ALGORITMO GENÉTICO HÍBRIDO PARA OTIMIZAÇÃO MULTIOBJETIVO: UMA APLICAÇÃO AO PROBLEMA DA MOCHILA

José Elias C. Arroyo
Vinícius A. Armentano
Departamento de Engenharia de Sistemas
Faculdade de Engenharia Elétrica e de Computação
Universidade Estadual de Campinas
Caixa postal 6101, Campinas-SP. 13083-970, Brasil
E-mail: {jclaudio, vinicius}@densis.fee.unicamp.br

Resumo

Este artigo propõe um algoritmo genético híbrido para gerar um conjunto de soluções dominantes de um problema de otimização combinatória multiobjetivo. O algoritmo utiliza fortemente o conceito de dominância de Pareto e contém estratégias de elitismo e preservação de diversidade na população, além de uma busca local multiobjetivo para explorar diferentes regiões em paralelo. A metaheurística é aplicada para resolver o problema da mochila com dois objetivos. A qualidade das soluções aproximadas é avaliada pela comparação com as soluções eficientes obtidas por um algoritmo Branch-and-Bound e por um algoritmo genético multiobjetivo propostos na literatura.

Palavras chave: Otimização combinatória multi-objetivo, Algoritmos genéticos, Problema da mochila

Abstract

In this article, we propose a hybrid genetic algorithm to generate a set of dominant solutions of a multiobjective combinatorial optimization problem. The algorithm uses strongly the concept of Pareto dominance and combines elitism, population diversity and a parallel multiobjective local search so as intensify the search in distinct regions. The metaheuristic approach is applied for the 0/1 knapsack multiobjective problem. The quality of the heuristic solutions is evaluated by a comparison with the efficient solutions given by a Branch-and-Bound algorithm and by a genetic algorithm, both from the literature.

Keywords: Multiobjective combinatorial optimization, Genetic algorithms, Knapsack problem

1. INTRODUÇÃO

Em problemas práticos de otimização, geralmente, é desejável otimizar mais de um critério de desempenho ao mesmo tempo. A otimização multiobjetivo procura otimizar várias funções objetivos conflitantes entre si. Nos problemas multiobjetivos, não existe uma única solução que otimize todos os objetivos simultaneamente, mas sim um conjunto de solucões, conhecido como conjunto eficiente ou Pareto-ótimo, tal que nenhuma solução é melhor que as soluções deste conjunto para todos os objetivos. A escolha de uma solução eficiente particular depende das características próprias do problema e é atribuída ao decisor (*decision maker*). A dificuldade em resolver problemas combinatórios multiobjetivos não somente é dada pela complexidade combinatorial, como no caso de problemas mono-objetivo, mas também pela busca de todas as soluções eficientes que crescem com o número de objetivos do problema. Metaheurísticas parecem ser a melhor alternativa para resolver problemas de otimização

multiobjetivos, pois são métodos flexíveis e eficientes. Os objetivos principais de toda metaheurística de otimização multiobjetivo são:

- Minimizar a distância do conjunto dominante encontrado ao conjunto Pareto-ótimo
- Obter uma boa distribuição das soluções no conjunto dominante gerado.

Métodos de busca local baseados em busca tabu (Hansen 1997, Gandibleux et al. 1997, Baykasoglu et al. 1999) e simulated annealing (Ulungu et al. 1998, Czyzak e Jaszkiewicz 1998) têm sido propostos para resolver problemas multiobjetivos. No entanto, a maioria esmagadora das publicações de metaheurísticas para problemas de otimização multiobjetivos são baseadas em algoritmos genéticos (Ehrgott e Gandibleux, 2000; Coello, 2000; Van Veldhuizen e Lamount, 2000a; Jones et al., 2002). Esta preferência se deve ao argumento questionável, que os algoritmos genéticos trabalham com uma população de soluções que podem conter informação sobre várias regiões do espaço de busca, e portanto, estes oferecem maiores possibilidades para encontrar o conjunto Pareto-ótimo ou uma aproximação dele. Depois do primeiro algoritmo genético multiobjetivo, (Vector Evaluated Genetic Algorithm - VEGA), proposto por Schaffer (1985), várias implementações de algoritmos genéticos para problemas multiobjetivos foram sugeridas (Horn et al., 1993; Fonseca e Fleming, 1993; Srinivas e Déb, 1995; Ishibuchi e Murata, 1998; Zitzler e Thiele, 1999; Jaskiewicz, 2002). De uma maneira geral, os algoritmos genéticos "puros" têm mostrado um desempenho inferior aos métodos baseados em busca em vizinhança, tais como busca tabu e simulated annealing (Glass et al., 1992; Ishibuchi et al., 1994). A hibridização dos algoritmos genéticos com outros métodos, em especial a busca em vizinhança, foi muito bem sucedida e estes algoritmos híbridos são extremamente competitivos com as demais metaheurísticas.

Neste artigo é proposto um algoritmo genético híbrido para resolver problemas de otimização multiobjetivo e obter um conjunto aproximado de soluções eficientes. A metaheurística é testada na resolução do problema da mochila 0/1 com dois objetivos. Este problema apresenta uma variedade de aplicações reais e recentemente tem sido objeto de estudo de muitos pesquisadores (Ulungu e Teghem, 1995; Visée et al., 1998; Zitzler e Thiele, 1999; Ben et al.,1999; Teghem et al., 2000; Gandibleaux e Freville 2000, Jaszkiewicz, 2002). A metaheurística híbrida proposta é comparada com o algoritmo Branch-and-Bound proposto por Ulungu e Teghem (1995) e com o algoritmo genético multiobjetivo desenvolvido por Zitzler e Thiele (1999).

