Um algoritmo híbrido entre Evolução Diferencial e Nelder-Mead inteiro para problemas de otimização

Felipe Luchi
Universidade Federal do Espírito Santo - UFES
Centro Tecnológico
Av. Fernando Ferrari, 514, Goiabeiras
CEP: 29075-910, Vitória - ES
luchifelipe@gmail.com

Renato A. Krohling
Universidade Federal do Espírito Santo - UFES
Centro Tecnológico
Av. Fernando Ferrari, 514, Goiabeiras
CEP: 29075-910, Vitória - ES
krohling.renato@gmail.com

RESUMO

Este trabalho propõe um algoritmo híbrido entre Evolução Diferencial e Nelder-Mead para resolver problemas de otimização inteira não linear. Evolução Diferencial é um algoritmo promissor em problemas de otimização não lineares com variáveis contínuas, utilizando topologia em anel para manter a diversidade na busca global. Nelder-Mead é um algoritmo utilizado para tratar problemas de otimização contínua. Ele é usado para intensificar a busca local até convergir para um ótimo. Neste trabalho, foi usada a versão discreta do Nelder-Mead para tratar problemas de otimização inteira. O algoritmo híbrido utiliza Evolução Diferencial para identificar regiões promissoras no espaço de busca para guiar a busca executada pelo Nelder-Mead. Os resultados apresentados são promissores e mostram a viabilidade da abordagem.

PALAVRAS CHAVE. Programação inteira não linear, Evolução Diferencial, Nelder-Mead.

Área Principal: Metaheuristica

ABSTRACT

This work proposes a hybrid algorithm based on Differential Evolution and Nelder-Mead to solve nonlinear integer optimization. Differential Evolution is a promising algorithm used in non-linear problems with continuous variables, using ring topology to maintain the diversity in global search. Nelder-Mead is a technique used in non-linear continuous optimization problems. It is used to exploit the search space until the convergence to an optima is achieved. In this work, it was used the discrete version of Nelder-Mead to deal with integer problems. The hybrid algorithm uses Differential Evolution to seek promising regions of search space to guide the Nelder-Mead. The results are promissing and show the viability of the approach.

KEYWORDS. Nonlinear integer programming, Differential Evolution, Nelder-Mead.

Main Area: Metaheuristics

1. Introdução

Muitos problemas de otimização do mundo real são não lineares. Alguns desses problemas não permitem valores fracionários em suas soluções, por serem impraticáveis ou não possuir o devido significado físico. Estes problemas são tratados como problemas de otimização inteira não linear e são encontrados em vários contextos como controle de estoque, projeto de rede de computadores, entre outros, de acordo com Arora, Huang e Hsieh (1994).

Um problema de otimização inteira não linear é definido de acordo com

$$\begin{aligned} \text{Min.} \quad & f(x) \\ \text{Sujeito a} \quad & a \leq x \leq b \\ & x \in \mathbb{Z} \end{aligned} \tag{1}$$

onde $f(x)$ é a função objetiva a ser minimizada, x representa o vetor de variáveis discretas do problema, a e b são os limites inferiores e superiores, respectivamente, e os elementos de x devem pertencer ao conjunto dos números inteiros $\mathbb{Z}$.

Abordagens clássicas são encontradas na literatura para resolver problemas de otimização inteira linear. Dentre as citadas por Arora, Huang e Hsieh (1994), encontra-se o *Branch and Bound*, que foi proposto por Land e Doig (1960) e o Corte de Gomory (1960), para problemas lineares, porém, muitos problemas de otimização do mundo real são não lineares. Sendo assim, Gupta e Ravindran (1983) extenderam o *Branch and Bound* utilizando um método baseado em gradiente reduzido generalizado para resolver problemas mistos não lineares. Este método garante encontrar a solução ótima somente em problemas lineares e, além disso, o tempo computacional pode aumentar exponencialmente para problemas de grandes dimensões.

Nelder & Mead (1965), abreviado por NM, propuseram um método para minimização de funções de D variáveis contínuas, que compara $D+1$ pontos através dos valores da função objetiva e substitui aquele, que possui o maior valor da função objetiva, por outro vértice. Esta abordagem se adapta à superfície, diminuindo o espaço de busca até convergir para o ótimo. Segundo os autores, esta abordagem supera aquela apresentada por Powell (1964), entretanto, depende da escolha de um ponto inicial, que, se escolhido inapropriadamente, conduz a busca à mínimos locais.

