \dots, x_n)$ um vetor, onde cada entrada $x_j \in \mathbb{S}$ é uma variável de decisão associada ao item $j \in N$, com $\mathbb{S} = \mathbb{R}$ ou $\mathbb{S} = \mathbb{Z}$. Para cada item $j \in N$, associamos as funções $f_j(x_j)$ e $g_j(x_j)$, respectivamente, na composição da função objetivo separável, $f(x)$, e na restrição, $g(x)$, do problema. Assim, o PMNL consiste de encontrar um valor para cada variável de decisão x_j , de modo que a função objetivo seja maximizada e a capacidade da mochila C , não seja excedida. Portanto, a formulação geral do PMNL será dada por:

$$\begin{aligned}
 (P) : \max \quad & f(x) \\
 \text{s.t.} \quad & g(x) \leq C \\
 & 0 \leq x_j \leq u_j, \quad j \in N \\
 & x_j \in \mathbb{S}, \quad j \in N, \\
 & u_j \in \mathbb{S}, \quad j \in N,
 \end{aligned}$$

Nesta formulação, a função $f(x)$ e restrição $g(x)$ são não convexas e separáveis, D'Ambrosio e Martello [2010]. Cabe mencionar que (P) é um problema de otimização NP-difícil, Martello e Toth [1990]. A partir do modelo (P), derivamos quatro PMNL. Nesses problemas, a função objetivo não convexa adotada será $f(x) = \sum_{j \in N} f_j(x_j)$, onde $f_j(x_j) = \frac{c_j}{1+b_j e^{-a_j(x_j+d_j)}}$, as restrições são do tipo $g(x) = \sum_{j \in N} g_j(x_j)$ e a capacidade da mochila será definida como $C = \frac{1}{2} g_j(u_j)$, tornando as instâncias do problema mais desafiadoras. Os coeficientes $a_j, b_j, c_j, d_j, w_j, p_j, q_j$ e u_j são valores associados a cada item $j \in N$ e estão disponíveis nas instâncias do PMNL. Portanto, os problemas adotados nesta pesquisa são definidos como a seguir:

$$\begin{aligned}
 (P1|P3) : \max \quad & \sum_{j \in N} \frac{c_j}{1+b_j e^{-a_j(x_j+d_j)}} \\
 \text{s.t.} \quad & \sum_{j \in N} w_j x_j \leq \frac{1}{2} \sum_{j \in N} w_j u_j \\
 & 0 \leq x_j \leq u_j, \quad j \in N \\
 & x_j \in \mathbb{S}, \quad j \in N. \\
 & u_j \in \mathbb{S}, \quad j \in N.
 \end{aligned}$$

$$\begin{aligned}
 (P2|P4) : \max \quad & \sum_{j \in N} \frac{c_j}{1 + b_j e^{-a_j(x_j + d_j)}} \\
 \text{s.t.} \quad & \sum_{j \in N} (\sqrt{p_j x_j + q_j} - \sqrt{q_j}) \leq \frac{1}{2} \sum_{j \in N} (\sqrt{p_j u_j + q_j} - \sqrt{q_j}) \\
 & 0 \leq x_j \leq u_j, \quad j \in N \\
 & x_j \in \mathbb{S}, \quad j \in N. \\
 & u_j \in \mathbb{S}, \quad j \in N.
 \end{aligned}$$

Observe que os problemas $(P1|P3)$ têm uma restrição linear, enquanto $(P2|P4)$ apresenta uma restrição não linear, côncava. Além disso, em $(P1)$ e $(P2)$, $\mathbb{S} = \mathbb{Z}$, ao passo que em $(P3)$ e $(P4)$, $\mathbb{S} = \mathbb{R}$.

