ALGORITMO HEURÍSTICO APLICADO AO PROBLEMA DA DIVERSIDADE MÁXIMA

José André de Moura Brito
Escola Nacional de Ciências – ENCE/IBGE
Rua André Cavalcanti, 106, Bairro de Fatima – Centro – Rio de Janeiro- RJ
jambrito@gmail.com

Augusto César Fadel
Instituto Brasileiro de Geografia e Estatística – IBGE
Avenida República do Chile, 500 – Centro – Rio de Janeiro-RJ
augustofadel@gmail.com

Gustavo Silva Semaan
Instituto do Noroeste Fluminense de Educação Superior (INFES)da Universidade Federal Fluminense(UFF)
Av. João Jasbick, s/nº - Aeroporto - Santo Antônio de Pádua -RJ
gustavosemaan@gmail.com

RESUMO

Este artigo traz a proposta de um algoritmo heurístico que foi aplicado à resolução do problema da diversidade máxima (PDM). Neste problema, dado um conjunto *N*, deve-se selecionar um subconjunto $M \subset N$ de forma que a diversidade, calculada a partir dos elementos de *M*, seja máxima. Com o objetivo de produzir soluções competitivas frente a outros algoritmos da literatura, foi desenvolvido um algoritmo que utilizou os conceitos da metaheurística BRKGA. A partir de experimentos realizados com um conjunto de 140 instâncias da literatura (disponíveis na biblioteca MDP), observou-se que o algoritmo proposto produziu soluções de boa qualidade, quando comparadas às melhores soluções disponíveis na literatura.

PALAVRAS CHAVE. Diversidade Máxima, Metaheurísticas e BRKGA.

Tópicos MH, PM e OC

ABSTRACT

This paper presents a heuristic algorithm to solve the maximum diversity problem (MDP). In this problem, given a set *N*, one must select a subset $M \subset N$ so that the diversity, calculated from the elements of *M*, is maximized. Aiming to produce competitive solutions in comparison to other algorithms in the literature, an algorithm that uses the concepts of the BRKGA metaheuristic was developed. From the experiments performed with a set of 140 datasets of the literature (available on the MDP library), it was observed that the proposed algorithm produced good quality solutions, when compared to the best solutions available in the literature.

KEYWORDS. Maximum Diversity, Metaheuristics and BRKGA.
Topics MH, MP and OC

1. Introdução

Muitos dos problemas de localização, que dizem respeito à alocação de facilidades em uma rede, têm como objetivo a minimização de uma função que quantifica as distâncias entre as facilidades ou entre as facilidades e os nós da rede [Handler e Mirchandani 1979], representados por “clientes”. Essas facilidades, por sua vez, podem estar associadas, por exemplo, a hospitais, escolas, quarteis de bombeiros etc. Todavia, há casos nos quais a proximidade entre as facilidades não é desejável, sendo necessário alocá-las de forma a maximizar uma função que permite avaliar

as distâncias entre as mesmas. Como exemplos onde a proximidade entre as facilidades não é desejável, temos a localização de usinas nucleares, tanques de óleo e depósitos de armamento bélico [Kuby 1987]. Em trabalhos mais antigos da literatura, problemas com esta característica têm sido referenciados por diferentes nomes como, por exemplo, o problema de dispersão [Erkut 1990], o problema de dispersão máxima [Kuby 1987], o problema da *p*-defesa [Moon e Chaudhry 1984] etc. Já em trabalhos mais recentes, este problema tem sido referenciado por Problema da Diversidade Máxima, doravante denotado por PDM.

Resolver o PDM consiste em selecionar, a partir de um conjunto *N* (formado por *n* elementos) um subconjunto *M* (formado por *m* elementos), de forma que a soma das distâncias entre os elementos de *M*, tomados dois a dois, seja máxima. De acordo com [Kuo et al. 1993], a definição da distância a ser utilizada está associada à aplicação considerada. O PDM, que pertence à classe de problemas NP-difíceis [Glover et al. 1977], tem várias aplicações importantes propostas na literatura [Erkut e Neuman 1991], [Kuo et al. 1993], [Glover et al. 1998]. Face à complexidade e às inúmeras aplicações deste problema, encontram-se, na literatura, diversos trabalhos que trazem propostas de algoritmos heurísticos e metaheurísticas para a sua resolução.

No particular caso deste trabalho, de forma a resolver o PDM, foi desenvolvido um algoritmo heurístico que utiliza os conceitos da metaheurística algoritmos genéticos de chaves aleatórias viciadas [Gonçalves e Resende, 2011]. Além da introdução, este artigo está dividido em mais quatro seções. A seção dois traz uma descrição detalhada do PDM, incluindo uma revisão de literatura e a apresentação de uma formulação de programação linear inteira mista, que foi proposta por [Kuo et al. 1993] para resolver este problema. A seção três apresenta a metaheurística algoritmos genéticos de chaves aleatórias viciadas (do inglês *biased random key genetic algorithm*) e a seção quatro traz a descrição do algoritmo BRKGA que foi desenvolvido para a resolução do PDM. Concluindo o trabalho, a seção cinco traz um conjunto de resultados e análises concernentes à aplicação do novo algoritmo em um conjunto de 140 instâncias da literatura. Os resultados do algoritmo BRKGA foram comparados aos melhores resultados alcançados por outros algoritmos da literatura, disponíveis na biblioteca MDPLIB, incluindo os dois algoritmos propostos no trabalho recente de [Moro 2017]. Nos dois casos, o novo algoritmo teve uma boa performance.

2. Problema da Diversidade Máxima

No problema da diversidade máxima deve-se selecionar, a partir de um conjunto *N* formado por *n* elementos, um subconjunto *M*, formado por *m* elementos ($M \subset N$), de forma que os elementos de *M* possuam maior diversidade entre si. Esta diversidade, por sua vez, pode ser definida a partir de uma medida distância, que é função dos *l* atributos dos elementos de *N*.

Formalizando o PDM, considere que x_i e x_j sejam dois elementos quaisquer selecionados de *N* para compor *M*. Além disso, $x_i=(x_{i1},x_{i2},...,x_{il})$ e $x_j=(x_{j1},x_{j2},...,x_{jl})$ são os vetores com *l* atributos de x_i e x_j. Conforme comentado em [Silva et al. 2006], "O índice de diversidade representa o grau de diferença existente entre os atributos dos elementos x_i e x_j, ou seja, a distância entre eles, que deve ser calculada usando-se uma dentre as métricas de distância existentes. De acordo com a área de aplicação, uma métrica pode refletir melhor o problema que outra [Kuo et al. 1993]."