Um algoritmo biologicamente inspirado, foi proposto por Storn e Price (1995) para a minimização de funções não lineares e não diferenciáveis, com variáveis contínuas, denominado Evolução Diferencial (do inglês *Differential Evolution*, abreviado por DE). Este algoritmo requer poucos parâmetros para ajustar, é robusto e na versão original é apropriado para a resolução de problemas de otimização contínua. O algoritmo Evolução Diferencial possui a vantagem de não necessitar de uma solução inicial, porém pode ficar preso a mínimos locais.

Li, Wang e Sun (2007) utilizam o método de Lagrange para encontrar soluções exatas para problemas de otimização inteira não lineares. O algoritmo encontra a solução utilizando uma função de perturbação revisada através de sucessivas reformulações. Além disso, o algoritmo utiliza-se de outras técnicas para garantir a convergência do método de Lagrange. Segundo os resultados dos autores, trata-se de um algoritmo promissor para problemas de otimização inteira não linear de tamanho médio. Entretanto, depende de um solução inicial, que se escolhida inapropriadamente, conduz a mínimos locais.

Vários trabalhos utilizam estratégias híbridas em busca de melhores soluções. Nasab (2014), propôs um algoritmo híbrido usando *fuzzy* e Algoritmos Genéticos para resolver o problema de alocação de máquinas. Burke, Li e Qu (2014) propuseram um híbrido para o problema de escalonamento hospitalar, entre Programação Inteira, que gera soluções de subproblemas, e Busca em Vizinhança Variável (do inglês *Variable neighbourhood search*, abreviado por VNS). García, García-Rodenas e Gómez (2014) utilizaram uma estratégia híbrida para clusterização de dados com restrições no domínio do tempo e reconhecimento de padrões, utilizando algoritmos de otimização baseados em população e NM.

Wu et al. (2014) propuseram uma abordagem híbrida chamada Enxame de Partículas guiada por solução superior (do inglês *Superior Solution Guided Particle Swarm Optimization*, abreviado por SSG-PSO), combinado com duas técnicas baseadas em gradiente denominadas *Broyden Fletcher Goldfarb Shanno* (BFGS) e *Davidon Fletcher Powell* (DFP). Além disso, são utilizadas outras duas técnicas baseadas em derivadas que são Busca por Padrões (do inglês *Pattern Search*) e NM. Esta abordagem foi aplicada para resolver problemas de otimização sem restrições. Para problemas unimodais, SSG-PSO combinada com as técnicas baseadas em gradiente obteve bons resultados. Para problemas não diferenciáveis e descontínuos, SSG-PSO combinado NM e Busca por Padrões obteve melhores resultados, comparado ao SSG-PSO híbrido com DFP ou BFGS. Para problemas multimodais, o algoritmo SSG-PSO é eficaz devido a sua capacidade de exploração.

Schneider e Krohling (2014) propuseram um algoritmo híbrido utilizando DE, DE$_{best}$ e DE com topologias global-local, com Busca Tabu e a Técnica para avaliar o desempenho de alternativas através de similaridade com a solução ideal (TOPSIS) para dois tipos de problemas. Para problemas de otimização inteira com restrições, o desempenho dos algoritmos são bastantes similares, uma vez que eles convergem facilmente para um único ótimo. Contudo, para problemas de otimização inteira multi-objetivo com restrições, DE com topologias global-local obteve melhores resultados devido a sua capacidade de manter a diversidade da população.