2. OTIMIZAÇÃO MULTIOBJETIVO

Um problema geral de otimização multiobjetivo consiste em encontrar um vetor de variáveis de decisão (solução) que satisfaça restrições e otimize uma função vetorial cujos elementos representam as funções objetivos. Estas funções representam os critérios de otimalidade, que usualmente, são conflitantes. Portanto, o termo "otimizar" significa encontrar soluções com todos os valores dos objetivos aceitáveis para o decisor.

Formalmente, isto pode ser definido da seguinte maneira:

Minimizar (maximizar) = $( ) = (f_1( ) = z_1,..., f_r( ) = z_r),$ $\in$
Sujeito a $\in$

onde é o vetor decisão, é o vetor de objetivos, denota o *espaço de soluções factíveis* e = ( ) = { = ( ) | $\in$ } é a imagem de denominado *espaço objetivo factível*. Note que, a imagem de uma solução $= (x_1, x_2, ..., x_n) \in$ no espaço objetivo é um ponto $= (z_1, z_2, ..., z_r) =$ ( ), tal que $z_j = f_j( )$, $j = 1,..., r$ (r é o número de objetivos).

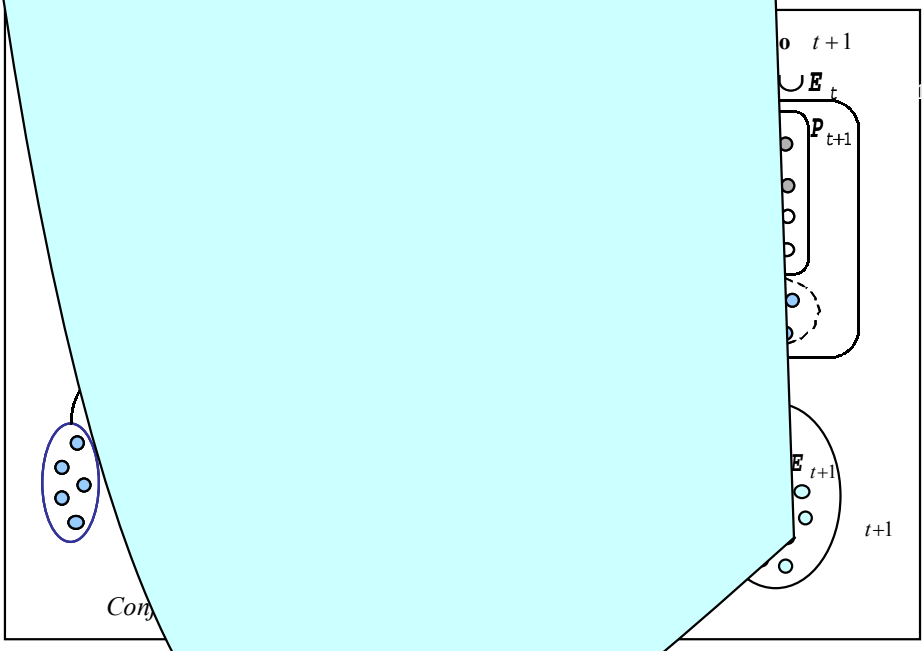
. Um ponto *domina* ' se $z_j = f_j( ) \leq z'_j = f_j(')$, $\forall j$ e $z_j < z'_j$ para pelo menos um j.

. Uma solução $*\in$ é *Pareto-ótimo* (ou *eficiente*) se não existe $\in$ tal que = ( ) domine * = (*). O conjunto de todas as soluções eficientes é denominado *conjunto eficiente* ou *conjunto Pareto-ótimo*.

3. DES JETIVO (AGHM)

- Elitis o corrente.
- Uso ntos da população e um meca cação da população.
- Uso d dominância de Pareto.

As enético desenvolvido em um trabalh e os passos e etapas do algoritmo g



GHM

A cada iteração t, constr de tamanho N e atualiza-se o conjunto de soluções dominantes $_t$ (melhor con ontrado até o momento). Na próxima iteração t+1, constrói-se uma população combinada $_{t+1} \cup _t$, onde $_t$ é o conjunto de N_{elite} soluções de elite selecionadas aleatoriamente a partir do conjunto de soluções dominantes $_t$.

Para atribuir um valor de fitness às soluções no AGHM, em primeiro lugar classificam-se as soluções da população combinada $_t \cup _{t-1}$. Adota-se uma técnica de classificação por dominância (*Nondominated Sorting technique*) inicialmente proposta por Goldberg (1989) e utilizada por Srinivas e Deb (1995) e Deb et al. (2000). Os valores de fitness são calculados usando a técnica de *crowding* proposta por Deb et al. (2000) que tenta preservar a diversidade da população $_t \cup _{t-1}$. Para cada ponto pertencente a uma sub-população, determina-se uma estimativa do número de pontos localizados em redor de . Esta estimativa é determinada considerando o vetor de objetivos das soluções e é dada pela *distância de crowding*. Usando as distâncias normalizadas de *crowding* de todos os pontos da população $_t \cup _{t-1}$, calcula-se o valor do fitness de cada solução.

Depois de atribuir um valor de fitness a cada solução da população combinada $_t \cup _{t-1}$, aplicam-se os operadores genéticos (recombinação e mutação) construindo uma população $'_t$ de soluções filhos ($|'_t| = N$).

A busca local multiobjetivo proposta é aplicada ao conjunto de soluções dominantes da população $'_t$. Ao final da busca, é construída uma população melhorada $_{t+1}$. O processo do algoritmo genético é repetido a partir desta população e considerando as soluções de elite da iteração anterior.

A seguir, é descrita uma nova estratégia de busca local multiobjetivo.

