ALGORITMOS HEURÍSTICOS PARA O PROBLEMA DA DIVERSIDADE MÁXIMA

Ivan Luis Duarte
Geiza Cristina da Silva (*)
Tatiana Alves Costa
Universidade Federal de Ouro Preto
Departamento de Ciências Exatas e Aplicadas – Rua 37 nº. 115 – Loanda
35930-970 João Monlevade – Minas Gerais – Brasil
ivanluisduarte@yahoo.com.br
{geiza, tatiana}@decea.ufop.br

RESUMO

O Problema da Diversidade Máxima (PDM) é um problema da área de Otimização Combinatória que tem por objetivo selecionar um número pré-estabelecido de elementos de um dado conjunto de maneira tal que a soma das diversidades entre os elementos selecionados seja a maior possível. Por pertencer à classe de problemas NP-Difícil, o desenvolvimento de novos métodos heurísticos na solução aproximada deste problema em tempo computacional aceitável, torna-se atrativo. Neste trabalho são propostos algoritmos heurísticos baseados nos conceitos da metaheurística *Iterated Local Search*. Resultados computacionais mostram que os algoritmos propostos apresentam um desempenho médio superior quando comparados com as melhores heurísticas da literatura.

PALAVARAS CHAVE. Problema da Diversidade Máxima, Iterated Local Search, metaheurísticas.

Áreas Principais: MH – Metaheurística, OC - Otimização Combinatória.

ABSTRACT

The Maximum Diversity Problem (MDP) is a problem of the Combinatorial Optimization area which asks to select a specified number of elements from a given set so that the sum of diversity between the selected elements is as large as possible. MDP belongs to the NP-hard class becoming attractive to the development of new heuristics to solve the problem in acceptable computational time. In this work heuristics algorithms based on concepts of the Iterated Local Search metaheuristic are proposed. Computational results show that the proposed algorithm produces an average performance better than well known heuristics from the literature.

KEYWORDS. Maximum Diversity Problem, Iterated Local Search, metaheuristics.

Main Areas: MH – Metaheuristic, OC – Combinatorial Optimization.

1. Introdução

O Problema da Diversidade Máxima (PDM) consiste em, dado um conjunto N com n elementos, selecionar um subconjunto $M \subset N$ de forma tal que os elementos de M possuam a maior diversidade possível entre eles, sendo a diversidade expressa por uma função d_{ij}, definida sobre todo $i < j \in M$.

Existem dois critérios através dos quais o problema pode ser descrito. Na abordagem MAXSUM o objetivo é maximizar a soma das distâncias entre os elementos de M, ou seja, $Max\, z(M) = \sum_{i,j \in M, i<j} d(i,j)$. A abordagem MAXMIM, ao contrário, considera a pior diversidade. O objetivo, neste caso, é $Max\, z(M) = \min\{d_{ij}\}, i < j, j \in M$.

Kuo et al. (1993) provaram que MAXSUM é NP-Difícil, utilizando uma redução ao problema Clique em grafo. Ghosh (1996) provou que assim como MAXSUM, MAXMIM é NP-Difícil, a partir de uma redução ao problema de recobrimento de vértices.

O estudo de complexidade do problema mostra claramente a limitação do uso exclusivo de métodos exatos para a sua solução. Neste trabalho é proposto um algoritmo de construção aleatório e guloso, assim como três heurísticas de busca local. Estes algoritmos são aplicados aos conceitos da metaheurística ILS (*Iterated Local Search*) para instâncias do problema abordando o critério MAXSUM. Os testes computacionais realizados mostraram que os algoritmos propostos apresentam um desempenho médio superior quando comparados com as melhores heurísticas da literatura.

O restante deste artigo está organizado conforme segue. Na seção 2 o Problema da Diversidade Máxima é apresentado. Na seção 3 é feita uma revisão da literatura assim e dois algoritmos da literatura conhecidos pelos bons resultados alcançados para o problema são descritos. Na seção 4 são discutidas as propostas de algoritmos deste trabalho. A seção 5 apresenta os resultados computacionais e finalmente, a seção 6 conclui o trabalho.

2. O Problema da Diversidade Máxima

O objetivo do Problema da Diversidade Máxima (PDM) é selecionar, a partir de um conjunto N de cardinalidade n, um subconjunto $M \subset N$ com m elementos que possuam a maior diversidade entre si. Uma medida de diversidade entre dois elementos pode ser representada pela distância entre eles. A distância, neste caso, é calculada com base no conjunto de atributos (componentes) dos elementos de N.

Desta forma, para definir o problema, considera-se o conjunto de índices $N = \{1, 2, ..., n\}$, e:

- t, o número de atributos que caracterizam os elementos de N;
- a_{ik}, o valor do atributo k, $k = \{1,..., t\}$ do elemento i, $i \in N$.
- $d(i,j)$, o índice de diversidade entre dois elementos distintos, onde i e j possuem os respectivos valores de atributos $(a_{i1}, a_{i2}, ..., a_{it})$ e $(a_{j1}, a_{j2}, ..., a_{jt})$.