Este trabalho apresenta uma abordagem híbrida, entre o DE, que atua identificando regiões promissoras para guiar a busca efetuada pelo NM para problemas de otimização inteira não linear. O restante deste artigo esta organizado da seguinte forma: a Sessão 2 apresenta a descrição dos algoritmos DE e MN. A Sessão 3 apresenta o esquema de hibridização entre o DE e NM. Os resultados de simulação são apresentados na Sessão 4 e as conclusões e direções para futuros trabalhos são dados na Sessão 5

2. Descrição dos algoritmos

2.1. Evolução Diferencial

Evolução Diferencial é uma algoritmo promissor para otimização de problemas multimodais com variáveis contínuas, desenvolvida por Storn e Price (1995), simples de implementar e possui poucos parâmetros para calibrar. DE inicia com uma população de NP indivíduos de D variáveis que são gerados aleatoriamente utilizando uma distribuição de probabilidade uniforme, que representam soluções candidatas do problema de otimização. A população do DE pode ser representada de acordo com

$$X_i^G = \left[x_{1,i}^G, x_{2,i}^G, \ldots, x_{D,i}^G\right] \tag{2}$$

onde $G = 1, 2, \ldots, G_{max}$ indicam as gerações do algoritmo e $i = 1, 2, \ldots, NP$ indicam os indivíduos da população.

Cada variável do problema possui limites inferiores e superiores. A população em $G = 0$ é inicializada aleatoriamente e deve cobrir todo o espaço de busca, sendo que $X_{min} = \{x_{1,min}, x_{2,min}, \ldots, x_{D,min}\}$ e $X_{max} = \{x_{1,max}, x_{2,max}, \ldots, x_{D,max}\}$ representam os limites inferiores e superiores respectivamente. A inicialização da população é definida de acordo com

$$x_{j,i} = x_{j,min} + rand_{i,j}[0,1] \cdot (x_{j,max} - x_{j,min}) \tag{3}$$

onde $rand[0,1]$ é um número aleatório gerado a partir de uma distribuição uniforme $U(0,1)$.

A mutação representa uma mudança, ou perturbação, entre elementos aleatórios. No DE, o vetor de variáveis atual é chamado *target*, o vetor obtido através da mutação é chamado *donor* e, por último, o vetor obtido através da combinação entre o *target* e *donor* é chamado *trial*. A mutação é definida de acordo com

$$V_i = X_{r_1} + F(X_{r_2} - X_{r_3}) \tag{4}$$

onde r_1, r_2 e r_3 são indivíduos escolhidos aleatoriamente na população de forma que $r_1 \neq r_2 \neq r_3$, F é chamado fator de mutação assumindo valores entre 0.4 e 1 que controla a ampliação da diferença entre X_{r_2} e X_{r_3}.

Para aumentar a diversidade entre os indivíduos, o *crossover* é feito após a mutação, através da troca de variáveis entre o *target* e *donor*. A operação de *crossover* é definida de acordo com

$$U_{i,j} = \begin{cases} V_{i,j}, & \text{se } U(0,1) \leq C_r, \text{ ou } j = j_{rand} \\ X_{i,j}, & \text{caso contrário} \end{cases} \quad (5)$$

onde C_r é a taxa de *crossover*, j_{rand} é um número aleatório entre $[1, D]$ para que pelo menos uma variável de V_i faça parte de U_i.

Após a mutação e *crossover*, a seleção é feita para garantir que somente os melhores indivíduos sejam transferidos para a próxima geração e é definida de acordo com

$$X_i = \begin{cases} U_i, & \text{se } U_i \text{ é melhor que } X_i \\ X_i, & \text{caso contrário} \end{cases} \quad (6)$$

onde U_i e X_i são os vetores *trial* e *target* respectivamente.

2.1.1. Evolução Diferencial com topologia em anel

Esta variação utiliza uma topologia em anel proposta por Li (2010), que equilibra a capacidade de exploração do DE. A mutação é feita utilizando os membros da vizinhança (vetor local). Este vetor utiliza uma vizinhança em anel de tamanho k, ou seja, o indivíduo i possui a vizinhança $[i-k, i+k]$ e é calculado de acordo com

$$\bar{L}_i = \bar{X}_i + \alpha(\bar{X}_{best_i} - \bar{X}_i) + \beta(\bar{X}_{r1} - \bar{X}_{r2}) \quad (7)$$

onde $\bar{X}_{best_i}$ é o melhor indivíduo da vizinhança e i, r_1 e r_2 são indivíduos escolhidos aleatoriamente na vizinhança de forma que $i \neq r_1 \neq r_2$.

O algoritmo DE com topologia em anel é apresentado no algoritmo 1.