Em particular, adotando-se a métrica euclidiana, tem-se que a diversidade d_{ij} entre dois elementos x_i e x_j, pode ser definida por:

$$d_{ij} = \sqrt{\sum_{f=1}^{l}(x_{if} - x_{jf})^2} \quad (1)$$

A partir do cálculo das distâncias entre todos os elementos do conjunto *N*, tomados dois a dois, define-se a matriz simétrica $D_{nxn}=[d_{ij}]$. Considerando esta matriz, o valor da diversidade do subconjunto *M*, em relação aos seus *m* elementos, pode ser definido por:

$$div(M) = \sum_{\forall x_i,x_j \in M \subset N, i<j} d_{ij} \quad (2)$$

O PDM, que pertence à classe de problemas NP-difíceis [Glover et al. 1977], tem sido abordado em diversos trabalhos da literatura, com diferentes nomes como, por exemplo, problema da dispersão máxima [Kuby 1987], problema da *p*-dispersão [Erkut 1990], *p*-defesa [Moon e Chaudhry 1984], dispersão equitativa [Prokopyev et al. 2009] etc. Ainda neste sentido, encontram-se, na literatura, diversas aplicações para o PDM como: Localização [Erkut e Neuman 1991], Tratamentos médicos [Kuo et al. 1993], Projeto de Produtos [Glover et al. 1998] etc.

O PDM pode ser formulado como um problema de programação quadrática 0-1 que, após a aplicação de uma linearização [vide Kuo et al. 1993], pode ser representado pela formulação de programação linear inteira abaixo.

$$Max\ z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} w_{ij} \tag{3}$$

$$\sum_{i=1}^{n} y_i = m \tag{4}$$

$$y_i + y_j - w_{ij} \leq 1 \quad i = 1, \dots, n-1, j = i+1, \dots, n \tag{5}$$

$$w_{ij} \leq y_i \qquad i = 1, \dots, n-1, j = i+1, \dots, n \tag{6}$$

$$w_{ij} \leq y_j \qquad i = 1, \dots, n-1, j = i+1, \dots, n \tag{7}$$

$$w_{ij} \in \{0,1\}\ \ i = 1, \dots, n-1, j = i+1, \dots, n \tag{8}$$

$$y_i \in \{0,1\} \qquad i = 1, \dots, n \tag{9}$$

A função maximizada em (3) corresponde à função objetivo definida em (2). A variável binária y_i assume valor 1 quando o i-*ésimo* elemento do conjunto *N* é selecionado para o subconjunto *M*. A restrição (4) garante que serão selecionados, exatamente, *m* elementos do conjunto *N* para o subconjunto *M*. As restrições (5), (6) e (7) garantem que w_{ij} assume o valor 1 se, simultaneamente, y_i e y_j assumem valor 1, ou seja, se os elementos *i e j* são selecionados para o conjunto *M*, o que implica que a distância entre estes dois elementos é considerada no computo da função objetivo. Esta formulação tem o número de variáveis e de restrições definidos, respectivamente, por: $n(n-1)/2 + n$ e $3n(n-1)/2+1$. A aplicação da formulação acima ou de uma enumeração exaustiva torna-se impraticável à medida que *n* aumenta. No caso da enumeração, de forma a garantir o ótimo global em relação à diversidade máxima, seria necessário avaliar a função objetivo definida na equação (2) para um total de C_n^m subconjuntos *M*, ou seja $\frac{n!}{(n-m)!m!}$.

Por conta da complexidade e das inúmeras aplicações do PDM, têm sido desenvolvidos, nas últimas décadas, vários trabalhos que trazem propostas de algoritmos baseados em heurísticas e metaheurísticas. Neste sentido, o artigo de [Martí et al. 2013] constitui-se como uma excelente referência, apresentando uma substancial revisão da literatura quanto aos métodos propostos para o PDM. Mais especificamente, são descritos 30 algoritmos que foram aplicados a um conjunto de 315 instâncias da literatura. Estes algoritmos, por sua vez, estão divididos em dez heurísticas, sendo oito construtivas e duas de melhoria, além de 20 algoritmos baseados em metaheurísticas. Como exemplo de uma heurística, temos o trabalho de [Erkut 1990], que traz a proposta de um algoritmo construtivo e de melhoria. Este algoritmo constrói uma solução inicial para o conjunto *M*, com os dois elementos de *N* mais afastados entre si (considerando as distâncias de *D*), sendo adicionado, ao conjunto *M*, em cada uma das *(m-2)* iterações seguintes, o elemento de *M* que produz o maior acréscimo à função objetivo.

Em relação aos algoritmos baseados em metaheurísticas temos: GRASP [Duarte e Martí 2007] e [Santos et al. 2005], o GRASP com reconexão por caminhos [Silva et al. 2007], Busca Tabu [Palubeckis 2007] e [Aringhieri et al. 2008], Simulated Annealing [Kincaid 1992], [Brimberg 2009], Scatter Search [Aringhieri e Cordone 2006] e [Gallego et al. 2009], Algoritmos Meméticos [Katayama e Narihisa 2005] e, mais recentemente, o trabalho de [Aringhieri e Cordone 2011] que traz a proposta de um algoritmo VNS. Dentre estes algoritmos, destacamos:

https://proceedings.science/p/85527?lang=pt-br

(i) o algoritmo baseado em Busca Tabu, denominado ITS, proposto por [Palubeckis 2007], que alterna a aplicação de um procedimento de Busca Tabu com procedimentos de perturbação e (ii) o algoritmo proposto por [Gallego et al 2009], baseado em uma variante do Scatter Search, denotado por G_SS. Mais recentemente, temos o trabalho de dissertação de [Moro 2017], que implementou um algoritmo que combina o GRASP com reconexão por caminhos, e outro baseado nos algoritmos genéticos de chaves aleatórias viciadas (BRKGA).

3. Metaheurística BRKGA

O algoritmo genético de chaves aleatórias viciadas (do inglês, *Biased Random-Key Genetic Algorithm* - BRKGA), proposto por [Gonçalves e Resende 2011], é uma metaheurística que tem sido aplicada em diversos problemas de otimização [Brito et al. 2015], [Festa 2013] e [Oliveira, Chaves e Lorena 2017]. Em um BRKGA a população é constituída por p cromossomos, representados por vetores de chaves aleatórias, cujas componentes têm valores reais gerados, aleatoriamente, segundo uma distribuição uniforme [0,1]. Em cada geração do BRKGA são aplicados, nesses vetores, um procedimento denominado decodificador, seguido dos procedimentos de seleção, cruzamento e mutação. O decodificador, que consiste em um procedimento feito especificamente para cada problema ao qual o BRKGA é aplicado, transforma cada um destes vetores em uma solução viável para o problema de otimização.

Em linhas gerais, um BRKGA parte, em sua primeira geração, de uma população inicial composta de p vetores de chaves aleatórias, gerando novos vetores em cada uma das gerações seguintes. Mais especificamente, em cada geração, após a aplicação do decodificador e o cálculo da função objetivo, aplica-se o procedimento de seleção. A partir deste procedimento, a população é particionada em dois conjuntos, a saber: um conjunto menor C_e denominado elite, formado pelos p_e vetores de chaves aleatórias (correspondente a um percentual de p), associados às melhores soluções segundo o valor da função objetivo, e um conjunto complementar formado pelos p-p_e vetores restantes, denominado conjunto não-elite C_{ne}, sendo $p_e < p$-p_e.