3.1. Busca Local Multiobjetivo

Nesta seção é proposto um método de busca local multiobjetivo (BLM) com o objetivo de melhorar um conjunto de soluções dominantes. O método é baseado no conceito de dominância de Pareto e explora em paralelo várias regiões do espaço de soluções. A busca começa com um conjunto de soluções dominantes e a partir de cada solução deste conjunto gera-se uma vizinhança ( ). Constrói-se então o próximo conjunto $'$ das soluções vizinhas não dominadas por onde $'$ é um subconjunto de $\bigcup_{\in}$ ( ). Caso $' \neq \emptyset$, o processo se repete a partir deste novo conjunto até que se verifique algum critério de parada. No Algoritmo 1 apresentado a seguir, descreve-se os passos da busca local multiobjetivo BLM.

1

Entrada: (um conjunto de soluções dominantes).
N_{max} (número máximo de caminhos a explorar).
Niter (número máximo de iterações)
Saída: $_{BL}$ (conjunto de soluções dominantes melhoradas).

0) *Inicialização*:
Faça $t = 1$ (contador de iterações) e $_{BL} =$.
1) *Redução do conjunto* :
Se | | > N_{max} então
Reduza o conjunto selecionando sómente N_{max} soluções.
Seja o conjunto das soluções não selecionadas.
2) *Construção de um conjunto de soluções vizinhas* $'$:
Faça $' = \emptyset$ (conjunto das soluções vizinhas dominantes).
Para cada $_k \in$ faça
Gere a vizinhança de $_k$: ($_k$).
Para cada $\in$ ($_k$) faça
Se $\notin$ $_{BL}$ ou não é dominado por $_{BL}$ então
Faça $'$ = soluções dominantes de ($' \cup \{ \}$).
3) *Atualização do conjunto* $_{BL}$ *com as novas soluções vizinhas em* $'$
$_{BL}$ = soluções dominantes de ($_{BL} \cup '$).
4) Faça $'$ = soluções dominantes de ($' \cup$).
Se ($' \neq \emptyset$ e $t \leq Niter$) então
Faça = $'$, $t = t + 1$ e volte ao passo 1.
Senão pare.

Note que o conjunto de soluções não selecionadas no passo 1 são consideradas no passo 4, quando é definido o novo conjunto de soluções dominantes $'$ a ser explorado. Note também que a cada iteração da busca local explora-se N_{max} soluções em paralelo. Para selecionar as soluções a serem exploradas, é proposto um procedimento (válido somente para problemas bi-objetivos) que divide o conjunto em N_{max} subconjuntos $_i$ ordenados de acordo com o primeiro objetivo. A partir de cada subconjunto $_i$ (i = 1,...,N_{max}) seleciona-se aleatoriamente uma solução. Este procedimento é descrito a seguir:

2

Entrada: $= \{ {}^{1}, {}^{2}, ..., {}^{|\ |} \}$
(um conjunto de $|\ | > N_{max}$ soluções dominantes tal que $f_1({}^{i}) < f_1({}^{i+1})$).
N_{max} (número máximo de soluções).
Saída: (conjunto de N_{max} soluções dominantes).

1) Faça np = N_{max} e $p_1 = 1$.
2) Para $k = 2$ até N_{max} faça
   $p_k = p_{k-1} + \lfloor (|\ | - p_{k-1})/np \rfloor$, ($\lfloor a \rfloor$: maior inteiro $\leq a$).
   np = np -1.
3) Divida o conjunto em N_{max} subconjuntos da seguinte forma:
   ${}_1 = \{ {}^{1},..., {}^{p_1} \}$, ${}_2 = \{ {}^{p_2+1},..., {}^{p_3} \}$,..., ${}_{Nmax} = \{ {}^{p_{Nmax}+1},..., {}^{|\ |} \}$
4) De cada subconjunto ${}_i$ ($i = 1, ..., N_{max}$) selecione aleatoriamente uma solução e descarte as outras soluções. O novo conjunto é formado pelas soluções selecionadas.

Observe que, o método descrito acima requer que os pontos dominantes estejam ordenados ($f_1({}^{i}) < f_1({}^{i+1})$), por isto o método é aplicável somente a problemas bi-objetivos. Na Figura 5 mostra-se um exemplo da aplicação do Algoritmo 2 sobre o espaço bi-objetivo. Neste exemplo, um conjunto com $|\ | = 8$ soluções é reduzido a $N_{max} = 3$ soluções. A Figura 2, ilustra as imagens dos subconjuntos ${}_i$ ($i = 1,...,3$) e os correspondentes pontos ${}^{i} = ({}^{i})$. Note que, $p_2 = 3$ e $p_3 = 5$; e ${}_1 = \{ {}^{j} \in : 1 \leq j \leq p_2 \}$, ${}_2 = \{ {}^{j} \in : p_2+1 \leq j \leq p_3 \}$ e ${}_3 = \{ {}^{j} \in : p_3+1 \leq j \leq 8 \}$.

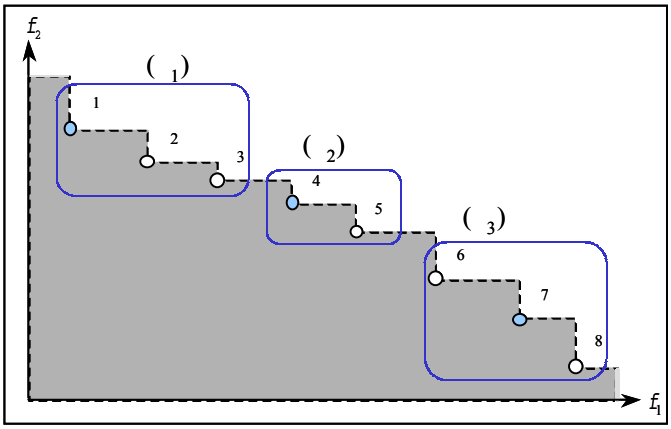


Figura 2. Seleção de soluções distribuídas por toda a fronteira dominante.