O índice de diversidade representa o grau de diferença existente entre os atributos dos elementos i e j, ou seja, a distância entre eles, que deve ser calculada usando-se uma dentre as métricas de distâncias existentes. De acordo com a área de aplicação, uma métrica pode refletir melhor o problema que outra (Kochenberg e Glover, 1999).

Seja D $[n \times n]$ uma matriz simétrica que armazena os índices de diversidade entre cada um dos elementos de N, preenchida da seguinte maneira: $d(i, j) = d(j, i)$ e $d(i, i) = 0$, $\forall (i, j) \in N$.

O valor da diversidade de M é determinado pela fórmula $div(M) = \sum_{i,j \in M, i<j} d(i,j)$.

O Problema da Diversidade Máxima pode ser aplicado nas mais diversas áreas relacionadas ao campo da pesquisa operacional como na administração de recursos humanos (Dhir et al., 1994), (Weitz e Lakshminarayanan, 1998), biologia (Agrafiotis, 1997) e preservação ecológica (Maser, 1999), (Unkel, 1985).

O trabalho de Kuo et al. (1993), além da definição do problema, apresentou os dois modelos matemáticos para o PDM mostrados a seguir.

O PDM pode ser representado como um Problema de Programação Inteira considerando-se $x_i, \forall_i \in N$, variável binária igual a 1, se e somente se o elemento i estiver na solução e 0, caso contrário, e $d(i, j)$ representando a diversidade entre i e j, $i, j \in N$.

(f_1) Maximizar

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d(i, j) x_i x_j \tag{1}$$

sujeito a:

$$\sum_{i=1}^{n} x_i = m, \tag{2}$$

$$x_i \in \{0,1\}, \forall i = 1,..., n. \tag{3}$$

A formulação f_1, descrita pelas equações 1-3 é um modelo não linear devido a sua função objetivo quadrática (1) e tem como meta maximizar a soma das diversidades de m elementos de N. A igualdade (2) exige que exatamente m elementos façam parte de uma solução viável e (3) indica que as variáveis do problema devem ser binárias.

Uma versão linearizada de f_1 descreve o PDM como um Problema de Programação Linear Inteira. Para isto, além das notações da formulação anterior, devem-se considerar as variáveis de decisão y_{ij} que se tornam iguais a 1 se i e j fizerem parte da solução e 0, caso contrário, $\forall_{i,j} \in N$, $i < j$ e $Q = \{(i, j): i, j \in \mathbf{N}, i < j\}$.

(f_2) Maximizar

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d(i, j) y_{ij} \tag{4}$$

sujeito a:

$$\sum_{i=1}^{n} x_i = m, \tag{5}$$

$$x_i + x_j - y_{ij} \le 1, \forall (i, j) \in Q, \tag{6}$$

$$-x_i + y_{ij} \le 0, \forall (i, j) \in Q, \tag{7}$$

$$-x_j + y_{ij} \le 0, \forall (i, j) \in Q, \tag{8}$$

$$y_{ij} \in \{0,1\}, \forall (i, j) \in Q, \tag{9}$$

$$x_i \in \{0,1\}, \forall i \in N. \tag{10}$$

Na formulação f_2, a função objetivo (4) deve maximizar a soma das diversidades de m elementos de N. A igualdade (5) exige que exatamente m elementos pertençam a uma solução viável. Nas restrições (6) pode-se observar que a variável binária y_{ij} assume valor 1 sempre que x_i e x_j receberem tal valor e y_{ij} assume valor 0 se x_i e x_j forem ambas iguais a 0 ou se x_i ou x_j (somente uma delas) for igual a 1. As restrições (7) exigem que no caso em que x_i pertencer à solução então deverá existir algum j tal que y_{ij} = 1. Da mesma forma que em (7), as restrições (8) exigem que se x_j pertencer à solução então deverá existir algum i tal que y_{ij} = 1. As restrições (9) e (10) representam as variáveis binárias do problema, sendo y_{ij} = 1 se x_i e x_j estiverem na solução e 0 caso contrário e cada x_i assumindo valor 1 se pertencer à solução e 0, caso contrário.

Outras formulações matemáticas para o PDM foram propostas em Glover (1995) para aplicações específicas na área de meio ambiente.

Além de Kuo et al. (1993), Cutler e Klastori (1997), Bhadury et al. (2000) e Martí et al. (2007) apresentam abordagens exatas para tratar o problema.

A seção 3 descreve os principais trabalhos utilizando metodologias heurísticas para o PDM.