Algoritmo 1: Evolução diferencial com topologia em anel

```
início
    inicialização de acordo com a equação (3)
    repita
        para i = 1 to NP faça
            selecionar r1, r2 tal que i ≠ r1 ≠ r2
            mutação de acordo com a equação (7)
            crossover de acordo com a equação (5)
            seleção de acordo com a equação (6)
        fim
    até critério de parada seja satisfeito;
fim
```

2.2. Nelder Mead para variáveis inteiras

Nelder-Mead é um algoritmo usado para minimização de funções não lineares. Uma vez que este trabalho trata problemas de otimização inteira, é necessário estender as operações do NM para variáveis inteiras.

Brea (2013) propôs uma nova abordagem do NM, denominada *Integer Mixed Simplex Algorithm*, para problemas de otimização misto não linear, composto por funções no espaço D-dimensional com variáveis reais e inteiras. As operações propostas por Nelder & Mead (1965) são

efetuadas sobre pontos reais, sendo assim, um novo grupo de operações é aplicado para pontos discretos. Neste trabalho, foi utilizado o NM somente com a estrutura de pontos discretos.

Dado um problema de minimização, NM utiliza $D+1$ pontos no espaço de busca. Seja y_i o valor da função objetivo no ponto P_i, y_h e y_l é o valor do maior e menor valor da função objetiva, respectivamente. Inicialmente deve-se calcular o centróide dos D pontos, de acordo com

$$\bar{P} = \frac{1}{D}\sum_{i=1}^{D} P_i \qquad (i \neq h) \tag{8}$$

A reflexão de um ponto é definida de acordo com

$$P^* = P_h + \alpha \cdot \mu \cdot sgnd(\bar{P} - P_h) \tag{9}$$

onde α é o coeficiente de reflexão inteiro, $\mu = \lceil |\bar{y} - y_h| \rceil$ e $sgnd(k)$ é definido de acordo com

$$sgnd(k) = \begin{cases} 1, & \text{se } k > 0 \\ 0, & \text{se } k = 0 \\ -1, & \text{se } k < 0 \end{cases} \tag{10}$$

A expansão de um ponto é definida de acordo com

$$P^{**} = P_h + \beta \cdot \mu \cdot sgnd(\bar{P} - P_h) \tag{11}$$

onde β é o coeficiente de expansão inteiro.

A contração de um ponto é definida de acordo com

$$P^{**} = P_h + \gamma \cdot \mu \cdot sgnd(\bar{P} - P_h) \tag{12}$$

onde γ é o coeficiente de contração inteiro.

A operação de encolhimento é similar aquela executada pelo NM para variáveis contínuas, porém, esta operação pode assumir valores contínuos como resultado. Portanto, o encolhimento é definido de acordo com

$$P_i = \left[\frac{(P_i + P_l)}{2}\right] \tag{13}$$

onde $[\cdot]$ representa o arredondamento para o inteiro mais próximo.

O algoritmo MN para variáveis inteiras é apresentado no algoritmo 2.

Algoritmo 2: Nelder-Mead para variáveis inteiras (Brea, 2013)

```
início
    repita
        calcular o centróide dos D_i(i ≠ h) de acordo com a equação (8)
        calcular P* de acordo com a equação (9)
        se y* < y_l então
            calcular P** de acordo com a equação (11)
            se y** < y_l então
                substituir P_h por P**
            senão
                substituir P_h por P*
            fim
        senão
            se y* > y_i, i ≠ h então
                se y* > y_h então
                    calcular P** de acordo com a equação (12)
                else
                    substituir P_h por P*
                    calcular P** de acordo com a equação (12)
                end
                se y** > y_h então
                    encolhimento de acordo com a equação (13)
                senão
                    substituir P_h por P**
                fim
            senão
                substituir P_h por P*
            fim
        fim
    até critério de parada seja satisfeito;
fim
```

3. Hibridização entre o Evolução Diferencial e Nelder-Mead

A hibridização entre DE e NM é similar a utilizada por Ponsich e Coello (2013), que utilizaram DE e Busca Tabu para resolver problemas de escalonamento *job-shop*, onde a busca local é executada após os operadores do DE a cada 10 iterações. Os pontos utilizados pelo NM são gerados utilizando uma função de distribuição normal em torno do melhor indivíduo encontrado pelo DE na iteração G. Os indivíduos no DE são contínuos, sendo assim, eles são discretizados utilizando arredondamento para o número inteiro mais próximo.