No algoritmo genético AGHM, aplica-se a busca local multiobjetivo para melhorar as soluções dominantes da população gerada pelos operadores genéticos. O procedimento BLM descrito no Algoritmo 1 é executado a partir do conjunto de soluções dominantes da população. Para evitar que a busca local consuma a maior parte do tempo computacional do algoritmo genético, sugere-se acionar a busca local a cada β iterações do algoritmo genético.

Ao finalizar a busca local BLM, deve-se atualizar a população ${}'_t$ com o conjunto ${}_{BL}$ de soluções obtido por BLM. A população é atualizada da seguinte maneira:

i) Se $|\ | = |\ |$, substitua em ${}'_t$ as soluções de com as soluções de .

ii) Se $|\ | > |\ |$, insira o conjunto em ${}'_t$ e remova de ${}'_t$ o conjunto e $|\ {}_{BL}| - |\ |$ soluções selecionadas aleatoriamente.

iii) Se $|\ | < |\ |$, insira o conjunto em ${}'_t$, escolha aleatoriamente $|\ | - |\ {}_I|$ soluções de e remova-os de ${}'_t$.

4. AGHM: APLICAÇÃO AO PROBLEMA DE MOCHILA MULTIOBJETIVO

Nesta seção, apresenta-se a implementação do algoritmo genético híbrido multiobjetivo AGHM para resolver o problema da mochila multiobjetivo 0/1.

Uma formulação do Problema da Mochila Multiobjetivo 0/1 (PMM) é a seguinte:

$$\begin{aligned} &\text{Maximizar} && z_j(\ ) = \sum_{i=1}^{q} c_i^j x_i\ , && j = 1,...,r \\ &\text{Sujeito a} && \sum_{i=1}^{q} w_i x_i \le W && \\ & && x_i \in \{0,1\}, && i = 1,...,q \end{aligned} \qquad \textbf{(PMM-1)}$$

onde r é o número de objetivos, q é o número de itens (ou objetos) a serem inseridos na mochila, c_i^j é a utilidade do item i de acordo com o objetivo j, w_i denota um atributo tal como peso ou volume do item i, W é a capacidade da mochila e $= (x_1, ..., x_q)$ é um vetor de variáveis binárias x_i tal que:

$$x_i = \begin{cases} 1, & \text{se o item } i \text{ está na mochila} \\ 0, & \text{caso contrário} \end{cases}$$

O problema consiste em encontrar um conjunto de itens que maximize as utilidades totais $z_j(\ )$.

O problema (PMM-1) foi estudado por Ulungu e Teghem (1995), Teghem et al. (2000) e Gandibleux e Freville (2000). Zitzeler e Thiele (1999), consideram o problema da mochila multidimensional, formulado da seguinte maneira:

$$\begin{aligned} &\text{Maximizar} && z_j(\ ) = \sum_{i=1}^{q} c_i^j x_i\ , && j = 1,...,r \\ &\text{Sujeito a} && \sum_{i=1}^{q} w_i^j x_i \le W_j && j = 1,...,r \\ & && x_i \in \{0,1\}, && i = 1,...,q \end{aligned} \qquad \textbf{(PMM-2)}$$

Neste trabalho são abordados os problemas (PMM-1) e (PMM-2). Apresenta-se a seguir a descrição da implementação dos componentes do algoritmo AGHM aplicado a estes problemas.

4.1. Componentes do Algoritmo Genético

4.1.1.

Uma solução para o problema da mochila é representada por um vetor de q elementos, $= (x_1,...,x_q)$, no qual, $x_i = 1$ se o item i pertence à mochila e, $x_i = 0$, caso contrário.

Para gerar um conjunto inicial de soluções dominantes do problema, utiliza-se uma heurística gulosa para maximizar a combinação linear dos objetivos do problema:

$$z(\ ) = \sum_{i=1}^{r} \lambda_j z_j(\ )\ , \quad \sum_{j=1}^{r} \lambda_j = 1\ , \quad 0 \le \lambda_j \le 1.$$

O vetor peso $\lambda = (\lambda_1,...,\lambda_r)$, geralmente, determina uma direção de busca na fronteira Pareto-ótima. Com o intuito de gerar N soluções que farão parte da população inicial do algoritmo genético AGHM, primeiramente, define-se um conjunto de vetores pesos $\Lambda = \{\lambda^d, \lambda^d = (\lambda_1^d,...,\lambda_r^d)\ , d = 0,..., N-1\}$. Em seguida, maximiza-se a função ponderada $z(\ )$ considerando cada um dos N vetores pesos. A heurística gulosa é descrita a seguir:

3

Entrada: λ^d, $d = 0,..., N-1$ (*vetores pesos - direções de busca*)
Saída: (*conjunto de soluções aproximadas*)
1) Faça $= \varnothing$.
2) Para $d = 0$ até $N-1$ faça
3) Faça $^d = (x_1,...,x_q) = (0,...,0)$.
4) Ordene os itens i em ordem decrescente de $\sum_{j=1}^{r} \frac{\lambda_j^d c_i^j}{w_i^j}$.
5) Faça $i = 1$ e $S_j = 0$, $\forall j = 1,...,r$.
6) Enquanto $S_j + w_i^j \le W_j$, $\forall j = 1,...,r$ faça
7) $x_i = 1$.
$S_j = S_j + w_i^j$, $\forall j = 1,...,r$.
$i = i + 1$.
8) Se $^d \notin$, então faça $= \cup \{ ^d \}$.
9) Fim.