3. Abordagens Anteriores

Em Ghosh (1996) foi proposto um algoritmo GRASP para o PDM e para comprovar a qualidade da heurística proposta foi implementado um algoritmo exato. Os resultados computacionais mostraram que o GRASP proposto alcança uma solução ótima ou uma solução muito próxima da ótima, mostrando desta forma sua eficiência ao menos para instâncias de pequeno porte ($n \leq 40$).

Agrafiotis (1997) explorou o problema de selecionar subconjuntos de componentes químicos. No trabalho introduziu duas medidas para quantificar a diversidade e apresentou uma solução para o problema baseada em *simulated annealing*. O método foi testado usando bases de dados formadas a partir de métodos estatísticos. Os resultados foram visualizados a partir do algoritmo de mapeamento não-linear de Sammon [Sa69]. O artigo não fornece detalhes dos testes computacionais, apenas cita que o método encontra resultados satisfatórios.

Em Cutler e Klastori (1997) foram apresentadas duas heurísticas para o PDM, uma relaxação lagrangeana e uma heurística gulosa. Para avaliar os resultados obtidos foi implementado um algoritmo *branch and bound* e, desta forma, os autores concluíram que a heurística gulosa apresentou bons resultados para um conjunto de testes com número de elementos pequeno ($n \leq 40$).

Weitz e Lakshminarayanan (1998) apresentaram cinco heurísticas para o PDM. Este trabalho foi realizado para resolver uma aplicação prática que é criar grupos de projetos em uma classe de estudantes, de forma tal que esses estudantes sejam inseridos em ambientes heterogêneos, baseado em critérios como, por exemplo, nacionalidade, idade e formação. As heurísticas foram testadas somente para uma instância composto de dados reais e foi implementado também um algoritmo exato. Os testes apontaram à heurística LCW (Método *Lotfi-Cerveny-Weitz*) como sendo a melhor para a solução do problema.

Em Glover et al. (1998) foram propostas quatro heurísticas sendo duas delas, construtivas e as outras destrutivas onde, a partir de uma solução heurística com e elementos ocorre uma eliminação progressiva até que se tenha uma solução viável com m elementos. A eficiência dessas heurísticas foi testada através de comparação com os resultados de um método exato e para isso foram criadas diversas instâncias de populações com até 30 elementos. Foi citado pelos autores que as heurísticas propostas obtiveram em média resultados próximos aos do algoritmo exato (a no máximo 2% do ótimo), com um tempo de computação muito menor.

O trabalho de Kochenberger e Glover (1999) desenvolveu uma heurística busca tabu para encontrar a máxima diversidade. Segundo os autores, a heurística foi testada gerando aleatoriamente instâncias de problemas com populações de tamanho que variam de $n = 100$ a $n = 1000$ elementos onde se desejava obter subconjuntos de m elementos mais diversos com tamanho entre 10% a 30% de n. Os autores citam que testes computacionais foram realizados, mas os resultados não foram comparados com qualquer outro método da literatura.

Silva et al. (2004) apresentaram três algoritmos construtivos e duas heurísticas de busca local utilizados para implementar várias heurísticas GRASP para o PDM. Diversos experimentos computacionais foram realizados com instâncias de médio porte propostas no trabalho e mostraram que seus procedimentos superaram métodos apresentados anteriormente.

Silva et al (2007) incorporaram à abordagem anterior um módulo de reconexão por caminhos. Os testes realizados comprovaram a eficiência do método híbrido proposto.

Aringhieri et al. (2008) propuseram um algoritmo busca tabu para o problema. A busca tabu proposta utiliza os conceitos de lista tabu dinâmica e memória secundária, esta última, armazena as segundas melhores soluções encontradas, mas não utilizadas em cada iteração da busca tabu e é utilizada para reiniciar a busca em casos específicos. Os autores citam que os resultados alcançados puderam equiparar ou superar os outros métodos.

Em Palubeckis (2008) foi desenvolvido um algoritmo *Iterated Tabu Search* (ITS). Foi apresentado também um algoritmo *steepest ascent*. Para comparação foram considerados os dois métodos propostos e os trabalhos de Silva et al. (2007) e Martí (2007). Instâncias de maiores portes foram utilizadas e os resultados reportados demonstram boa eficiência do algoritmo ITS.

Os trabalhos de Duarte e Martí (2007) e Gallego et al. (*to appear*) têm sido considerados referências de algoritmos heurísticos, obtendo os melhores resultados da literatura para o PDM. Por este motivo, os algoritmos apresentados nestes trabalhos, utilizados para efeito de comparação com a metodologia aqui proposta, são descritos a seguir.