3.1. Parâmetros do algoritmo

Os parâmetros utilizados no DE são listados a seguir. O tamanho da população e a taxa de *crossover*, são sugeridos por Storn e Price (1995) e são iguais a $10 \cdot D$ e 0.8, respectivamente. Os parâmetros α e β, são sugeridos por Das e Suganthan (2011) e ambos são iguais a 0.8.

Os parâmetros utilizados no NM são listados a seguir. O coeficiente de reflexão (α), coeficiente de expansão (β) e o coeficiente de contração (γ) são sugeridos por Nelder & Mead (1965) e são iguais a 1, 2 e 0.5, respectivamente. O número de iterações é o mesmo utilizado por Schneider e Krohling (2014) no mecanismo de busca local, e é igual a 1000.

O pseudocódigo do algoritmo DE+NM é apresentado no algoritmo 3.

Algoritmo 3: Algoritmo híbrido DE+NM

início
  inicialização
  G=1
  repita
    para i de 1 até NP **faça**
      selecionar r_1, r_2 de forma que $i \neq r_1 \neq r_2$
      mutação de acordo com a equação (7)
      crossover de acordo com a equação (5)
      seleção de acordo com a equação (6)
    fim
    se $G \bmod 10 = 0$ **então**
      atribui o melhor indivíduo da geração G a P_0
      gerar D candidatos através da distribuição de probabilidade normal em torno de P_0, para inicializar o NM
      executar NM
    fim
    G = G + 1
  até critério de parada seja satisfeito;
fim

4. Resultados computacionais

4.1. Problemas de teste

Neste trabalho é utilizado DE com topologia em anel. Para testar o desempenho do algoritmo foram utilizados três *benchmarks* de otimização inteira não linear, os mesmos utilizados por Tian, Ma e Zhang (1998).

Benchmark 1: *Shekel*.

$$\text{Min.} \quad f(x) = \sum_{j=1}^{m} \frac{(-1)}{|(x - A_j)^T (x - A_j) + c_j|} \tag{14}$$

onde A_j e c_j são dados pela tabela 1. O ótimo global encontra-se em $x_{opt} = (4, 4, 4, 4)$. As variações utilizadas são $m = 5$ (SQRN5), $m = 7$ (SQRN7) e $m = 10$ (SQRN10) e os mínimos globais são listados na tabela 2. Os limites inferiores e superiores utilizados são $0 \leq x_i \leq 10, (i = 1, 2, 3, 4)$.

A_j				c_j
4.0	4.0	4.0	4.0	0.1
1.0	1.0	1.0	0.1	0.2
8.0	8.0	8.0	8.0	0.2
6.0	6.0	6.0	6.0	0.4
3.0	7.0	3.0	7.0	0.6
2.0	9.0	2.0	9.0	0.6
5.0	5.0	3.0	3.0	0.3
8.0	1.0	8.0	1.0	0.7
6.0	2.0	6.0	2.0	0.5
7.0	3.6	7.0	3.6	0.5

Tabela 1: Matrizes A_j e c_j utilizadas em Shekel.

Benchmark 2: A função utilizada é definida pela equação (15), os limites inferiores e superiores, pontos ótimos e mínimos globais são apresentados na tabela 3.

Instância	Função	$f(x_{opt})$
p1-I	SQRN5	-10.1527
p1-II	SQRN7	-10.4023
p1-III	SQRN10	-10.5358

Tabela 2: Ótimos global da função *Shekel*.

$$\begin{aligned}\text{Min.} \quad & (x_1-3)^2\cdot\cos(\pi\cdot x_1)+(x_2-6)\sin(\tfrac{\pi}{4})+\\ & (x_3-2.5)^2\cdot(x_2+2)^{-1}+(x_3+2)^3\cdot e^{-x_4}\end{aligned} \tag{15}$$

Instância	Limites	x_{opt}	$f(x_{opt})$
p2-I	$[0,60]$	$(59,54,2,34)$	-3183.995
p2-II	$[0,80]$	$(79,78,2,33)$	-5847.9969
p2-III	$[0,100]$	$(99,94,2,32)$	-9303.9974

Tabela 3: Limites inferiores e superiores, pontos ótimos e ótimos globais do *benchmark* 2.