Para o caso bi-objetivo, o conjunto de vetores pesos Λ é determinado da seguinte maneira:

$$\Lambda = \{ \lambda^d \mid \lambda^d = (\lambda, 1-\lambda), \lambda = \frac{d}{N-1}, \ d = 0, ..., N-1 \}.$$

Note que, os N vetores pesos no conjunto Λ determinam diferentes direções de busca distribuídas por toda a fronteira de soluções dominantes.

Se após a execução do Algoritmo guloso 3, o número de soluções na população é menor que N então, para completar a população, gera-se $N - |$ | soluções através do método de geração de soluções diversas para problemas 0/1, sugerida por Glover (1998).

4.1.2.

- *Seleção*: A estratégia adotada para a seleção de soluções é baseada no princípio da roleta (*roulette wheel*).
- *Operador de Recombinação*: Foi implementado o operador *crossover dois pontos* para problemas 0/1 (Goldberg, 1989). Para cada par de soluções pais, gera-se aleatoriamente um número real *rand* sobre o intervalo [0,1]. Se *rand* < p_R (p_R: probabilidade de recombinação), as soluções pais são submetidos a recombinação gerando duas soluções filhos. Para tal, seleciona-se arbitrariamente dois pontos p_1 e p_2 ($1 \le p_1 < p_2 \le q$) e os componentes (bits) dos pais entre estes dois pontos são trocados, produzindo dois filhos.
- *Operador de Mutação*: Este operador é executado em cada filho tentando alterar os bits 0/1. Para cada bit de um filho, gera-se aleatoriamente um número real *rand* sobre o intervalo [0,1]; se *rand* < p_M, altera-se o bit de 0 para 1 ou de 1 para 0 (p_M é a probabilidade de mutação).

4.1.3.

Na população inicial, as soluções geradas através do método de geração de soluções diversas, sugerido por Glover (1998), podem ser infactíveis. Similarmente, os operadores genéticos podem gerar soluções infactíveis. Para factibilizar soluções, remove-se itens da mochila enquanto a restrição de capacidade seja violada. A ordem na qual os itens são removidos é determinado pela razão máxima de lucro/peso: $\gamma_i = max_{j=1}^{r} \{\frac{c_i^j}{w_i^j}\}$. Os itens são ordenados em ordem crescente de γ_i, i.e., os itens com valores pequenos de lucro e valores grandes de peso são removidos primeiro.

4.1.4.

A cada β iterações do algoritmo genético executa-se o procedimento da busca local multiobjetivo BLM (descrito no Algoritmo 1) adaptado para o problema da mochila multiobjetivo. O algoritmo BLM começa a partir do conjunto de soluções dominantes da população atual.

Para o problema da mochila multiobjetivo, foi considerada uma vizinhança baseada na remoção e inserção de itens (*drop-add*). Para uma solução $= (x_1, ..., x_q)$ define-se os conjuntos $J_1 = \{ j \mid x_j = 1\}$ (conjunto de itens na mochila) e $J_0 = \{ j \mid x_j = 0\}$ (conjunto de itens que não pertencem à mochila). A vizinhança de uma solução , ( ), é construída através do seguinte algoritmo:

4

1) Faça ( ) = ∅ e $C_j(\ ) = \sum_{i=1}^{q} w_i^j x_i$, $\forall j = 1,...,r$.
1) Para cada $\nu \in J_1$ faça
2) Para cada $\mu \in J_0$ faça
3) - = $(y_1, ..., y_q)$ tal que, $y_i = x_i$, $i = 1,...,q$, $i \neq \nu$, $i \neq \mu$, $y_\nu = 0$, $y_\mu = 1$.
4) - Se $(C_j(\ ) - w_\nu^j + w_u^j) \leq W_j$, $\forall j = 1,...,r$ então faça ( ) = ( ) ∪{ }.
5) Fim.

Note que, as soluções vizinhas ∈ ( ) são sempre soluções factíveis, portanto não é necessário aplicar a estas soluções o algoritmo de factibilização. Para cada solução vizinha obtida a partir de , os valores dos *r* objetivos são calculados da seguinte maneira:

$$f_j(\ ) = f_j(\ ) - c_\nu^j + c_\mu^j \ , \ j = 1, ..., r.$$

4.2. Determinação de Parâmetros

Alguns parâmetros foram testados para o algoritmo AGHM aplicado ao problema da mochila multiobjetivo. Combinações de parâmetros que geraram bons resultados são apresentados a seguir:

- Tamanho da população (*N*): Este parâmetro depende do tamanho do problema e é fixado da forma como apresentado na Tabela 1.

Tabela 1. Tamanho da população

$q = 50, 100$	$q = 150, 200, 250$	$q = 300, 350, 400, 450, 500$	$q = 750$
$N = 100$	$N = 150$	$N = 200$	$N = 250$

- Número máximo de soluções de elite: $N_{elite} = 30$.
- Probabilidade de recombinação: $p_R = 0.8$.
- Probabilidade de mutação: $p_M = 0.01$.
- Ativação da busca local: A cada $\beta = 4$ iterações.
- Número máximo de caminhos a serem exploradas pela busca local: $N_{max} = 6$.
- Número máximo de iterações da busca local: $Niter = 12$.
- Critério de parada: Como a busca local multiobjetivo BLM explora N_{max} caminhos em paralelo durante no máximo *Niter* iterações então, o algoritmo genético precisa de poucas iterações para encontrar soluções de boa qualidade. O algoritmo genético termina após a execução de 150 iterações, o que significa que a busca local BLM é ativada, aproximadamente, $150/\beta \approx 38$ vezes.

5. RESULTADOS COMPUTACIONAIS

Nesta seção analisa-se o comportamento e o desempenho da metaheurística proposta AGHM aplicado para resolver o problema da mochila multiobjetivo.