3.1. Algoritmo Tabu_D2

Proposto em Duarte e Márti (2007), implementa uma variante da heurística destrutiva `D2` de Glover et al. (1998) para a construção de uma solução. Seja *Sel* o conjunto de elementos selecionados para compor a solução. No início do procedimento, este conjunto possui todos os elementos do conjunto *N*, isto é: *Sel* = *N*. A cada passo, um elemento i^* de menor $D(i^*)$ é retirado da solução, até que se obtenha *Sel* = *M*. O cálculo de $D(i^*)$ é mostrado na Equação 11.

$$D(i^*) = \min_{i \in Sel}[D(i) - \beta \times R(j) \times \frac{freq[i]}{\max\_freq} + \delta \times R(j) \times \frac{qual[i]}{\max\_qual}. \quad (11)$$

Nesta equação, $D(i)$ é o somatório das distâncias dos pares i e j selecionados para compor as soluções prévias, isto é, $D(i) = \sum_j d_{ij} x_i x_j$; $freq[i]$ corresponde à freqüência na qual o elemento *i* aparece em soluções prévias e $qual[i]$ é a média da qualidade (relacionada à função objetivo) das soluções anteriores que incluem o elemento *i*. Os valores *max_freq* e *max_qual* são os máximos de $freq[i]$ e $qual[i]$. Os fatores de penalidade β e δ foram definidos pelos autores como 0,1 e 0,0001, respectivamente. Além disso, $R(j) = \max_{i \in Sel}(D(i)) - \min_{i \in Sel}(D(i))$.

Como mecanismo de memória de curto prazo da busca tabu foi implementado o método `LS_TS`. A cada iteração, um elemento *i* pertencente à solução é selecionado aleatoriamente. A probabilidade da seleção do elemento *i* é inversamente proporcional a $D(i)$. O conjunto $N - M$ de elementos fora da solução é investigado, obtendo-se um elemento *j* que proporciona o primeiro movimento de melhora na troca do elemento *i* por *j*. Se nenhum movimento de melhora é obtido, o melhor movimento de piora é escolhido. A troca é realizada e os elementos *i* e *j* são incluídos em uma lista tabu. Durante um número de iteração, estes elementos são considerados tabu e, portanto, são proibidos de serem usados nas trocas. O método termina após um número consecutivo de movimentos sem melhora. É citado que o número de iterações no qual um elemento permanece tabu é $0{,}28 \times m$ para elementos que estão sendo inseridos na solução e de $0{,}028 \times m$ para os elementos que estão deixando a solução. O número de iterações sem melhora é dado por $0{,}1 \times n$.

3.2. Algoritmo SS

Em (Gallego, *to appear*) é proposta uma implementação da metaheurística *scatter search*, cujos conceitos são detalhados em Laguna e Martí (2003), denominado algoritmo `SS`. O método consiste em explorar um espaço de soluções que fazem parte de um conjunto soluções de referência, denominado *RefSet*. Este conjunto armazena boas soluções que foram encontradas durante o processo de busca. Os passos do algoritmo `SS` são:

1. Um conjunto *P* com *tam* soluções diversas é construído utilizando `Tabu_D2`.
2. O procedimento `LS_TS` é aplicado como método de melhoria a todas as soluções em *P*.
3. O conjunto inicial de soluções de referência *RefSet* consiste das *b* = 12 soluções em *P*, das quais *q* = 30% são escolhidas de acordo com a qualidade das soluções e $(1 - q)$% são escolhidas de acordo com a diversidade das mesmas. Uma medida

de distância *dist* é aplicada para definir a diversidade de uma solução em relação ao conjunto de *RefSet*. A distância entre uma solução *t* e as contidas em *RefSet*, é dada por: $dist(t, RefSet) = bm - \sum_{r=1}^{b} \sum_{i \in M} x_i^r$. Esta fórmula conta o número de vezes que cada elemento da solução *t* aparece nas soluções de *RefSet* e subtrai este valor da distância máxima possível (*bm*).

4. O método de geração de subconjuntos considera cada par de soluções de *RefSet* como subconjuntos de soluções de tamanho 2.
5. O método de combinação das soluções utiliza `Tabu_D2` para a união de elementos das soluções contidas nos subconjuntos definidos no Passo 4.
6. Novamente, o método de melhoria é aplicado a todas as soluções geradas no Passo 5.
7. *RefSet* é atualizado se for obtida uma solução melhor do que as já armazenadas através dos Passos 5 e/ou 6. Se *RefSet* tiver sido atualizado, o procedimento retorna ao Passo 4, caso contrário, é encerrado.

4. *Iterated Local Search*

A metaheurística *Iterated Local Search* (ILS) é um procedimento iterativo no qual buscas locais são aplicadas a novas soluções de partida, obtidas através de perturbações em soluções de ótimos locais.