A população que será utilizada na geração seguinte é composta por três conjuntos de vetores, quais sejam: (i) os p_e vetores elite do conjunto C_e definido na geração anterior; (ii) p_m vetores chamados de mutantes, que nada mais são do que vetores de chaves aleatórias gerados de forma equivalente aos vetores da população inicial e (iii) p-p_e-p_m vetores "filhos", obtidos a partir da aplicação do cruzamento uniforme [Spears e De Jong 1991]. Cada um destes vetores é produzido selecionando-se, para o cruzamento, um vetor v_1 de C_e e um vetor v_2 de C_{ne}. Uma vez selecionados esses dois vetores, é gerado um vetor auxiliar v_a com valores reais selecionados aletoriamente do intervalo $[0,1]$ e de igual tamanho a v_1 e v_2. Também é definido o valor de $\rho_e > 0.5$ correspondente à probabilidade de uma chave (elemento) de v_1 compor o vetor filho (v_f). Cada valor de v_a é comparado com ρ_e tal que, se o valor na i-ésima posição de v_a for menor ou igual a ρ_e, temos que a i-ésima posição do vetor filho herda o valor da i-ésima posição de v_1. Em caso contrário, a i-ésima posição do vetor filho herda o valor da i-ésima posição de v_2.

A Figura 1 ilustra a evolução de uma população de uma geração g para a geração $g+1$, considerando os três conjuntos de vetores citados acima; seguido pelo Quadro 1 traz um pseudocódigo que resume todos os passos do BRKGA.

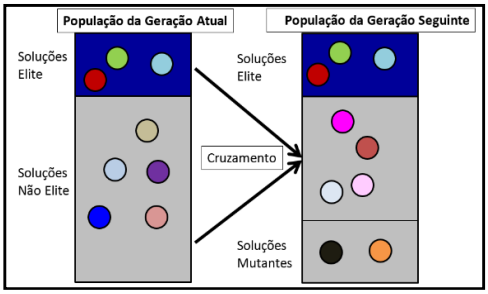


Figura 1 - Representação do BRKGA com a transição entre a geração g e a geração g+1.
(Baseada em figura utilizada na palestra de Maurício Resende [Resende 2013]).

1) Gerar a população inicial composta por p vetores de chaves aleatórias.
2) Aplicar o decodificador em cada vetor, produzindo soluções viáveis para o problema de otimização.
3) Calcular o valor da função objetivo para cada solução e ordenar, decrescentemente, as soluções de acordo com o valor da função objetivo (considerando um problema de maximização).
4) Selecionar os p_e vetores de chaves aleatórias correspondentes às melhores soluções ordenadas da população na geração g e copiar estes vetores para a população da geração seguinte.
5) Gerar p_m vetores mutantes, seguindo o mesmo procedimento do passo 1 e adicionar à população da próxima geração.
6) Cruzar os p_e vetores com os (p-p_e) vetores restantes (não elite) da população atual, produzindo (p-p_e-p_m) vetores de chaves aleatórias e adicionar estes vetores à população da próxima geração.
7) Se o critério de parada não for atendido (número de gerações, por exemplo), retornar ao passo 2.

Quadro 1 – Pseudocódigo do BRKGA

4. Algoritmo BRKGA para o PDM – Representação, Decodificação e Cálculo da Função

Pelo exposto na seção anterior, a implementação de um algoritmo baseado no BRKGA depende, basicamente, da representação dos cromossomos e da especificação do procedimento decodificador, que produz as soluções viáveis para o problema de otimização em questão.

O decodificador proposto neste artigo é bem similar ao proposto no trabalho de [Moro 2017], uma vez que produz a mesma solução viável inicial, a menos da forma de representação dos vetores solução. Mais especificamente, após a decodificação, no trabalho de [Moro, 2017] é produzido um vetor (cromossomo) de tamanho n e com valores 0 e 1, com 1 indicando que o i-ésimo elemento (objeto) será alocado ao conjunto M. Conforme será visto a seguir, o grande diferencial do decodificador deste artigo diz respeito ao procedimento de busca local aplicado em cada solução viável inicial (solução decodificada) e no procedimento de decodificação inversa, que atualiza os vetores de chaves aleatórias que serão novamente decodificados nas gerações seguintes.

Assim sendo, no que diz respeito ao decodificador proposto para o PDM, neste trabalho, cada cromossomo foi definido como um vetor u com n posições (número elementos do conjunto N) preenchidas com valores reais gerados segundo uma uniforme [0,1], sendo a i-ésima posição de u correspondente ao número do elemento x_i que pertence ao conjunto N. Uma vez gerado o vetor u, calcula-se o seu P-quantil, definido por $q(P)$ [Bussab e Morettin 2005]. Dado um vetor y (com n observações em ordem crescente), define-se como P-quantil, com $0 < P < 1$, o valor $q(P)$ tal que 100P% dos elementos de y que são menores ou iguais a $q(P)$ e os restantes 100(1 - P)% dos elementos de u que são maiores ou iguais a $q(P)$. Em linhas gerais, os quantis dividem os valores de y em subconjuntos de dados de dimensão essencialmente igual, dando origem a pontos de corte, dados por $q(P)$, que determinam as fronteiras entre os subconjuntos consecutivos. São exemplos particulares de P-quantis os quartis (P={0.25, 0.50, 0.75}), que particionam o conjunto de elementos em quatro subconjuntos, e decis (P={0.1, 0.2, ..., 0.9}), que particionam o conjunto de elementos em dez subconjuntos. Considerando o vetor y definido acima, temos que a fórmula para o cálculo de $q(P)$ é dada pela expressão abaixo:

$$q(P) = \begin{cases} y_{nP+1} & \text{se nP não é inteiro} \\ \frac{(y_{nP}+y_{nP+1})}{2} & \text{se nP é inteiro} \end{cases} \quad (10)$$

No que concerne ao decodificador proposto, o valor associado a *P* foi *m/n*. Assim sendo, após a geração do vetor *u*, seus valores são colocados em ordem crescente e atribuídos a um vetor *y*. Dado que $P=m/n \Rightarrow nP$ corresponde ao inteiro *m*, o que implica $q(P)=(y_m+y_{m+1})/2$.

Uma vez calculado *q(P)*, efetua-se uma busca em *u*, para determinar as entradas u_i ($i=1,...,n$) tais que $u_i \leq q(P)$. Os índices associados aos elementos u_i, que estão entre as posições 1 e *m* de *y*, corresponderão aos elementos de *N*, inicialmente selecionados para o conjunto *M*. Estes índices são atribuídos a um vetor *z* e os demais índices, associados aos elementos u_i cujas entradas estão entre as posiçoes $m+1$ e *n* de *y* (os pertencentes ao conjunto *N-M*), são atribuídos a um vetor *s* ($s=(s_1,s_2,...,s_k)$ $(k=n-m)$). A partir dos elementos de *z*, calcula-se a diversidade de *M* de acordo com a equação (2), apresentada na seção 2. O vetor *s* é definido para ser utilizado posteriormente em um procedimento de busca local simples, que foi incorporado ao decodificador. Este procedimento tem como objetivo tornar o BRKGA, simultaneamente, mais eficaz e eficiente, no sentido de produzir soluções de boa qualidade, não necessitando de um substancial número de gerações e de uma população grande. Basicamente, efetua trocas entre os elementos de *z* (conjunto *M*) e os elementos de *s* (conjunto *N-M*). A ideia é avaliar se alguns dos elementos de *M* podem ser substituídos por elementos de *N-M*, de forma que a função definida em (11) tenha um acréscimo, o que implica aumento na diversidade do conjunto *M*.