Benchmark 3: A função utilizada é definida pela equação (16), os limites inferiores e superiores, pontos ótimos e ótimos globais são apresentados na tabela 4.

$$\begin{aligned}\text{Min.} \quad & (x_1-2.5)^2\cdot(x_2+12.6)^2\cdot(x_3+25.4)+(x_3-4.5)^2\cdot(x_4+18.4)^{-1}\\ & \cdot e^{x_2-6.5}+x_4^3\cdot(x_5+10.8)^2\cdot\sin\left(\frac{\pi}{10}\cdot(x_6+1)\cdot x_5\right)\end{aligned} \tag{16}$$

Instância	Limites	x_{opt}	$f(x_{opt})$
p3-I	$[-5,5]$	$(2,-5,-5,5,5,-2)$	-30910.4240
p3-II	$[-10,10]$	$(2,-10,-10,10,9,-6)$	-392013.9740
p3-III	$[10,30]$	$(10,10,10,30,29,26)$	-41752008.4528
p3-IV	$[-30,-10]$	$(-30,-30,-30,-30,-29,-26)$	-10414515.1499

Tabela 4: Limites inferiores e superiores, pontos ótimos e ótimos globais do *benchmark* 3.

4.2. Resultados e discussões

Na tabela 5 são listados os resultados encontrados utilizando o algoritmo híbrido DE+NM e as colunas indicam inicialmente, a identificação do problema, na segunda coluna, o melhor resultado encontrado, na terceira, o pior resultado encontrado seguido da média e desvio padrão. Cada problema testado foi executado 100 vezes.

A tabela 6 apresenta a comparação das taxas de sucesso entre DE+NM e o algoritmo D&BMS proposta por Tian, Ma e Zhang (1998). A primeira coluna indica o problema testado, a segunda coluna indica a taxa de sucesso obtida pelo DE+NM, e a última coluna indica a taxa de sucesso do algoritmo D&BMS. Números em negrito indicam o melhor resultado.

Nota-se pela tabela 6 que DE+NM obteve altas taxas de sucesso em todas as instâncias de p1 e p2. Com taxas de sucesso de 96% em p1-I e p1-II e 100% em p1-III, DE+NM superou a eficácia do D&BMS. Além disso, todas as taxas de sucesso em p2 foram 100%, igualando a eficácia do D&BMS nas instâncias II e III, e superando em 1% na instância I. Por outro lado, DE+NM obteve baixas taxas de sucesso no problema p3, superando o D&BMS somente na instância p3-III com a taxa de 42%.

Problema	Melhor	Pior	Média	Desvio padrão
p1-I	-10.1527	-2.68	-9.92	1.13
p1-II	-10.4023	-2.75	-10.15	1.24
p1-III	-10.5358	-10.5358	-10.5358	0
p2-I	-3183.995	-3183.99	-3183.99	0
p2-II	-5847.996	-5847.99	-5847.99	0
p2-III	-9303.997	-9303.99	-9303.99	0
p3-I	-30910.424	-29799.09	-30823.74	185.83
p3-II	-392013.974	-207707.27	-379553.98	27112.40
p3-III	-41752008.452	-23461107.89	-40553069.87	2596259.47
p3-IV	-10414515.149	-7028152.79	-10036261.94	497364.24

Tabela 5: Resultados obtidos utilizando o algoritmo híbrido DE+NM.

Problema	DE+NM(%)	D&BMS(%)
p1-I	**96**	25
p1-II	**96**	29
p1-III	**100**	24
p2-I	**100**	99
p2-II	**100**	**100**
p2-III	**100**	**100**
p3-I	35	**80**
p3-II	9	**42**
p3-III	**42**	39
p3-IV	33	**48**

Tabela 6: Comparação entre DE+NM e D&BMS.

Problema	Tempo médio(s)
p1-I	3.96
p1-II	3.56
p1-III	3.57
p2-I	3.51
p2-II	4.69
p2-III	3.57
p3-I	5.43
p3-II	5.37
p3-III	5.42
p3-IV	5.51

Tabela 7: Tempo médio de execução para os problemas de teste.

O tempo médio de execução de cada problema testado é mostrado na tabela 7. Os tempos são indicados em segundos. A maior média de tempo obtida foi de 5.51 segundos no problema p3 que possui $D = 6$ e as execuções mais rápidas foram obtidas em p1, com a melhor média de 3.56 segundos.