3. Descrição da Meta-heurística

O BRKGA-MP, descrito em Andrade et al. [2021], é uma meta-heurística usada para fornecer boas soluções para problemas de otimização combinatória. Um componente importante do BRKGA-MP é o módulo chamado decodificador. Este módulo consiste em um algoritmo determinístico que é usado para mapear um vetor de chaves aleatórias em uma solução viável do PMNL, além de retornar o valor da função objetivo. A dinâmica evolutiva do BRKGA-MP utiliza a ideia de que indivíduos mais adaptáveis ao ambiente têm maiores probabilidades de sobrevivência do que os menos adaptáveis, produzindo um maior número de descendentes. A população inicial, denotada por P , é composta por $|P|$ indivíduos, no espaço $[0, 1]^n$, gerados de forma aleatória e são independentes do problema a ser tratado. Em cada passo do processo evolutivo uma nova geração é formada e a qualidade de cada indivíduo é determinada. A seguir a população é ordenada e dividida em dois subconjuntos: um conjunto elite, P_e , e outro conjunto não elite, $P \setminus P_e$. O conjunto elite é composto dos, $|P_e|$, melhores indivíduos da população, já o conjunto não elite, contém os demais indivíduos, $|P| - |P_e|$. Na prática o conjunto elite sempre é menor do que o seu complemento sendo integralmente copiado para a próxima geração. Assim, os melhores indivíduos são preservados para as gerações futuras. Diferentemente do algoritmo genético tradicional, o BRKGA-MP implementa a mutação por meio da introdução de um pequeno número de mutantes P_m , representado por, $|P_m|$ novos vetores de chaves aleatórias. Para completar a nova população é necessário criar um número de descendentes igual a $|P| - (|P_e| + |P_m|)$, novos indivíduos. Esses descendentes são gerados usando o cruzamento *Multi-Parent*, na qual vários indivíduos são selecionados aleatoriamente na população atual, parte do conjunto elite e outros do conjunto não elite. Eles são combinados na realização da operação de cruzamento com reposição em que todos os indivíduos gerados farão parte da próxima população. Assim, o BRKGA-MP simula um processo evolutivo permitindo que ao final da sua execução o melhor indivíduo encontrado represente uma solução viável de excelente qualidade para o problema.

3.1. Módulo Decodificador

O objetivo deste módulo decodificador é mapear um vetor de chaves aleatórias $y = (y_1, \dots, y_n) \in [0, 1]^n$, um indivíduo BRKGA-MP, em uma solução viável $x = (x_1, \dots, x_n) \in \mathbb{S}^n$

do PMNL e avaliar a qualidade do indivíduo, y , por retornar a sua aptidão. O módulo decodificador consiste dos seguintes passos: para cada item, j , um nível de importância é associado de acordo com sua relevância. A relevância é definida pelo vetor, r , onde cada item, j , está associado a uma razão $r_j = f_j(x_j)/g_j(x_j)$, isto é descrito entre as linhas 3 e 10. Na linha 11, a rotina OrdenaIndices() ordena as razões em r , de forma decrescente, e retorna um conjunto de índices, K . Esta mesma ideia é aplicada a relaxação do problema da mochila linear para obter a solução ótima. Nesta pesquisa esta estratégia tem-se mostrado ser muito eficiente em fornecer solução de boa qualidade para o PMNL. Entre as linhas 12 e 20, os itens são escolhidos para serem colocados na mochila de acordo com a ordem definida em K , até a capacidade C . Para cada item, j , não acrescentado a mochila, $x_j = 0$, nós associamos $y_j = 0$ à componente do indivíduo avaliado. Cabe observar que $x_j = 0$ implica em $f_j(x_j) \geq 0$, assim $f_j(x_j)$ continua contribuindo para a função objetivo, linha 19. Ao final, o decodificador retorna a qualidade da solução representada no indivíduo de entrada, y .