Com este objetivo em mente, são selecionados *w* ($w=mínimo\ \{m,50\}$) (valor 50 foi definido experimentalmente) elementos do vetor *z*, definindo-se o vetor *x*, tal que $x=(x_1,...,x_r,...,x_w)$. Em seguida, seleciona-se o 1º elemento de *x* (x_1) e calcula-se a soma das distâncias deste elemento aos demais elementos de *M* (entre as posições *1* e *m* de *z*) através da equação (11).

$$f_{x_1} = \sum_{\forall x_j \in M, x_j \neq x_1} d_{x_1 x_j} \tag{11}$$

De igual forma, calcula-se a soma das distâncias de cada elemento s_i de *s* aos demais elementos de *M*, excluindo-se o elemento x_1:

$$f_{s_1} = \sum_{\forall x_j \in M, x_j \neq x_1} d_{s_1 x_j},\ f_{s_2} = \sum_{\forall x_j \in M, x_j \neq x_1} d_{s_2 x_j},\ ...,\ f_{s_k} = \sum_{\forall x_j \in M, x_j \neq x_1} d_{s_k x_j} \tag{12}$$

Ao final deste passo, a partir dos valores de $f_{x_1}, f_{s_1}, \dots, f_{s_k}$, calcula-se mínimo de:

$$\{f_{x_1} - f_{s_1}, f_{x_1} - f_{s_2}, ..., f_{x_1} - f_{s_i}, \dots, f_{x_1} - f_{s_k}\} \tag{13}$$

Caso este mínimo seja negativo, significa que a substituição de x_1 por um elemento s_i de *s* produz acréscimo em *f*. Neste caso, são atualizados: (i) os vetores *x* e *z*, substituindo-se x_1 por s_i; (ii) o vetor *s*, substituindo-se s_i por x_1 e (iii) a função objetivo a partir de expressão:

$$f = f - f_{x_1} + f_{s_i} \tag{14}$$

Igual procedimento é aplicado aos demais elementos de *x*. Assim sendo, para o *r*-ésimo elemento de *x* são avaliadas as seguintes expressões:

$$f_{x_r} = \sum_{\forall x_j \in M, x_j \neq x_r} d_{x_r x_j}, \tag{15}$$

$$f_{s_1} = \sum_{\forall x_j \in M, x_j \neq x_r} d_{s_1 x_j},\ f_{s_2} = \sum_{\forall x_j \in M, x_j \neq x_r} d_{s_2 x_j},\ ...,\ f_{s_k} = \sum_{\forall x_j \in M, x_j \neq x_r} d_{s_k x_j}, \tag{16}$$

$$\{f_{x_r} - f_{s_1}, f_{x_r} - f_{s_2}, ..., f_{x_r} - f_{s_i}, \dots, f_{x_r} - f_{s_k}\} \quad \text{e} \quad f = f - f_{x_r} + f_{s_i} \tag{17}$$

Depois de avaliar os *w* elementos do vetor *x* na fase de busca local, a solução final associada à *z* (a nova configuração do conjunto *M*) deve ter correspondência com o vetor de chaves aleatórias *u*, inicialmente utilizado para gerar *z* e *s*. Para garantir esta correspondência, e possibilitar que este novo vetor *u* seja utilizado nas gerações seguintes; após a avaliação de cada

https://proceedings.science/p/85527?lang=pt-br

elemento de *x* e atualização em *z*, realiza-se atualização correspondente em *u*, o que implica uma "decodificação inversa" [Silva e Resende 2013]. Mais especificamente, partindo do fato de que os valores em *z* e *s* correspondem aos índices de *u*, para cada troca de valores entre as posições *1* e *m de z e 1 e (n-m) de s*, efetua-se a seguinte atualização em *u*: *auxiliar* ←*u[z[i]], u[z[i]]* ←*u[s[j]], u[s[j]]* ←*auxiliar*.

De forma a ilustrar todos os passos associados à aplicação do procedimento decodificador contemplando, inclusive, a busca local e a decodificação inversa, considere o exemplo da Figura 2, com *n*=10, *m*=5 e *P=m/n*=0,5.

Vetores / Índices	1	2	3	4	5	6	7	8	9	10
u	0,410	0,622	0,552	0,160	0,197	0,519	0,952	0,368	0,947	0,431
y	0,160	0,197	0,368	0,410	0,431	0,519	0,552	0,622	0,947	0,952
q(0.5)=(y_5+y_6)/2	0,475									
z [índices de u tq $u_i \leq q(0.5)$] (Conjunto M)	1	4	5	8	10					
x	1	4	5	8	10					
s (representa N-M)	2	3	6	7	9					

Figura 2 – Representação e decodificação de uma solução

A partir dos vetores *x* e *s* da Figura 2, e supondo, inicialmente, $f = 100$, $f_{x_1} = 50$, $f_{s_1} = 30$, $f_{s_2} = 45, f_{s_3} = 60, f_{s_4} = 10\ , f_{s_5} = 20$, ao calcular-se o mínimo a partir da expressão (13), temos que o elemento associado à s_3 (elemento 6) é candidato a entrar no conjunto *M*. Por conseguinte, x_1 (x_1=1) será alocado ao conjunto *N-M*. Desta forma, atualizando-se *u* e os vetores z, *x* e *s* temos as novas configurações dos conjuntos *M* e *N-M*. No caso particular caso de *u*, efetua-se:

$$auxiliar \leftarrow u[x[1]],\ u[x[1]] \leftarrow u[s[3]] \text{ e } u[s[3]] \leftarrow auxiliar \Rightarrow$$

$$auxiliar \leftarrow u[1],\ u[1] \leftarrow u[6] \text{ e } u[6] \leftarrow u[1] \Rightarrow$$

$$auxiliar=0{,}410;\ u[1]=0{,}519 \text{ e } u[6]=0{,}410$$

Avaliando-se novamente as entradas de *u* tais que $u_i \leq q(0.5)$, temos: *z=(4,5,6,8,10)*

Vetores / Índices	1	2	3	4	5	6	7	8	9	10
u	0,519	0,622	0,552	0,160	0,197	0,410	0,952	0,368	0,947	0,431
y	0,160	0,197	0,368	0,410	0,431	0,519	0,552	0,622	0,947	0,952
q(0.5)=(y_5+y_6)/2	0,475									
z [índices de u tq $u_i \leq q(0.5)$] (Conjunto M)	4	5	6	8	10					
x atualizado	6	4	5	8	10					
s atualizado	2	3	1	7	9					

Figura 3 – Atualização dos vetores após a aplicação da decodificação e busca local

Considerando os vetores *x* e *s* na Figura 3, na iteração seguinte da busca local toma-se o elemento x_2 e efetua-se a avaliação de f_{x_2}, f_{s_1}, f_{s_2} , f_{s_3}, f_{s_4} e f_{s_5}, e assim sucessivamente, até que todos os elementos de *x* tenham sido avaliados.