O método funciona da seguinte maneira (Lourenço et al, 2002): Dada uma solução corrente *s**, uma perturbação é aplicada levando a uma solução *s*'. Em seguida, *s*' é explorada através de um procedimento de busca local através do qual é obtida a solução *s**', que corresponde a um ótimo local. Um critério de aceitação é invocado para decidir, entre *s** e *s**', qual delas deve se tornar corrente na próxima iteração. A Figura 1 apresenta o pseudocódigo do algoritmo ILS.

```
Procedimento ILS
      s0 = GeraSolucaoInicial
      s* = BuscaLocal (s0)
      Faça
        s' = Perturbacao(s*, história)
        s*' = BuscaLocal(s')
        s* = CriterioAceitacao (s*, s*', história)
      Enquanto critério de parada não for satisfeito
Fim-Procedimento.
```

Figura 1. Pseudo-código da metaheurística ILS

Existem quatro componentes fundamentais do método a serem considerados, representados no pseudo-código por meio das funções: i) `GeraSolucaoInicial`, ii) `BuscaLocal`, iii) `Perturbação` e iv) `CriterioAceitacao`. Mais detalhadamente, i) retorna uma solução inicial. Esta pode ser gerada aleatoriamente ou através de algum procedimento guloso; ii) retorna um ótimo local; iii) perturba a solução através de um ou mais movimentos na solução corrente; e iv) decide qual solução será considerada na próxima iteração do algoritmo.

Uma perturbação não deve ser demasiadamente pequena para que o procedimento não retorne frequentemente à solução corrente e poucas novas soluções sejam exploradas, assim como, não deve ser grande demais, pois neste caso o método recairia em um algoritmo de múltiplos reinícios aleatórios.

O mecanismo de perturbação junto com o de busca local define a possível transição entre a solução corrente s^* para a solução vizinha s^{*}'. O critério de aceitação determina se *s**' é aceito ou não como a nova solução corrente. Pode servir como um procedimento de intensificação se for definido de forma que somente melhores soluções são aceitas, isto é, para um problema onde o objetivo é a maximização tem-se: `CriterioAceitacao` retorna $s^{*'}$ se $Z(s^{*'}) > Z(s^{*})$ e; s^{*}, caso contrário, onde Z é a função objetivo do problema.

Neste trabalho, as quatro componentes do ILS foram definidas da seguinte maneira:

1. Construção da solução inicial: Uma solução inicial é construída da seguinte maneira: Para cada elemento $i \in$ N são selecionados os m elementos $j \in N - M$ que possuem os maiores índices de diversidade em relação ao i. As diversidades destes elementos são somadas totalizando s_i. A partir daí, os m elementos com maiores valores de s_i são escolhidos para compor a solução.
2. Mecanismo de Perturbação: Foram utilizados sete níveis de perturbação, sendo os seis primeiros definidos com base em um valor percentual que indica o número de trocas que são realizadas entre elementos dentro e fora da solução. No primeiro nível são escolhidos, aleatoriamente, 2% dos elementos pertencentes à solução e, cada um desses, é trocado com um outro elemento não pertencente à mesma, também selecionado de forma aleatória. Nos cinco níveis seguintes, os percentuais adotados são 10, 40, 50, 70 e 80%, respectivamente. A utilização do sétimo nível de perturbação é condicionada aos casos nos quais todos os níveis de perturbação definidos anteriormente resultarem em um número de trocas entre mais de dois elementos. Nestes casos, a perturbação de nível sete é utilizada para alterar levemente a solução. A cada iteração do método um nível de perturbação é escolhido aleatoriamente.
3. Algoritmos de busca local: Na fase de busca local são utilizados três algoritmos baseados nos conceitos clássicos apresentados em Sousa (2005) denominados como melhor vizinho, primeiro de melhora e busca randômica.
    a. O primeiro é uma implementação do algoritmo de busca local proposto por Ghosh (1996). Cada elemento i pertencente à solução é trocado por cada elemento j fora desta. A cada troca é associada uma melhoria $\Delta z(i, j)$ calculada por: $\Delta z(i,j) = \sum_{u \in M - \{i\}} d(j,u) + d(i,u)$. A cada iteração, todas as possíveis trocas são testadas e a solução obtida da troca dos elementos i, j de maior Δz é considerada corrente para a próxima iteração. A busca prossegue até que, em uma iteração, todas as trocas efetuadas retornem $\Delta z(i, j) < 0$.
    b. No segundo algoritmo de busca local é calculada a melhoria $\Delta z(i, j)$ referente às trocas e a solução resultante da primeira troca na qual $\Delta z(i, j) > 0$ torna-se a solução corrente da próxima iteração. A busca prossegue até que nenhuma solução de melhora possa ser obtida.
    c. No terceiro algoritmo, a implementação da busca local randômica consiste em escolher aleatoriamente dois elementos $i \in M$ e $j \in N - M$ e realizar a troca entre eles. Se o valor de $\Delta z(i, j)$ associado a esta troca for maior do que 0 a solução produzida torna-se corrente na próxima iteração. Caso contrário, um novo par de elementos é selecionado aleatoriamente. A busca prossegue até que um determinado critério de parada seja alcançado. Neste trabalho os critérios de parada adotados foram número de iterações sem melhora e tempo de processamento.
4. Critério de aceitação: De acordo com o critério de aceitação adotado, uma solução $s^{*'}$ passa ser corrente caso melhore o valor da função objetivo da solução s^{*}, ou seja, se $Z(s^{*'}) > Z(s^{*})$.