Algoritmo 1: Decodificador para os Problemas

Entrada: $y, a, b, c, d, w, p, q, u, C$

```

1   $fitness = 0$ ;
2   $cap = 0$ ;
3  para  $j=1$  até  $N$  faça
4     $x_j = \text{ComputaX}(y_j, u_j)$ ;
5    se  $(x_j > 0)$  então
6       $f_j = c_j / (1 + b_j e^{-a_j(x_j+d_j)})$ ;
7       $g_j = \text{ComputaRestrição}(x_j, w_j, p_j, q_j)$ ;
8       $r_j = f_j / g_j$ ;
9    senão
10      $r_j = 0$ ;
11   $K = \text{OrdenaIndices}(r)$ ;
12  para  $j=1$  até  $N$ ,  $j \in K$  faça
13     $x_j = \text{ComputaX}(y_j, u_j)$ ;
14     $g_j = \text{ComputaRestrição}(x_j, w_j, p_j, q_j)$ ;
15    se  $(g_j \leq C - cap)$  então
16       $fitness += c_j / (1 + b_j e^{-a_j(x_j+d_j)})$ ;
17       $cap += g_j$ ;
18    senão
19       $fitness += c_j / (1 + b_j e^{-a_j(0.0+d_j)})$ ;
20       $y_j = 0$ ;

```

Saída: $fitness$

Tabela 1: Descrição das rotinas do decodificador

Rotina	Descrição
ComputaX(y_j, u_j)	Retorna a j -ésima componente, x_j , da solução viável do PMNL associada a componente, y_n , do indivíduo, y . E mais, $x_j = \lfloor (u_j + 1)y_j \rfloor$, se $x_j \in \mathbb{Z}$ ou $x_j = u_j y_j$, se $x_j \in \mathbb{R}$;
ComputaRestrição(x_j, w_j, p_j, q_j)	Calcula e retorna o j -ésimo termo que compõem a restrição do problema. Retorna $w_j x_j$ nos problemas (P1 P3) e retorna $\sqrt{p_j x_j + q_j} - \sqrt{q_j}$, em (P2 P4);
OrdenaIndices(r)	Retorna o conjunto K onde cada componente, K_j , corresponde ao índice da j -th maior razão em r .

3.2. Procedimento de Busca Local

O mecanismo de busca local, BL, tem sido amplamente utilizado para intensificar a procura por soluções de qualidade de problemas de otimização combinatória.

Algoritmo 2: Procedimento de Busca Local

Entrada: y

```

1  para  $i = 1$  até  $N - 1$  faça
2    para  $j = i + 1$  até  $N$  faça
3       $x_i = \text{ComputaX}(y_i, u_i)$ ;
4       $x_j = \text{ComputaX}(y_j, u_j)$ ;
5       $f_i = c_i / (1 + b_i e^{-a_i(x_i + d_i)})$ ;
6       $f_j = c_j / (1 + b_j e^{-a_j(x_j + d_j)})$ ;
7       $g_i = \text{ComputaRestrição}(x_i, w_i, p_i, q_i)$ ;
8       $g_j = \text{ComputaRestrição}(x_j, w_j, p_j, q_j)$ ;
9       $f_i^j = c_j / (1 + b_j e^{-a_j(x_i + d_j)})$ ;
10      $f_j^i = c_i / (1 + b_i e^{-a_i(x_j + d_i)})$ ;
11      $g_i^j = \text{ComputaRestrição}(x_i, w_j, p_j, q_j)$ ;
12      $g_j^i = \text{ComputaRestrição}(x_j, w_i, p_i, q_i)$ ;
13     se  $((f_i^j + f_j^i) - (f_i + f_j) > 0)$  então
14       se  $((g_i^j + g_j^i) - (g_i + g_j) \leq (C - cap))$  então
15         troca  $y_i$  com  $y_j$ ;
16          $cap += (g_i^j + g_j^i) - (g_i + g_j)$ ;

```

Saída: y

Nesta pesquisa, o procedimento de busca local é aplicado apenas ao melhor indivíduo, y , da população atual, a cada geração do BRKGA-MP. O custo computacional desta busca é $\mathcal{O}(n^2)$. Após o processo de aplicação da busca local o novo indivíduo obtido é reinserido na população atual, potencializando o processo evolutivo. Nesta busca, entre as linhas 3-8, as componentes, x_i e

x_j , as funções, f_i e f_j e as restrições, g_i e g_j , são calculadas. Nas linhas 9-12 avaliamos as funções $f_j(x_i)$ e $f_i(x_j)$, junto com as restrições $g_j(x_i)$ e $g_i(x_j)$, trocando x_i por x_j e vice-versa. Esses resultados contribuem para as duas condições investigadas, linhas 13 e 14. Se essas condições são satisfeitas, o valor da função objetivo pode ser aumentado sem que a capacidade da mochila seja violada. Na linha 15 efetuamos as trocas das componentes no indivíduo de entrada e na linha 16 a capacidade relativa da mochila é atualizada. No final deste procedimento, se houver alguma troca de componentes, teremos um indivíduo com o seu *fitness* melhorado.