Concluindo a exposição do BRKGA, ressalta-se que o cruzamento uniforme propicia a troca dos *m* primeiros elementos que definirão a configuração inicial dos conjuntos *M* e *N-M*.

5. Resultados Computacionais

De forma a avaliar o novo algoritmo BRKGA, doravante denotado por BRKGAPDM, foi realizado um experimento computacional, considerando a sua aplicação em conjunto de 140 instâncias, selecionadas dentre as 315 instâncias disponíveis na biblioteca MDPLIB (http://www.optsicom.es/mdp/#instances). Nesta biblioteca encontram-se, também, as melhores soluções produzidas para cada uma destas instâncias (maiores valores de diversidade). Tais soluções foram obtidas a partir da aplicação dos algoritmos G_SS proposto em [Gallego et al. 2009] e ITS proposto em [Palubeckis 2007], sendo estipulado, para cada instância e algoritmo, o

tempo máximo de execução de duas horas. Estes dois algoritmos foram executados em uma workstation Intel Core 2 Quad Q8300 com 6 GB de memória RAM e sistema operacional Linux (informações do site). O algoritmo BRKGAPDM foi implementado em *R* e executado em um computador com 16GB de memória RAM e dotado de quatro processadores INTEL I7-6700 de 3.4 GHz e sistema operacional Windows 10 (64 bits). Além da comparação das soluções do BRKGAPDM com as melhores soluções disponíveis na biblioteca MDPLIB, também foram considerados na comparação os resultados produzidos pelos algoritmos GRASP e BRKGA propostos em [Moro 2017], doravante denominados GRASPM e BRKGAM.

Considerando os recursos da arquitetura *multicore* do computador, e o fato de o software *R* ter o pacote *parallel (http://gforge.se/2015/02/how-to-go-parallel-in-r-basics-tips/)*, que agrega várias funções de paralelismo, paralelizou-se, no algoritmo BRKGAPDM, o procedimento decodificador que agrega a busca local.

Ainda em relação ao BRKGAPDM, os parâmetros utilizados em todos os experimentos foram definidos, tomando por base, as recomendações feitas em [Gonçalves e Resende 2011] e a partir de experimentos preliminares. Neste sentido, foram selecionadas, dentre as 140 instâncias, 10 instâncias sobre as quais o algoritmo foi aplicado 10 vezes, considerando todas as combinações (729) dos parâmetros: p=75, 100 e 150; p_e=0.10, 0.15, 0.20; p_m=p_e; ρ_e=0.7,0.8 e 0.9, g=250, 500 e 1000 e w_{min}=25, 50 e 100 (número de elementos selecionados de z para x). Além do número máximo de gerações associado ao parâmetro g, foram adotados, em todas as 72.900 execuções (parâmetros x instâncias x execuções), mais dois critérios de parada para o BRKGAPDM, quais sejam: 100 gerações sem melhoria (aumento no valor da função objetivo) e tempo máximo de CPU de cerca de duas horas (7.300 segundos).

Considerando cada combinação de parâmetros e cada instância, foi calculada a mediana associada aos valores de função objetivo obtido nas 10 execuções. A combinação dos parâmetros correspondente ao maior número de soluções com maiores valores de mediana (em todas as instâncias) foi adotada como a combinação padrão. Assim sendo, foram adotados, respectivamente, os seguintes valores para p, p_e, p_m, ρ_e, g e w_{min}: 150, 0.20, 0.20, 0.75, 500 e 50.

Antes da apresentação e análise dos resultados, no Quadro 2 há uma breve descrição das 140 instâncias (matrizes de distância D) utilizadas nos experimentos computacionais reportados neste trabalho.

Instância	Descrição
SOM-b	20 instâncias geradas por [Silva et al. 2004] e utilizada em outros artigos. Os tamanhos das instâncias são tais que: • Para n=100, m=10, 20, 30 e 40; • Para n=200, m=20, 40, 60 e 80; • Para n=300, m=30, 60, 90 e 120; • Para n=400, m=40, 80, 120 e 160; • Para n=500, m=50, 100, 150 e 200
GKD-c	20 instâncias geradas por [Duarte e Martí 2007], todas com n=500 e m=50
MDG-a	40 instâncias geradas por [Duarte e Martí 2007], sendo vinte com n=500 e m=50 e vinte com n=2.000 e m=200. Estas instâncias também foram utilizadas em [Palubeckis 2007]
MDG-b	40 instâncias geradas por [Duarte e Martí 2007], sendo vinte com n=500 e m=50 e vinte com n=2.000 e m=200. Estas instâncias também foram utilizadas em [Gallego et al. 2009]
MDG-c	20 instâncias similares às utilizadas em [Palubeckis 2007] com n=3.000 e m=300, 400, 500 e 600

Quadro 2 – Informações sobre as instâncias utilizadas nos experimentos

A Tabela 1 tem, em sua primeira e sexta colunas, a identificação de cada uma das 140 instâncias utilizadas nos experimentos com o BRKGAPDM. Nas 2ª e 7ª colunas temos o valor da função objetivo associado à melhor solução disponível na literatura, para cada instância (Best),

considerando a aplicação dos algoritmos G_SS e ITS. As colunas de 3 a 5 e 8 a 10 trazem os gaps (em percentual) do BRKGAPDM e dos algoritmos GRASPM e BRKGAM [Moro 2017], em relação à solução apresentada na 2ª e 6ª colunas.

Considerando os conjuntos GKD-c, SOM-b, MDGA-a e MDG-b (120 instâncias), e definindo como solução vencedora a melhor solução encontrada, incluindo a solução da literatura e as soluções dos três algoritmos, temos que o BRKGAPDM produziu soluções vencedoras em 43% dos casos, seguido pelos algoritmos BRKGAM com 23% e GRASPM com 18%.