4. Resultados computacionais

Com base na metodologia descrita na Seção 3 foram propostas três diferentes versões de heurísticas ILS, que se diferenciam pelo procedimento de busca local adotado, conforme Tabela 1.

Algoritmo	Busca Local
ILS1	Melhor Vizinho (Ghosh, 1996)
ILS2	Primeiro de Melhora
ILS3	Randômica

Tabela 1. Algoritmos ILS propostos

Os algoritmos propostos foram implementados usando linguagem C e compilados na versão 3.3.3 do GCC. Os testes foram realizados em um computador Intel Pentium IV 3 Ghz com 1 Gb de memória RAM. Cada heurística foi executada 10 vezes com o tempo máximo de processamento de 30 segundos para cada execução.

Para efeito de comparação foi utilizado o conjunto de instâncias proposto em Silva et. al. (2004) e os algoritmos descritos na seção 3. No conjunto de instâncias utilizado, o número de variáveis $n \in \{100, 200, 300, 400, 500\}$ e o parâmetro $m \in \{0.1n, 0.2n, 0.3n, 0.4n\}$. Cada valor de diversidade $d(i, j)$ foi gerado aleatoriamente dentro do intervalo [0, 9].

A Tabela 2 apresenta os valores de diversidades obtidos pelos algoritmos comparados. A coluna 1 apresenta as instâncias e as colunas 2 e 3 mostram os valores de *n* e *m* de cada uma destas. Na coluna 4 encontra-se o melhor valor conhecido da literatura para cada instância (Martí, 2008). As colunas 5 e 6 apresentam, respectivamente, os valores reportados em Gallego (*to appear*) para os procedimentos **TabuD2** e **SS**. Estes, correspondem ao melhor valor obtido em 10 execuções de 30 segundos com os algoritmos codificados pelos autores em linguagem Java, executados com Java *Runtime Environment* 1.5 e testados em um computador Pentium IV 3 GHZ com 3 Gb de memória RAM. Nas colunas 7 a 9 são apresentados os melhores valores de diversidades obtidos, em 10 execuções, pelos algoritmos propostos **ILS1**, **ILS2**, **ILS3**.

				Literatura		Propostos		
Instâncias	**n**	**m**	**Melhor Valor**	**TabuD2**	**SS**	**ILS1**	**ILS2**	**ILS3**
Silva.1	*100*	*10*	**333**	**333**	**333**	**333**	**333**	**333**
Silva.2	*100*	*20*	**1195**	**1195**	**1195**	**1195**	**1195**	**1195**
Silva.3	*100*	*30*	**2457**	**2457**	**2457**	**2457**	**2457**	**2457**
Silva.4	*100*	*40*	**4142**	4139	**4142**	**4142**	**4142**	**4142**
Silva.5	*200*	*20*	**1247**	1245	**1247**	**1247**	**1247**	**1247**
Silva.6	*200*	*40*	**4450**	4448	**4450**	**4450**	**4450**	**4450**
Silva.7	*200*	*60*	**9437**	9426	**9437**	**9437**	**9437**	**9437**
Silva.8	*200*	*80*	**16225**	**16225**	**16225**	**16225**	**16225**	**16225**
Silva.9	*300*	*30*	**2694**	**2694**	**2694**	**2694**	**2694**	**2694**
Silva.10	*300*	*60*	**9689**	9683	**9689**	**9689**	**9689**	**9689**
Silva.11	*300*	*90*	**20743**	20715	**20743**	**20743**	**20743**	**20743**
Silva.12	*300*	*120*	**35881**	35871	**35881**	35875	**35881**	**35881**
Silva.13	*400*	*40*	**4658**	4651	**4658**	**4658**	**4658**	**4658**
Silva.14	*400*	*80*	16950	16929	16950	**16956**	**16956**	**16956**
Silva.15	*400*	*120*	**36317**	36288	**36317**	36306	36310	**36317**
Silva.16	*400*	*160*	**62487**	62429	**62487**	62460	62446	**62487**
Silva.17	*500*	*50*	**7141**	7117	**7141**	**7141**	7136	7133
Silva.18	*500*	*100*	**26258**	26229	**26258**	26239	26242	**26258**
Silva.19	*500*	*150*	**56572**	56547	**56572**	**56572**	**56572**	**56572**
Silva.20	*500*	*200*	**97344**	97269	**97344**	97317	**97344**	97335

Tabela 2. Comparação entre os melhores resultados reportados em Martí (2008) e algoritmos ILS propostos

A Tabela 2 demonstra que os algoritmos propostos sempre são capazes de alcançar a melhor solução em cada instância e, para a instância `Silva.14` o melhor resultado até então conhecido na literatura pôde ser melhorado.