4. Resultados Computacionais

Os módulos foram implementados na linguagem C++ e compilados com o compilador g++. Todos os testes foram feitos em uma máquina com processador Intel(R) Core(TM) i7-4770 CPU @ 3.40GHz e 8GB RAM, com o sistema Cygwin_64 3.3.5 (2022).

4.1. Descrição das Instâncias

Selecionamos um conjunto de 160 instâncias para cada tipo de problema descrito na seção 2, totalizando 640 instâncias de teste. Todas as instâncias e soluções, do tipo *Non Linear Knapsack Problem*, podem ser acessadas em <https://site.unibo.it/operations-research/en/research/library-of-codes-and-instances-1>. Essas instâncias estão divididas em dois conjuntos: (i) instâncias correspondentes aos problemas ($P1|P3$) com restrições lineares, *Nonconvex nonconcave profits*, *Linear weights*, e (ii) instâncias correspondentes aos problemas ($P2|P4$) com restrições não lineares, *Nonconvex nonconcave profits*, *Nonlinear weights*. As instâncias caracterizam-se por conter os parâmetros gerados aleatoriamente, onde $a_j \in [0.1, 0.2]$, $b_j \in [0, 100]$, $c_j \in [0, 100]$, $d_j \in [-100, 0]$, $w_j \in [1, 100]$, $p_j, q_j \in [1, 20]$ e $u_j = 100$, para cada $j \in N$, com $n \in \{10, 20, 50, 100, 200, 500, 1000, 2500\}$.

4.2. Ajuste de Parâmetros do BRKGA-MP

Os parâmetros do BRKGA-MP, sem a estratégia de busca local, foram refinados utilizando o *IRACE PACKAGE*, Lopez-Ibanez et al. [2016]. Os parâmetros ajustados nesta pesquisa foram: o tamanho da população, $|P|$, o tamanho do conjunto elite, $|P_e|$, o tamanho do conjunto mutante, $|P_m|$, o número de indivíduos do conjunto elite, $|t_e|$, e o número total de indivíduos, $|t_p|$, para o cruzamento *Multi-Parent*. Ajustamos, ainda, o tipo de função utilizada no cruzamento *Multi-Parent*, *bias*, e o intervalo de reset da população do BRKGA-MP, i_{reset} . Utilizamos um limite de tempo de 120 segundos para cada simulação nesta etapa de ajuste de parâmetros.

Nós geramos um conjunto de instâncias especialmente para esta etapa, <https://drive.google.com/drive/folders/1NV73tCmNCtVZhX2ikHAW6l5ZnaMVCakX?usp=sharing>, disponíveis na pasta *Instancias_Tunning*. Para cada tipo de problema foram geradas 4 instâncias com $n \in \{10, 20, 50, 100, 200, 500, 1000, 2500\}$, totalizando 128 instâncias para esta fase de ajuste de parâmetros, o que equivale uma amostra de 20% considerando todas as instâncias de teste. O intervalo dos valores para cada um dos parâmetros refinados, bem como a melhor configuração dos mesmos estão apresentados na Tabela 2.

Tabela 2: Parâmetros e intervalos considerados e a melhor configuração encontrada

Parâmetros	Intervalos	(P_1)	(P_2)	(P_3)	(P_4)
$ P $	$\{50, 100, \dots, 400, 500, \dots, 1000, 2500\}$	900	2500	900	1000
$ P_e $	$[0.05, 0.30]$	0.18	0.21	0.20	0.10
$ P_m $	$[0.05, 0.30]$	0.14	0.09	0.12	0.05
$ t_e $	$[1, 9]$	5	5	5	4
$ t_p $	$[2, 10]$	7	8	6	10
$bias$	$\{0, 1, 2, 3, 4, 5\}$	3	5	4	3
i_reset	$\{100, 200, \dots, 1000, 1500, 2000, 3000\}$	1000	1000	3000	1000

4.3. Resultados Numéricos

Em nossos experimentos, para cada instância do PMNL, o BRKGA-MP foi executado 10 vezes, por 120 segundos. Ao final de cada execução selecionamos a melhor solução do PMNL, denotada por x_B , e a melhor solução geral dentre as 10 execuções, x_B^* . Denotaremos, ainda, por x^* , a melhor solução para cada instância disponível em <https://site.unibo.it/operations-research/en/research/library-of-codes-and-instances-1>.