Tabela 1 – Resultados dos algoritmos BRKGAPDM, GRASPM e BRKGAM

Instância	Best	BRKGAPDM	BRKGAM	GRASPM	Instância	Best	BRKGAPDM	BRKGAM	GRASPM
GKD-c_1_n500_m50	19.485,19	0,00	0,00	0,00	MDG-a_31_n2000_m200	114.139,00	0,45	1,16	0,68
GKD-c_2_n500_m50	19.701,54	0,00	0,00	0,00	MDG-a_32_n2000_m200	114.092,00	0,52	0,96	0,58
GKD-c_3_n500_m50	19.547,21	0,00	0,00	0,00	MDG-a_33_n2000_m200	114.124,00	0,57	1,37	0,73
GKD-c_4_n500_m50	19.596,47	0,00	0,00	0,00	MDG-a_34_n2000_m200	114.203,00	0,83	1,06	0,88
GKD-c_5_n500_m50	19.602,63	0,00	0,00	0,00	MDG-a_35_n2000_m200	114.180,00	0,51	1,08	0,84
GKD-c_6_n500_m50	19.421,55	0,00	0,00	0,00	MDG-a_36_n2000_m200	114.252,00	0,61	1,28	0,77
GKD-c_7_n500_m50	19.534,31	0,00	0,00	0,00	MDG-a_37_n2000_m200	114.213,00	0,49	1,13	0,80
GKD-c_8_n500_m50	19.487,32	0,00	0,00	0,00	MDG-a_38_n2000_m200	114.378,00	0,27	1,15	0,82
GKD-c_9_n500_m50	19.221,63	0,00	0,00	0,00	MDG-a_39_n2000_m200	114.201,00	0,52	1,46	0,73
GKD-c_10_n500_m50	19.703,35	0,00	0,00	0,00	MDG-a_40_n2000_m200	114.191,00	0,23	1,17	0,74
GKD-c_11_n500_m50	19.587,13	0,00	0,00	0,00	MDG-b_1_n500_m50	778.030,63	0,02	0,43	0,85
GKD-c_12_n500_m50	19.360,24	0,00	0,00	0,00	MDG-b_2_n500_m50	779.963,69	0,00	1,24	1,31
GKD-c_13_n500_m50	19.366,70	0,00	0,00	0,00	MDG-b_3_n500_m50	776.768,44	0,23	0,55	0,93
GKD-c_14_n500_m50	19.458,57	0,00	0,00	0,00	MDG-b_4_n500_m50	775.394,63	0,20	0,37	0,45
GKD-c_15_n500_m50	19.422,15	0,00	0,00	0,00	MDG-b_5_n500_m50	775.611,06	0,00	0,64	0,95
GKD-c_16_n500_m50	19.680,21	0,00	0,00	0,00	MDG-b_6_n500_m50	775.153,69	0,00	0,87	0,84
GKD-c_17_n500_m50	19.331,39	0,00	0,00	0,00	MDG-b_7_n500_m50	777.232,88	0,00	0,57	0,24
GKD-c_18_n500_m50	19.461,39	0,00	0,00	0,00	MDG-b_8_n500_m50	779.168,75	0,00	0,43	0,10
GKD-c_19_n500_m50	19.477,33	0,00	0,00	0,00	MDG-b_9_n500_m50	774.802,19	0,00	0,76	1,24
GKD-c_20_n500_m50	19.604,84	0,00	0,00	0,00	MDG-b_10_n500_m50	774.961,31	0,09	0,47	0,89
SOM-b_1_n100_m10	333,00	0,00	0,00	9,61	MDG-b_11_n500_m50	777.468,88	0,00	0,48	0,52
SOM-b_2_n100_m20	1.195,00	0,00	0,00	0,84	MDG-b_12_n500_m50	775.492,94	0,00	0,87	0,76
SOM-b_3_n100_m30	2.457,00	0,00	0,00	0,00	MDG-b_13_n500_m50	780.191,88	0,28	0,09	1,64
SOM-b_4_n100_m40	4.142,00	0,00	0,00	0,12	MDG-b_14_n500_m50	782.232,75	0,00	0,19	0,30
SOM-b_5_n200_m20	1.247,00	0,00	0,00	2,09	MDG-b_15_n500_m50	780.300,38	0,00	0,50	1,93
SOM-b_6_n200_m40	4.450,00	0,00	0,00	0,40	MDG-b_16_n500_m50	775.436,25	0,00	0,19	0,84
SOM-b_7_n200_m60	9.437,00	0,00	0,00	0,23	MDG-b_17_n500_m50	776.619,13	0,00	0,68	1,00
SOM-b_8_n200_m80	16.225,00	0,00	0,00	0,01	MDG-b_18_n500_m50	775.850,75	0,01	0,54	0,97
SOM-b_9_n300_m30	2.694,00	0,37	0,67	0,74	MDG-b_19_n500_m50	778.802,94	0,20	0,64	0,79
SOM-b_10_n300_m60	9.689,00	0,00	0,42	0,70	MDG-b_20_n500_m50	778.644,81	0,00	0,46	0,63
SOM-b_11_n300_m90	20.743,00	0,00	0,32	0,30	MDG-b_21_n2000_m200	11.299.894,86	0,59	1,29	0,74
SOM-b_12_n300_m120	35.881,00	0,01	0,01	0,02	MDG-b_22_n2000_m200	11.286.775,59	0,58	1,06	0,67
SOM-b_13_n400_m40	4.658,00	0,00	0,71	1,14	MDG-b_23_n2000_m200	11.299.940,87	0,62	1,17	0,71
SOM-b_14_n400_m80	16.956,00	0,19	0,14	0,25	MDG-b_24_n2000_m200	11.290.874,16	0,59	0,99	0,77
SOM-b_15_n400_m120	36.317,00	0,00	0,15	0,21	MDG-b_25_n2000_m200	11.296.066,66	0,42	1,04	0,77
SOM-b_16_n400_m160	62.487,00	0,04	0,14	0,14	MDG-b_26_n2000_m200	11.292.295,89	0,35	1,23	0,72
SOM-b_17_n500_m50	7.141,00	0,11	0,66	1,34	MDG-b_27_n2000_m200	11.305.676,85	0,52	1,07	0,86
SOM-b_18_n500_m100	26.258,00	0,08	0,17	0,20	MDG-b_28_n2000_m200	11.279.916,06	0,54	1,13	0,80
SOM-b_19_n500_m150	56.572,00	0,02	0,02	0,04	MDG-b_29_n2000_m200	11.297.188,33	0,39	1,26	0,71
SOM-b_20_n500_m200	97.344,00	0,10	0,04	0,15	MDG-b_30_n2000_m200	11.296.414,93	0,72	1,09	0,57
MDG-a_1_n500_m50	7.833,83	0,00	0,68	1,90	MDG-b_31_n2000_m200	11.288.901,49	0,61	1,21	0,87
MDG-a_2_n500_m50	7.771,66	0,28	0,80	1,70	MDG-b_32_n2000_m200	11.279.820,13	0,67	1,12	0,69
MDG-a_3_n500_m50	7.759,36	0,19	0,65	0,78	MDG-b_33_n2000_m200	11.296.297,93	0,58	1,17	0,77
MDG-a_4_n500_m50	7.770,24	0,07	1,07	1,23	MDG-b_34_n2000_m200	11.281.245,50	0,35	1,15	0,50
MDG-a_5_n500_m50	7.755,23	0,09	1,05	0,72	MDG-b_35_n2000_m200	11.307.424,26	0,48	1,14	0,76
MDG-a_6_n500_m50	7.773,71	0,00	0,43	1,17	MDG-b_36_n2000_m200	11.289.469,27	0,35	0,85	0,53
MDG-a_7_n500_m50	7.771,73	0,00	0,56	0,67	MDG-b_37_n2000_m200	11.290.544,79	0,48	1,25	0,75
MDG-a_8_n500_m50	7.750,88	0,15	0,44	0,90	MDG-b_38_n2000_m200	11.288.570,85	0,47	0,96	0,67
MDG-a_9_n500_m50	7.770,07	0,16	0,45	0,83	MDG-b_39_n2000_m200	11.295.054,20	0,37	1,35	0,61
MDG-a_10_n500_m50	7.780,35	0,02	0,80	1,40	MDG-b_40_n2000_m200	11.307.104,80	0,49	1,33	0,67
MDG-a_11_n500_m50	7.770,95	0,13	0,96	1,54	MDG-c_1_n3000_m300	24.884.110,00	0,32	*	*
MDG-a_12_n500_m50	7.757,65	0,09	0,57	0,51	MDG-c_2_n3000_m300	24.905.330,00	0,51	*	*
MDG-a_13_n500_m50	7.798,43	0,22	1,25	1,06	MDG-c_3_n3000_m300	24.900.820,00	0,44	*	*
MDG-a_14_n500_m50	7.795,63	0,00	0,56	1,13	MDG-c_4_n3000_m300	24.900.424,00	0,51	*	*
MDG-a_15_n500_m50	7.736,84	0,30	0,84	0,75	MDG-c_5_n3000_m300	24.899.703,00	0,39	*	*
MDG-a_16_n500_m50	7.792,77	0,27	0,52	1,94	MDG-c_6_n3000_m400	43.465.087,00	0,57	*	*
MDG-a_17_n500_m50	7.787,20	0,00	1,27	0,29	MDG-c_7_n3000_m400	43.472.804,00	0,36	*	*
MDG-a_18_n500_m50	7.756,26	0,00	0,70	1,61	MDG-c_8_n3000_m400	43.437.261,00	0,35	*	*
MDG-a_19_n500_m50	7.755,41	0,14	1,01	0,91	MDG-c_9_n3000_m400	43.437.861,00	0,46	*	*
MDG-a_20_n500_m50	7.733,86	0,14	0,63	1,02	MDG-c_10_n3000_m400	43.476.251,00	0,51	*	*
MDG-a_21_n2000_m200	114.259,00	0,51	1,17	0,93	MDG-c_11_n3000_m500	66.987.801,00	0,27	*	*
MDG-a_22_n2000_m200	114.327,00	0,58	1,39	0,60	MDG-c_12_n3000_m500	67.021.888,00	0,41	*	*
MDG-a_23_n2000_m200	114.123,00	0,54	1,11	0,79	MDG-c_13_n3000_m500	67.014.051,00	0,31	*	*
MDG-a_24_n2000_m200	114.040,00	0,53	1,21	0,72	MDG-c_14_n3000_m500	66.979.606,00	0,32	*	*
MDG-a_25_n2000_m200	114.064,00	0,48	1,04	0,72	MDG-c_15_n3000_m500	66.992.877,00	0,24	*	*
MDG-a_26_n2000_m200	114.204,00	0,33	1,06	0,62	MDG-c_16_n3000_m600	95.556.435,00	0,22	*	*
MDG-a_27_n2000_m200	114.338,00	0,70	1,35	0,76	MDG-c_17_n3000_m600	95.645.826,00	0,40	*	*
MDG-a_28_n2000_m200	114.158,00	0,48	1,30	0,77	MDG-c_18_n3000_m600	95.629.207,00	0,53	*	*
MDG-a_29_n2000_m200	114.132,00	0,34	1,28	0,63	MDG-c_19_n3000_m600	95.633.549,00	0,34	*	*
MDG-a_30_n2000_m200	114.197,00	0,39	1,31	0,79	MDG-c_20_n3000_m600	95.643.586,00	0,40	*	*