Inicialmente, o AGHM é testado considerando dez instâncias do problema (PMM-1) com dois objetivos onde, o número de itens q varia de 50 a 500. A qualidade das soluções geradas pela metaheurística AGHM é avaliada através da comparação com as soluções Pareto-ótimas obtidas pelo algoritmo Branch-and-Bound (B&B) proposto por Ulungu e Teghem (1995). Vale mencionar que, as instâncias e as respectivas soluções ótimas foram gentilmente fornecidas por Daniels Tuyttens, professor da Faculdade Politécnica de Mons, Bélgica (Tuyttens, et al., 2000).

Também foram consideradas quatro instâncias do problema (PMM-2) onde os números de itens são q = 100, 250, 500 e 750. Estas instâncias foram geradas por Zitzler e Thiele (1999). As soluções dominantes obtidas pelo AGHM são comparadas com as soluções dominantes geradas pelo algoritmo SPEA (Strength Pareto Evolutionary Algorithm) proposto por Zitzler e Thiele (1999). Os resultados do SPEA são fornecidos no seguinte site: http://www.tik.ee.ethz.ch/~zitzler. Vale ressaltar que, os resultados de SPEA foram melhores quando comparados com os resultados de cinco algoritmos genéticos multiobjetivos da literatura (Zitzler e Thiele, 1999).

Neste trabalho, para avaliar as soluções geradas pelas metaheurísticas são usados os seguintes métodos de avaliação:

- Medida de distância, proposto por Czyzak e Jaszkiewicz (1998) e Ulungu et al.(1998);
- Erro de utilidade, proposto por Daniels (1992).

Ambas medidas são adaptadas para problemas de maximização. As definições destas metodologias são apresentadas em Arroyo e Armentano (2001).

Os testes computacionais da metaheurística AGHM foram executados em uma estação de trabalho SUN Ultra 60.

5.1. AGHM Comparado com Branch-and-Bound (B&B)

Na Tabela 2, para cada tamanho de instância, apresenta-se o número de soluções eficientes obtidas pelo algoritmo B&B, o número total de soluções e o número de soluções eficientes obtidas pela metaheurística AGHM. Para as 10 instâncias testadas, o algoritmo B&B encontrou 7003 soluções eficientes sendo que a metaheurística AGHM identificou 3590 (51,26%) soluções eficientes.

Na Figura 3 ilustra-se o número de soluções eficientes obtidos pelo algoritmo B&B e pela metaheurística AGHM. Observe que o número de soluções eficientes cresce consideravelmente quando o número de itens é incrementado.

Tabela 2. Número de soluções obtidas pelo algoritmo B&B e pela metaheurística

Número de itens q	B&B	AGHM		
	NSE	NTS	NSEM	%NSE
50	34	34	34	100
100	172	171	170	98,84
150	244	243	234	95,90
200	439	429	413	94,08
250	629	575	474	75,36
300	713	655	520	72,93
350	871	768	417	47,88
400	1000	872	420	42,00
450	1450	1166	430	29,66
500	1451	1217	478	32,94
Total	7003	6130	3590	51,26

NSE: número de soluções eficientes obtidas pelo algoritmo B&B.

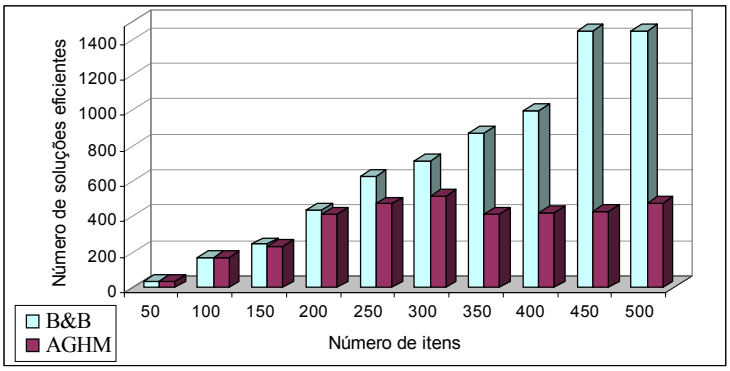


Figura 3. Número de soluções eficientes obtidas pelo algoritmo B&B e pela metaheurística AGHM.

O desempenho da metaheurística AGHM é avaliado através da medida de distância (Czyzak e Jaszkiewicz,1998; Ulungu et al., 1998) e erro de utilidade (Daniels, 1992). Para cada medida, consideram-se o caso médio e o pior caso. Na Tabela 3, para cada tamanho de instância, mostra-se o desempenho da metaheurística AGHM em relação às duas medidas. Observe que, os valores das medidas de distância e erro de utilidade são muito pequenos, o que significa que as soluções encontradas pela metaheurística estão bem próximas às soluções eficientes e que são soluções de boa qualidade de acordo com a função de utilidade linear.

Tabela 3. Desempenho da metaheurística AGHM

Número de itens	Medida de Distância		Erro de Utilidade (%)	
	D_{med}	D_{max}	E_{med}	E_{max}
50	0	0	0	0
100	0,0000048	0,00083	0	0
150	0,000015	0,00079	0,0026	0,0441
200	0,000032	0,00233	0,00097	0,0556
250	0,000317	0,0610	0,0027	0,0784
300	0,000121	0,0217	0,00187	0,0733
350	0,000113	0,0362	0,0108	0,0649
400	0,000007	0,00159	0,0072	0,0653
450	0,000053	0,00091	0,0107	0,0613
500	0,000084	0,0311	0,007468	0,062
Média	0,0000746	0,01564	0,004431	0,05049

Na Tabela 4 mostra-se, para cada tamanho de instância, o tempo (em segundos) gasto pela metaheurística AGHM.