O custo dessas soluções denotaremos por s_B , s_B^* e s^* , respectivamente. Esta notação será utilizada na base de dados disponível para consulta em <https://drive.google.com/drive/folders/1NV73tCmNctVZh2ikHAW6l5ZnaMVCakX?usp=sharing>. Apresentamos, nesta base de dados, as tabelas contendo os resultados computacionais, para cada tipo de PMNL. Na primeira coluna de cada tabela apresentamos o nome da instância. Na segunda coluna apresentamos o custo associado a melhor solução encontrada, s_B^* , e o tempo despendido até encontrá-las, $t_{x_B^*}$, em segundos. Apresentamos, ainda, o custo da melhor solução para cada instâncias disponível na literatura, s^* . Na última coluna da tabela, o ganho percentual relativo, $gap_{s_B^*} = \frac{100(s_B^* - s^*)}{s^*} \%$, é apresentado, comparando s_B^* e s^* . Novos limitantes inferiores para as instâncias do PMNL são apresentados quando $gap_{s_B^*} \geq 0$. Apresentamos, também, nesta base de dados, os resultados das execuções do BRKGA-MP, para cada tipo de PMNL e para cada instância de teste.

Nas tabelas 3 e 4, a seguir, nós apresentamos o percentual das instâncias que foram melhoradas, ou seja, o percentual de novos limites inferiores apresentados para cada grupo de instâncias, com $s_B^* > s^*$, obtidos com o BRKGA-MP, sem busca local e com busca local, respectivamente. Cada grupo é constituído por 20 instâncias.

Tabela 3: Percentual de $gap_{s_B^*} \geq 0$, sem Busca Local

Grupo de instâncias(n)	(P_1)	(P_2)	(P_3)	(P_4)
10	95.0%	90.0%	75.0%	85.0%
20	100.0%	95.0%	75.0%	80.0%
50	70.0%	90.0%	10.0%	70.0%
100	45.0%	10.0%	0.0%	0.0%
200	40.0%	0.0%	0.0%	0.0%
500	0.0%	0.0%	0.0%	0.0%
1000	0.0%	0.0%	0.0%	0.0%
2500	0.0%	0.0%	0.0%	0.0%
média	43.75%	35.63%	20.0%	29.38%

Tabela 4: Percentual de $gap_{s_B}^* \geq 0$, com BL

Grupo de instâncias(n)	(P1)	(P2)	(P3)	(P4)
10	95.0%	95.0%	80.0%	90.0%
20	95.0%	95.0%	85.0%	90.0%
50	80.0%	95.0%	45.0%	90.0%
100	90.0%	50.0%	30.0%	30.0%
200	100.0%	15.0%	85.0%	0.0%
500	95.0%	0.0%	85.0%	0.0%
1000	90.0%	0.0%	30.0%	0.0%
2500	0.0%	0.0%	0.0%	0.0%
média	80.60%	42.10%	55.00%	37.50%