A Tabela 3 faz uma comparação entre a diferença percentual do melhor resultado obtido pelos algoritmos propostos e os algoritmos da literatura. Diferenças percentuais iguais a zero são representadas pelo símbolo (-).

	Literatura		**Propostos**		
Instâncias	**TabuD2**	**SS**	**ILS1**	**ILS2**	**ILS3**
Silva.1	-	-	-	-	-
Silva.2	-	-	-	-	-
Silva.3	-	-	-	-	-
Silva.4	0,00072	-	-	-	-
Silva.5	0,00160	-	-	-	-
Silva.6	0,00045	-	-	-	-
Silva.7	0,00117	-	-	-	-
Silva.8	-	-	-	-	-
Silva.9	-	-	-	-	-
Silva.10	0,00062	-	-	-	-
Silva.11	0,00135	-	-	-	-
Silva.12	0,00028	-	0,00017	-	-
Silva.13	0,00150	-	-	-	-
Silva.14	0,00159	0,00035	-	-	-
Silva.15	0,00080	-	0,00030	0,00019	-
Silva.16	0,00093	-	0,00043	0,00066	-
Silva.17	0,00336	-	-	0,00070	0,00112
Silva.18	0,00110	-	0,00072	0,00061	-
Silva.19	0,00044	-	-	-	-
Silva.20	0,00077	-	0,00028	-	0,00009
Soma	0,01669	0,00035	0,00190	0,00216	0,00121

Tabela 3. Diferenças percentuais entre algoritmos da literatura e propostos

A diferença percentual foi obtida calculando-se a diferença entre o melhor valor conhecido para cada instância e os resultados obtidos pelas heurísticas. Este valor é dividido pelo melhor valor.

Pode-se notar que, em se tratando dos algoritmos propostos, pelo menos um deles, é capaz de produzir o melhor valor em todas as instâncias.

Quanto aos algoritmos da literatura considerados, a diferença percentual média é de aproximadamente 2% no caso do algoritmo *TabuD2* e de menos de 0,04% para o algoritmo *SS*.

A robustez dos algoritmos propostos pode ser avaliada de acordo com a diferença percentual entre a melhor solução conhecida e a solução obtida por estes e, ainda, de acordo com a diferença percentual entre a melhor solução conhecida e a solução média alcançada em 10 execuções de cada algoritmo. Estes resultados são apresentados na Tabela 4.

Instâncias	ILS1		ILS2		ILS3	
	% melhor	% média	% melhor	% média	% melhor	% média
Silva.1	-	-	-	-	-	-
Silva.2	-	-	-	-	-	-
Silva.3	-	-	-	-	-	-
Silva.4	-	-	-	-	-	-
Silva.5	-	-	-	-	-	-
Silva.6	-	-	-	0,00022	-	0,00022
Silva.7	-	-	-	-	-	-
Silva.8	-	-	-	-	-	-
Silva.9	-	0,00186	-	0,00074	-	0,00037
Silva.10	-	0,00062	-	0,00052	-	0,00041
Silva.11	-	0,00092	-	0,00067	-	0,00039
Silva.12	0,00017	0,00020	-	-	-	0,00022
Silva.13	-	0,00172	-	0,00064	-	0,00086
Silva.14	-	0,00201	-	0,00083	-	0,00088
Silva.15	0,00030	0,00124	0,00019	0,00072	-	0,00058
Silva.16	0,00043	0,00093	0,00066	0,00078	-	0,00066
Silva.17	-	0,00364	0,00070	0,00224	0,00112	0,00168
Silva.18	0,00072	0,00107	0,00061	0,00065	-	0,00076
Silva.19	-	0,00016	-	0,00055	-	0,00030
Silva.20	0,00028	0,00111	-	0,00035	0,00009	0,00032
Média	0,00190	0,01545	0,00216	0,00891	**0,00121**	**0,00766**

Tabela 4. Diferenças percentuais entre as melhores soluções e soluções médias

Primeiramente, é feita uma análise em relação às melhores soluções (colunas 2, 4 e 6): o algoritmo ILS3 apresenta menores diferenças percentuais, estando em média a 0,1% da melhor solução conhecida. Os algoritmos ILS2 e ILS1 têm diferenças percentuais de aproximadamente 0,2%, ambos.

Quanto à diferença percentual média da melhor solução para as soluções médias obtidas, os métodos ILS1, ILS2 e ILS3 são de aproximadamente 1,5%, 0,9% e 0,8%, respectivamente.

Os resultados apresentados mostram que a heurística ILS produz resultados promissores para o problema.