*Os dois algoritmos propostos em [Moro 2017] não foram aplicados no conjunto MDG-c

Dando continuidade à análise, a Tabela 2 traz, por conjunto de instâncias, as medidas resumo relacionadas aos gaps (em valores percentuais) dos 3 algoritmos em relação à solução literatura (Best). Novamente, o algoritmo BRKGAPDM teve um desempenho melhor frente aos algoritmos GRASPM e ao BRKGAM. Em particular, a partir desta tabela é possível observar que

os gaps médios e medianos do BRKGAPDM foram, em todos os casos, inferiores aos dos dois algoritmos, indicando soluções mais próximas das melhores soluções encontradas na literatura (inferiores 0,4%).

Tabela 2 – Medidas Resumo dos gaps dos Algoritmos por Conjunto de Instâncias

Conjunto	Algoritmo	Min.	1o Quartil	Mediana	Média	3o Quartil	Max.
SOM-b	BRKGAPDM	0,000	0,000	**0,025**	**0,083**	0,175	0,450
	BRKGAM	0,000	0,000	0,030	0,173	0,208	0,710
	GRASPM	0,000	0,135	0,240	0,927	0,765	9,610
MDG-a	BRKGAPDM	0,000	0,120	**0,275**	**0,303**	0,510	0,830
	BRKGAM	0,430	0,695	1,060	0,982	1,220	1,460
	GRASPM	0,290	0,720	0,790	0,924	1,030	1,940
MDG-b	BRKGAPDM	0,000	0,068	**0,320**	**0,315**	0,513	0,860
	BRKGAM	0,090	0,530	0,915	0,846	1,155	1,350
	GRASPM	0,100	0,660	0,760	0,783	0,863	1,930
MDG-c	BRKGAPDM	0,220	0,320	**0,395**	**0,393**	0,473	0,570

*As medidas para o conjunto GKD-c foram suprimidas, tendo em vista que todos os gaps foram de 0%

Ainda neste sentido, a Figura 4 traz um gráfico com os percentuais correspondentes aos casos (excluindo o conjunto MDG-c) em que o BRKGAPDM produziu soluções melhores, iguais ou piores à melhor solução produzida entre os algoritmos GRASPM e BRKGAM.

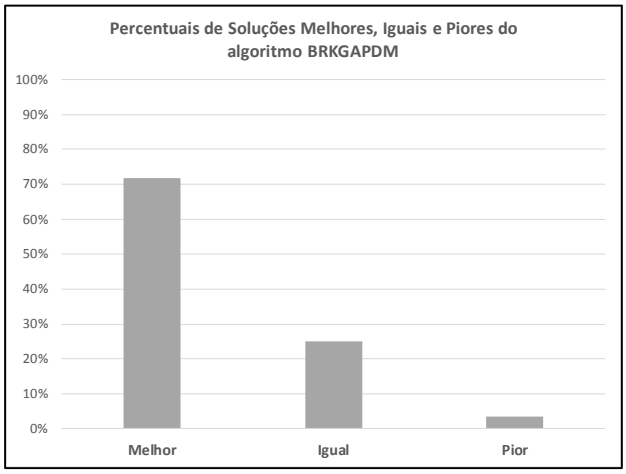


Figura 4 – Classificação das soluções do BRKGAPDM em relação os algoritmos BRKGAM e GRASPM

A Tabela 3 traz, por grupo de instâncias, as medidas resumo dos tempos de processamento do BRKGAPDM. Para o conjunto SOM o tempo mediano foi inferior a quatro minutos, seguido pelo conjunto GDC-c com tempo mediano inferior a oito minutos. Como era de se esperar, os conjuntos MDG-a e MDG-b, com um subconjunto de instâncias de médio porte (n=2000 e m=200), demandaram tempos de processamento maiores, sendo observados tempos medianos de 41 e 48 minutos respectivamente. Para o conjunto MDG-c, que agrega as instâncias de maior porte, todos os tempos foram da ordem de duas horas.