5. Conclusão

Neste trabalho, nós consideramos a meta-heurística BRKGA-MP como proposta de solução dos problemas propostos e apresentamos uma eficiente estratégia de busca local. Os resultados computacionais foram bastante satisfatórios quando comparamos com os resultados disponíveis na literatura, neles apresentamos novos limites inferiores para grande parte das instâncias. Em relação a busca local, Tabela 4, os resultados apresentam um ganho substancial ao utilizá-la, eles apresentam um ganho médio de 76.00% em relação a sua não utilização, Tabela 3, reforçando o potencial da combinação da meta-heurística com a busca local. Portanto, com relação aos resultados alcançados nesta combinação, os melhores percentuais de novos limites inferiores foram alcançados para os grupos de instâncias dos problemas com restrição linear, $(P1|P3)$, apresentando percentual médio de 80.60%|55.00% contra 42.10%|37.50% de percentual de novos limitantes para os grupos de instâncias, $(P2|P4)$, com restrição não linear. Observamos nesta pesquisa que as instâncias de maior porte necessitariam de um tempo maior de execução para alcançar melhores resultados, visto que os melhores indivíduos BRKGA-MP são encontrados ao final do período de execução. Em relação aos resultados alcançados para os grupos de instâncias, $(P2|P4)$, pensamos que um estudo mais profundo da relação $r_j = f_j(x_j)/g_j(x_j)$ pode ser feito no sentido de melhorá-los. Entendemos como principal resultado científico deste trabalho os novos limitantes disponibilizados que podem servir de referência para futuras pesquisas.

Referências

- Andrade, C. E., Toso, R. F., Gonçalves, J. F., e Resende, M. G. C. (2021). The multi-parent bi-ased random-key genetic algorithm with implicit path-relinking and its real-world applications. *European Journal of Operational Research*, 289(1):17–30.
- Cacchiani, V., Iori, M., Locatell, A., e Martello, S. (2022a). Knapsack problems - an overview of recent advances, part i: Single knapsack problems. *Computers & Operations Research*.
- Cacchiani, V., Iori, M., Locatell, A., e Martello, S. (2022b). Knapsack problems - an overview of recent advances. part ii: Multiple, multidimensional, and quadratic knapsack problems. *Computers & Operations Research*.
- Chaves, A., Lorena, L., Senne, E., e Resende, M. (2016). Hybrid method with CS and BRKGA applied to the minimization of tool switches problem. *Computers & Operations Research*, 67: 174–183.

- D'Ambrosio, C. (2010). Web page. <https://site.unibo.it/operations-research/en/research/library-of-codes-and-instances-1>.
- D'Ambrosio, C. e Martello, S. (2010). Heuristic algorithms for the general nonlinear separable knapsack problem. *Computers & Operations Research*, 38:505–513.
- D'Ambrosio, C., Martello, S., e Mencarelli, L. (2018). Relaxations and heuristics for the multiple non-linear separable knapsack problem. *Computers & Operations Research*, 93:79–89.
- Fampa, M., Lubke, D., Wang, F., e Wolkowicz, H. (2020). Parametric convex quadratic relaxation of the quadratic knapsack problem. *European Journal of Operational Research*, 281(1):36–49.
- Goldberg, D. E. (1989). *Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley.
- Goncalves, J. F. e Resende, M. G. C. (2011). Biased random key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, p. 487–525.
- Hoto, R., Matioli, L., e Santos, P. (2020). A penalty algorithm for solving convex separable knapsack problems. *Applied Mathematics and Computation*, 387.
- Katoh, N. e Ibaraki, T. (1998). *Resource Allocation Problems*. Springer, Boston, MA.
- Kummer, A. F., de Araújo, O. C., Buriol, L. S., e Resende, M. G. (2024). A biased random key genetic algorithm for the home health care problem. *Electronic Notes in Discrete Mathematics*, 31:1859–1889.
- Lopez-Ibanez, M., Dubois-Lacoste, J., Caceres, L. P., Birattari, M., e Stützle, T. (2016). The Irace Package: Iterated Racing for Automatic Algorithm Configuration. *Operations Research Perspectives*, 3:43–58.
- Martello, S. e Toth, P. (1990). *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, New York.
- Mencarelli, L., D'Ambrosio, C., Di Zio, A., e Martello, S. (2016). Heuristics for the general multiple non-linear knapsack problem. *Electronic Notes in Discrete Mathematics*, 55:69–72.
- Pinto, B. Q. B. Q., Ribeiro, C., Rosseti, I., e Plastino, A. (2018). A biased random key genetic algorithm for the maximum quasi-clique problem. *European Journal of Operational Research*, p. 849–865.
- Zudio, A., da Silva Costa, D. H., Masquiao, B. P., Igor, M. C., e 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.