Conclusões

Neste artigo foi descrita uma metodologia aplicada a obtenção de soluções heurísticas para o problema da diversidade máxima. Os procedimentos implementados no trabalho incluem uma heurística de construção aleatória e gulosa, três algoritmos de busca local baseados em conceitos clássicos e um esquema de perturbação de solução que combinados produzem uma abordagem baseada na metaheurística ILS. Os métodos propostos mostraram-se simples no que se à implementação. A despeito disto, os resultados alcançados demonstram que são superiores, na média, quando comparados aos métodos da literatura para um conjunto formado por 20 instâncias. Em particular, foi possível encontrar uma nova melhor solução para uma destas instâncias. Com isto, conclui-se que o método proposto é promissor no tratamento de instâncias do PDM. A fim de comprovar a robustez do método, nos próximos passos deste estudo, deseja-se utilizar outros conjuntos de instâncias disponíveis na literatura com um número maior de variáveis.

Agradecimentos

Esta pesquisa foi parcialmente suportada pelo CT-Info/MCT/CNPq Universal 15/2007 e PROPP/UFOP. Os autores agradecem também à FAPEMIG pelo apoio financeiro.

Referências

Agrafiotis, D. K. (1997), Stochastic algorithms for maximizing molecular diversity, *Journal Chem. Inf. Comput. Sci*., 37, 841-851.

Aringhieri, R., Cordone, R. e Melzani, Y. (2008), Tabu search versus GRASP for the maximum diversity problem, 4OR: *A quarterly Journal of Operations Research*, 6, 45-60.

Bhadury, J., Mighty, E. J., e Damar, H. (2000), Maximizing workforce diversity in project teams: a network flow approach, *Omega*, 28, 143-153.

Cutler, M. e Klastorin, T., The max diversity problem, Relatório Técnico, University of Washington, Department of Management Science, 1997.

Dhir, K., Glover, F. e Kuo, C.-C. (1994), Optimizing diversity for engineering management, *Proceedings of the IEEE International Engineering Management Conference*.

Duarte, A. E Martí, R. (2007)**,** Tabu search for the maximum diversity problem, *European Journal of Operational Research*, 178, 71-84.

Gallego, M., Duarte, A., Laguna, M. Martí, R. Hybrid heuristics for the maximum diversity problem, *Computational Optimization and Applications (To appear)*.

Glover, F., Kuo, C.-C. e Dhir, K. (1995), A discrete optimization model for preserving biological diversity. *Applied Mathematical Modeling*, 19, 696-701.

Glover, F., Kuo, C.-C. & Dhir, K. (1998), Heuristic algorithms for the maximum diversity problem. *Journal of Information & Optimization Science*, 109-132.

Ghosh, J.B. (1996), Computational aspects of the maximum diversity problem, *Operations Research Letters*, 19, 175–181.

Kochenberger, G. e Glover, F., Diversity data mining, *Relatório Técnico*, The University of Mississipi, 1999.

Kuo, C.-C., Glover, F. e Dhir, K. (1993), Analyzing and modeling the maximum diversity problem by zero-one programming. *Decision Science*, 24, 1171-1185.

Laguna, M. e Martí, R., *Scatter Search: Methodology and Implementations in C*, Kluwer Academic Publishers, Boston, 2003.

Lourenço, H. R., Martin, O. C. e Stützle, T., Iterated Local Search, *Handbook of Metaheuristics*, F. Glover and G. Kochenberger (Eds.), International Series in Operations Research & Management Science, vol. 57. Kluwer Academic Publishers, Norwell, MA., 321-353, 2002.

Maser, C, *Ecological diversity in sustainable development: the vital and forgotten dimension*. Saint Lucie Press, 1999.

Martí, R., Gallego, M. e Duarte, A., An Exact Method for the Maximum Diversity Problem, *Relatório Técnico*, Univesitat de València, 2007.

Martí, R., Rafael Martí - Departamento de Estadística e Investigación Operativa de la Universitat de València. Disponível em: < http://www.uv.es/rmarti/>. Acesso em: 24 de abril de 2008.

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

Resende, M. G., Martí, R., Gallego, M. e Duarte, A., GRASP and path relinking for the Max-min diversity problem, *Relatório Técnico*, AT&T Labs Research, TD-7BSN88, 2008.

Silva, G. C., Ochi, L. S., e Martins, S. L. (2004), Experimental Comparison of Greedy Randomized Adaptive Search Procedures for the Maximum Diversity Problem, *Lecture Notes on Computer Science*, 3059, 498-512.

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

Sousa, M. J. F. (2008). Inteligência Computacional para Otimização - Notas de aula. Ouro Preto: UFOP. Disponível em: <http://www.iceb.ufop.br/prof/marcone>. Acesso em: 22 abril de 2008.

Unkel, W. C., (1985), Natural diversity and national forest. *Natural Areas Journal*, 5, 8-13.

Weitz, R. e Lakshminarayanan, S. (1998), An empirical comparison of heuristic methods for creating maximally diverse group, *Journal of the Operational Research Society*, 49, 635-646.

(*) Autor para quem as correspondências devem ser encaminhadas.