Tabela 4. Tempos computacionais (em segundos) da metaheurística BTMO

$q = 50$	$q = 100$	$q = 150$	$q = 200$	$q = 250$	$q = 300$	$q = 350$	$q = 400$	$q = 450$	$q = 500$
1,15	7,34	17,02	44,25	104,23	120,45	168,43	283,14	466,74	553,23

5.2. AGHM Comparado com a Metaheurística SPEA

O AGHM é comparado com o algoritmo genético SPEA proposto por Zitzler e Thiele (1999). Para as quatro instâncias testadas (q = 100, 250, 500, 750), a metaheurística AGHM obteve 1950

soluções dominantes, enquanto SPEA obteve apenas 220. Na Tabela 5, para cada tamanho de problema, mostra-se o número de soluções dominantes obtidas por cada metaheurística. A Figura 4 ilustra, para cada tamanho de problema, os conjuntos de pontos obtidos pelas metaheurísticas comparadas. Note que, AGHM sempre encontra pontos bem distribuídos por toda a fronteira de Pareto. A metaheurística SPEA apresenta pontos dominantes concentradas numa pequena faixa da fronteira de Pareto e, a distribuição destes pontos piora com o aumento do tamanho do problema.

Tabela 5. Número de soluções encontradas pelas metaheurísticas

Metaheurística	Número de soluções dominantes				Total
	$q = 100$	$q = 250$	$q = 500$	$q = 750$	
AGHM	109	363	597	881	1950
SPEA	69	67	39	45	220

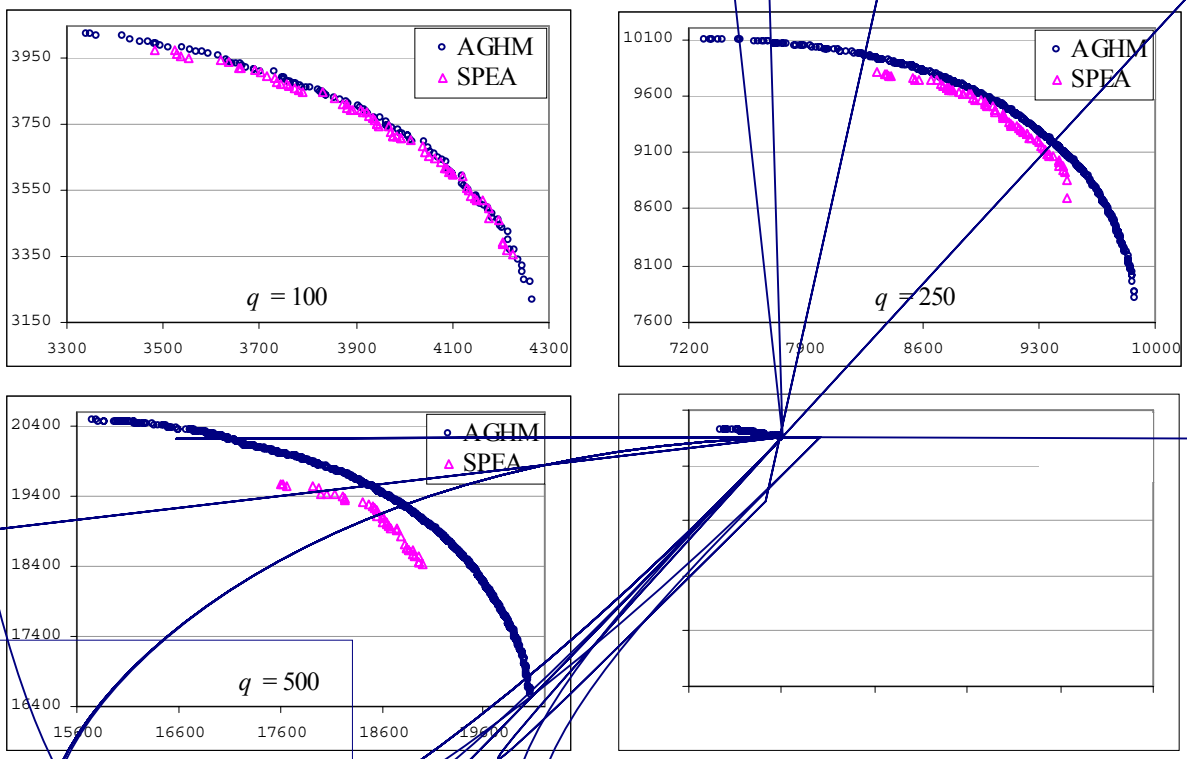


Figura 4. Conjunto de pontos dominantes obtidos pelas metaheurísticas AGHM e SPEA.

7. CONCLUSÕES

Neste artigo foi proposto um algoritmo genético híbrido multiobjetivo e foi aplicado para resolver o problema da mochila multiobjetivo. No algoritmo genético proposto foi incorporada uma estratégia de busca em vizinhança, baseada no conceito de dominância de Pareto, que realiza uma intensificação em paralelo sobre diferentes regiões gerando, rapidamente, várias soluções dominantes e distribuídas por todas a fronteira de Pareto. Através dos testes computacionais realizados sobre instâncias do problema da mochila com dois objetivos, a metaheurística proposta mostrou bom desempenho quando comparada com Branch-and-Bound e um comportamento superior que a metaheurística SPEA da literatura.

REFERÊNCIAS BIBLIOGRÁFICAS

Arroyo J.E.C. & Armentano, V.A. (2001) "Um Algoritmo Genético para Problemas de Otimização Combinatória Multiobjetivo", *Anais do XXXIII Simpósio Brasileiro de Pesquisa Operacional*, pp. 1060-1078.

Baykasoglu A., Owen S. & Gindy N. (1999) "A taboo search based approach to find the Pareto optimal set in multiple objective optimization", *Engineering Optimization*, vol. 31, pp. 731-748.