Tabela 3 – Medidas Resumo dos Tempos do BRKGAPDM por Conjunto de Instâncias

Conjunto	Min.	1o Quartil	Mediana	Média	3o Quartil	Max.
SOM-b	15	98	226	241	395	577
GKD-c	337	393	458	520	594	878
MDG-a	409	571	2.469	2.822	4.585	7.221
MDG-b	457	610	2.860	3.125	5.246	7.214
MDG-c	7.202	7.219	7.231	7.229	7.236	7.261

Neste trabalho foi apresentado um novo algoritmo para o PDM, considerando a utilização da metaheurística BRKGA. Este algoritmo (BRKGAPDM) foi aplicado em um expressivo conjunto de instâncias da literatura (total de 140), sendo observados bons resultados (gaps inferiores a 1%) para instâncias de porte variado em relação às cardinalidades de *N* e *M*. Estes resultados constituem-se como um bom indicador de que BRKGAPDM pode ser uma boa alternativa à resolução do problema de diversidade máxima. Em trabalhos futuros, serão implementadas e avaliadas novas decodificações, buscas locais mais sofisticadas considerando, inclusive, a metaheurística VNS.

Referências

Aringhieri, R. and R. Cordone. (2006).Better and faster solutions for the maximum diversity problem. Technical report, Universit degli Studi di Milano, Polo Didattico e di Ricerca di Crema.

Aringhieri, R.R. Cordone, and Y. Melzani. (2008). "Tabu search vs. grasp for the maximum diversity problem". *4OR: A Quarterly Journal of Operations Research* 6(1):45–60.

Aringhieri, R. and R. Cordone. (2011).Comparing local search metaheuristics for the maximum diversity problem. *Journal of the Operational Research Society* 62: 266-280.

Brimberg, J., N. Mladenovic, D. Urosevic and E. Ngai. (2009). Variable neighborhood search for the heaviest *k*-subgraph. *Computers & Operations Research* 36(11): 2885-2891.

Brito, J.A.M, Semaanm G.S. e Brito, L.R. (2015). Resolução do Problema dos k-medoids Via Algoritmo Genético de Chaves Aleatórias Viciadas. *Revista Pesquisa Naval*, 27:126-142.

Bussab, W.O. e Morettin, P.A. (2005). Estatística Básica, 5ª edição, Editora Saraiva.

Duarte, A. e R. Martí. (2007).Tabu search and grasp for the maximum diversity problem. *European Journal of Operational Research* 178: 71–84.

Erkut, E. (1990). The discrete p-dispersion problem. *European Journal of Operational Research* 46(1): 48–60.

Erkut, E. and S. Neuman. (1991).Comparison of four models for dispersing facilities. *INFOR Canadian Journal of Operational Research and Information Processing* 29: 68–86.

Festa, P. (2013). A biased random-key genetic algorithm for data clustering, in: SI:BIOCOMP, Math. Biosci.v. 245, no 1, p.76–85.

Gallego, M., A. Duarte, M. Laguna, and R. Martí. (2009). Hybrid heuristics for the maximum diversity problem". *Computational Optimization and Applications* 44(3): 411-426.

Glover, F. Hersh, G. e Millan, C. (1977). Selecting subsets of maximum diversity, MS/IS Report no 77-9, University of Colorado at Boulder.

Glover, F., C.C. Kuo, and K.S. Dhir. (1998). "Heuristic algorithms for the maximum diversity problem". *Journal of Information and Optimization Sciences*, 19(1): 109–132.

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

Handler, G. Y. e Mirchandani, P. B. (1979). Location on Networks: Theory and Algorithms. MIT Press, Cambridge, Mass.

Katayama, K. and H. Narihisa. (2005). An evolutionary approach for the maximum diversity problem. In W. Hart, N. Krasnogor and J. E. Smith (eds), *Recent advances in memetic algorithms*, volume 166, pp 31–47. Springer, Berlin.

Kincaid, R.K. (1992). Good solutions to discrete noxious location problems via metaheurísticas. *Annals of Operations Research* 40(1): 265–281.

Kuby, M. J. (1987). Programming models for facility dispersion: The p-dispersion and maximum dispersion problem. *Geographical Analysis*, 19(4):315-329.

Kuo, C.C., F. Glover and K.S. Dhir. (1993). "Analyzing and Modeling the Maximum Diversity Problem by Zero-One programming. *Decision Sciences* 24(6): 1171–1185.

Martí, R., Gallego, M., Duarte, A. e Pardo, E. G. (2013) Heuristics and metaheuristics for the maximum diversity problem. Journal of Heuristics 19(4): 591-615.

Moon, I.D. e Chaudhry, S.S. (1984). An analysis of network location problems with distance constraints. *Management Science* 30(3): 290-307.

Moro, M.A. (2017). Metaheurística GRASP e BRKGA Aplicada ao Problema da Diversidade Máxima. Dissertação de Mestrado. Campinas, SP.

Oliveira, R.M., Chaves, A.A. e Lorena, L.A.N. (2017). A comparison of two hybrid methods for constrained clustering problems. *Applied Soft Computing*, 54:256–266.

Palubeckis, G. (2007). Iterated tabu search for the maximum diversity problem. *Applied Mathematics and Computation* 189: 371-383.

Prokopyev, O.A, Kong, N. e Martinez-Torres, D.L (2009). The equitable dispersion problem. *European Journal of Operational Research* 197(1):59-67.

Resende, M.G.C. (2013). Biased random-key genetic algorithms. Talk given at XLV Symposium of the Brazilian Operational Research Society (XLV SBPO) Natal, RN.

Santos, L.F., M.H. Ribeiro, A. Plastino, and S.L. Martins. (2005). A hybrid grasp with data mining for the maximum diversity problem. In Maria J. Blesa, Christian Blum, Andrea Roli, and Michael Sampels (eds.), *Hybrid Metaheuristics*, volume 3636 of *Lecture Notes in Computer Science*, pp 116–127. Springer.

Silva, G.C., Ochi, L.S. e Martins, S.L. (2004). Experimental Comparison of Greedy Randomized Adaptive Search Procedures for the Maximum Diversity Problem. In *Experimental and Efficient Algorithms*, volume 3059 of *Lecture Notes in Computer Science*, p. 498–512. Springer Berlin / Heidelberg.

Silva, G.C., Ochi, L.S. e Martins, S.L. (2006). Proposta e avaliação de heurísticas grasp para o problema da diversidade máxima. *Pesquisa Operacional*, 26(2):321-360.

Silva, G.C., M.R.Q. Andrade, L.S. Ochi, S.L. Martins e A. Plastino. (2007). New heuristics for the maximum diversity problem. *Journal of Heuristics* 13(4): 315–336.

Silva, R.M.A e Resende, M.G.C (2013). Algoritmo genético de chaves aleatórias viciadas para problemas de otimização global com restrições de caixa sujeitas a restrições não-lineares. Anais do XLV Simpósio Brasileiro de Pesquisa Operacional, p. 3680-3678, Natal, RN.

Spears, W.M., De Jong, K.A. (1991). On the virtues of parameterized uniform crossover. *In*: *Proceedings* of the Fourth International Conference on Genetic Algorithms, p. 230-236