5. Conclusão

Este trabalho apresenta o algoritmo híbrido DE+NM para problemas de otimização inteira não linear. Três *benchmarks* foram utilizados e o ótimo global foi encontrado em todos eles. DE+NM mostrou-se eficaz para o problema p1 pois obteve altas taxas de sucesso, 96% para p1-I e p1-II, e 100% para p1-III. O problema p2 obteve as melhores taxas com 100% de acerto em to-

das as instâncias. Por último, o problema p3 foi o mais desafiador, o algoritmo híbrido DE+NM obteve a maior taxa de acerto em p3-III, com 42% sendo superado pelo D&BMS em p3-I, p3-II e p3-IV. Uma sugestão para futuros trabalhos é investigar esta abordagem utilizando problemas de otimização inteira não lineares com restrições.

Referências

Aurora, J.S.; Huang, M.W.; Hsieh, C.C. (1994), Methods for optimization variables: a review, *Structural Optimization*, v. 8, p. 69-85.

Brea, E. (2013), Una extensión del método de Nelder-Mead a problemas de optimización no lineales enteros mixtos, *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería*, vol. 29, p. 163-174.

Burke, E.K.; Li. J.; Qu, R. (2014), A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems, *European Journal of Operational Research*, v. 23, p. 484-493.

Das, S.; Suganthan, P. N. (2011), Differential evolution: a survey of the state-of-the-art, IEEE *Transactions on Evolutionary Computation*, v. 14, n. 1, p. 4-31.

García, L.L.; García-Ródenas, R.; Gómez, G.A. (2014), Hybrid meta-heuristic optimization algorithms for time-domain-constrained data clustering, *Applied Soft Computing*, v. 23, 319-332.

Gomory, R. (1960), An algorithm for the mixed integer problem, *Technical Report RM-2597*, The Rand Corporation.

Gupta, O.K.; Ravindran, A. (1983), Nonlinear integer programming and discrete optimization, *Journal of Mechanical Design*, v. 105, p. 160-164.

Land, A.H.; Doig, A.G. (1960), An automatic method of solving discrete programming problems, *Econometrica*, v. 28, n. 3, p. 497-520.

Li, D.; Wang, J.; Sun, X.L. (2007), Computing exact solution to nonlinear integer programming: convergent lagrangian and objective level cut method, *Journal of Global Optimization*, v. 39, p. 127-154.

Li, X. (2010), Niching without niching parameters: particle swarm optimization using a ring topology. *IEEE Transactions on Evolutionary Computation*, v. 14, n. 1, p. 150-169.

Nasab, H.H. (2014), A hybrid fuzzy-GA algorithm for the integrated machine allocation problem with fuzzy demands, *Applied Soft Computing*, v. 23, p.417-431.

Nelder, J.A.; Mead R. (1965), A simplex method for function minimization, *The Computer Journal*, v. 13, p. 308-313.

Ponsich, A.; Coello, C.A.C. (2013), A hybrid differential evolution-tabu search algorithm for the solution of job-shop scheduling problems, *Applied Soft Computing*, v. 13, p. 462-474.

Powell, M.J.D. (1964), An efficient method for finding the minimum of a function of several variables without calculating derivates, *The Computer Journal*, v. 7, p. 155-162.

Schneider, E.R.F.A.; Krohling R.A. (2014), A hybrid approach using TOPSIS, differential evolution, and tabu search to find multiple solutions of constrained non-linear optimization problems, *Knowledge-Based Systems*, v. 62, p. 47-56.

Storn, R.; Price, K. (1995), Differential evolution - a simple and efficient adaptative scheme for global optimization over continuous space, *Technical Report TR-95-012*, ICSI.

Tian, P.; Ma, J.; Zhang, D. (1998), Non-linear integer programming by darwin and boltzmann mixed strategy, *European Journal of Operational Research*, v. 105, p. 224-235.

Wu, G.; Qiu, D.; Yu, Y.; Pedrycz, W.; Ma, M.; Li, H. (2014), Superior solution guided particle swarm optimization combined with local search techniques, *Expert Systems with Applications*, v. 41, p. 7536-7548.