Ben A.F., Chaouachi J. & Krichen S. (1999) "A hybrid heuristic for multiobjective knapsack problems", In:Voss S,Martello S,Osman I,Roucairol C(eds) Meta-heuristics. Advances and trends in local search paradigms for optimization, pp. 205-212. Kluwer, Dordrecht.

Coello C.A.C. (2000) "An Updated Survey of GA-Based Multiobjective Optimization Techniques", *ACM Computing Surveys*, vol. 32, nº 2, pp. 109-143.

Czyzak P. & Jaszkiewicz A. (1998) "Pareto Simulated Annealing – a metaheuristic technique for multiple objective combinatorial optimization", *Journal of Multi-Criteria Decision Analysis*, vol. 7, pp. 34-47.

Daniels R.L. (1992) "Analytical evaluation of multi-criteria heuristics", *Management Science*, vol. 38, pp. 501-513.

Deb K. Agrawal S. Pratab A. & Meyarivan T. (2000) "A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II". KanGAL report.

Ehrgott M. & Gandibleux X. (2000) "A survey and annotated bibliography of multicriteria combinatorial optimization", *OR Spektrum*, forthcoming.

Fonseca C.M. & Fleming P.J. (1993) "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization", In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, California, pp. 416-423.

Gandibleux X. & Fréville A. (2000) "Tabu search based procedure for solving the 0-1 multiobjective knapsack problem: The two objectives case", *Journal of Heuristics*, 6, pp. 361-383.

Gandibleux, X., Mezdaoui, N. & Fréville A. (1997) "A tabu search procedure to solve multiobjective combinatorial problems", in: *Advances in Multiple Objective and Goal Programming*, R. Caballero, F. Ruiz and R. Steuer (Editors), volume 455 of Lecture Notes in Economics and Mathematical Systems, pp. 291-300, Springer.

Glass C.A. Potts C.N. & Shade P. (1992) "Genetic algorithms and neighborhood search for scheduling unrelated parallel machines", Preprint series No.OR47, University of Southampton, UK.

Glover F.W. (1998) "A Template for Scatter Search and Path Relinking", in *Artificial Evolution. Lecture Notes in Computer Science* 1363. J.-K. Hao. E. Lutton. E. Ronald. M. Schoenauer and D. Snyers (Eds.). Springer-Verlag. pp. 13-54.

Goldberg D.E. (1989) "Genetic Algorithms in Search Optimization and Machine Learning", Reading, Massachusetts: *Addison-Wesley*.

Hansen M.P. (1997) "Tabu search for multiobjective optimization: MOTS". Technical Report, Technical University of Denmark. Paper presented at *The 13th International Conference on Multiple Criteria Decision Making*, January 6-10, Cape Town, South Africa.

Hansen M.P. & Jaszkiewicz A. (1998) "Evaluating the quality of approximations to the nondominated set", Technical Report, Institute of Mathematical Modelling (1998-7), Technical University of Denmark.

Horn J. & Nafpliotis N. (1993) "Multiobjective optimization using the niche Pareto genetic algorithm", IlliGAL Report 93005, Illinois Genetic Algorithm Laboratory, University of Illinois, Urbana, Champaign.

Ishibuchi H. & Murata T. (1998) "A multi-objective genetic local search algorithm and its application to flowshop scheduling", *IEEE Transactions on Systems, Man and Cybernetics-part C: Applications and Reviews*", vol. 28, pp. 392-403.

Ishibuchi H., Yamamoto N., Murata T. & Tanaka H. (1994) "Genetic Algorithms and Neighborhood Search Algorithms for Fuzzy Flowshop Scheduling Problems", *Fuzzy Sets and Systems*, vol.67, pp.81-100.

Jaskiewicz A. (2002) "Genetic local search for multi-objective combinatorial optimization", *European Journal of Operational Research* vol.137, pp. 50-71.

Jones D.F., Mirrazavi S.K. & Tamiz M. (2002) "Multi-objective meta-heuristics: An overview of the current state-of-art", *European Journal of Operational Research* vol.137, pp. 1-19.

Schaffer J.D. (1985) "Multiple objective optimization with vector evaluated genetic algorithms", *International Conference on Genetic Algorithms*, Morgan Kaufmann, New York, pp. 93-100.

Srinivas N. & Deb K. (1995) "Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms" *Evolutionary Computation*, vol. 2(3) pp. 221-248.

Teghem J., Tuyttens D. & Ulungu E.L. (2000) "An iteractive heuristic method for multi-objective combinatorial optimization", *Computer and Operations Research*, vol. 27, pp.621-634.

Tuyttens D., Teghem J., Fortemps Ph. and Van Nieuwenhuyze K. (2000) "Performance of the MOSA Method for the Bicriteria Assignment Problem", *Journal of Heuristics*, 6(3), pp. 295-310.

Ulungu E.L. & Teghem J. (1995) "The Two Phases Method: An Efficient Procedure to Solve Bi-Objective Combinatorial Optimization Problems" *Foundations of Computing and Decision Sciences* 20(2), 149-165.

Ulungu E.L., Teghem J. & Ost Ch. (1998) "Efficiency of interactive multi-objective simulated annealing through a case study", *Journal of the Operational Research Society*, vol.49, pp. 1044-1050.

Van Veldhuizen D.A. & Lamont G.B. (2000a) "Multiobjective evolutionary algorithms: Analysing the state-of art", *Evolutionary Computation*, vol. 8(2), pp. 125-147.

Van Veldhuizen D.A. & Lamont G.B. (2000b) "On Measuring Multiobjective Evolutionary Algorithm Performance", *2000 Congress on Evolutionary Computation*, volume 1, pp. 204-211, Piscataway, New Jersey.

Zitzler E. & Thiele L. (1999) "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach", *IEEE Transactions on Evolutionary Computation*, vol.3, nº 4, pp. 257-271.