RESOLUÇÃO DE PROBLEMAS MULTIMODAIS USANDO UM ALGORITMO GENÉTICO DE CHU-BEASLEY MODIFICADO COM UMA BUSCA LOCAL EFICIENTE

Henrique Uzinski
huzinski@gmail.com

Leonardo Henrique Faria Macedo Possagnolo
leohfmp@gmail.com

Rubén Augusto Romero Lázaro
ruben@dee.feis.unesp.br

Laboratório de Planejamento de Sistemas de Energia Elétrica – LaPSEE
Departamento de Engenharia Elétrica – UNESP – Ilha Solteira
Avenida Brasil, 56 – Centro
15.385-000 ILHA SOLTEIRA, SP, BRASIL

RESUMO

Este artigo apresenta um Algoritmo Genético de Chu-Beasley (AGCB) para resolver problemas multimodais com restrições não-lineares e variáveis contínuas canalizadas. O AGCB manipula as infactibilidades de maneira simples, e por ser não geracional permite controle total de diversidade. São propostas modificações no AGCB original e uma estratégia de busca local eficiente, baseada em sensibilidades da função objetivo. Resultados obtidos para cinco problemas disponíveis na literatura demonstram a eficiência do método proposto.

PALAVRAS CHAVE. Algoritmo Genético de Chu-Beasley. Otimização global. Problemas multimodais.

Área principal: MH – Metaheurísticas.

ABSTRACT

This paper presents a Chu-Beasley's Genetic Algorithm (CBGA) to solve multimodal problems with nonlinear constraints and bounded continuous variables. The CBGA easily handles infeasibility, and because it is not generational, allows full control of diversity. Modifications in the original CBGA are proposed, together with an effective local search strategy, based on the objective function sensitivities. Results obtained for five problems available in the literature demonstrate the effectiveness of the proposed method.

KEYWORDS. Chu-Beasley's Genetic Algorithm. Global optimization. Multimodal problems.

Main area: MH – Metaheuristics.

1. Introdução

Vários problemas de otimização, que aparecem nas áreas de engenharia, economia, biologia, etc., são descritos por modelos matemáticos complexos, pois são altamente não lineares e apresentam característica multimodal, isto é, possuem várias soluções de boa qualidade. Para estes problemas, técnicas de otimização clássicas não fornecem resultados satisfatórios.

Métodos de otimização global são técnicas robustas para resolver problemas complexos de otimização. Eles podem ser classificados em métodos exatos (FLOUDAS, 2000) e heurísticos (ou estocásticos) (ZHIGLJAVSKY E ŽILINSKAS, 2008). Dentre os métodos estocásticos se destacam as meta-heurísticas (GLOVER E KOCHENBERGER, 2003) que fornecem soluções de boa qualidade, com tempos e esforços computacionais aceitáveis.

Existe uma variedade de meta-heurísticas de alto desempenho que apresentam vantagens e desvantagens na resolução de problemas complexos. Com a finalidade de resolver estes problemas multimodais, com característica altamente não linear e variáveis contínuas canalizadas, é apresentado neste trabalho um algoritmo evolucionário, com alguns aspectos modificados, para ser particularmente útil em encontrar múltiplas soluções de alta qualidade, devido à abordagem populacional.

A meta-heurística apresentada é baseada no Algoritmo Genético de Chu-Beasley (AGCB), que foi apresentado em 1997. As principais características do AGCB é a manutenção máxima da diversidade da população e uma estratégia de substituição de um novo elemento que garante que as soluções de excelente qualidade não sejam perdidas durante o processo de resolução, como ocorre no Algoritmo Genético tradicional (CHU E BEASLEY, 1997). Uma nova estratégia de busca local, baseada em sensibilidades também é apresentada neste trabalho.

Os problemas envolvidos nos testes apresentados neste trabalho podem ser encontrados em Floudas e Pardalos (1990), Himmelblau (1972), Hock e Schittkowski (1981), Koziel e Michalewicz (1999), Michalewicz, Nazhiyath e Michalewicz (1996). Eles se caracterizam por apresentarem múltiplas soluções ótimas e regiões de buscas complexas.

O texto deste artigo está organizado da seguinte forma: na seção 2 discutem-se as características gerais dos problemas que serão resolvidos, na seção 3 apresenta-se o AGCB original e as modificações propostas, na seção 4 é apresentada em detalhe uma proposta de metodologia de busca local eficiente, baseada em sensibilidades, na seção 5 apresenta-se os resultados para cinco problemas disponíveis na literatura e na seção 6 são apresentadas as conclusões gerais sobre o método.

As principais contribuições do trabalho são: 1) desenvolvimento de uma meta-heurística eficiente para resolver problemas multimodais, com função objetivo es restrições altamente não lineares e variáveis canalizadas e 2) nova estratégia de busca local eficiente baseada em sensibilidades.

2. Características Gerais dos Problemas Propostos

Os problemas multimodais caracterizam-se por apresentarem múltiplas soluções ótimas, onde algumas podem ser melhores soluções globais e outras melhores soluções locais. A multimodalidade na busca e otimização geralmente provoca dificuldade em um algoritmo de otimização, uma vez que existem muitas soluções atrativas que o algoritmo pode estar direcionado. Portanto, o algoritmo de otimização multimodal tem a tarefa de encontrar múltiplas soluções ótimas simultaneamente ou uma após a outra de forma sistemática (DEB E SAHA, 2010).

Os resultados dos problemas multimodais se encontram em regiões de busca limitadas por restrições, onde a solução ótima localiza-se entre o limite das regiões factíveis e infactíveis ou em uma região factível separada (FLOUDAS E PARDALOS, 1990).

Pretende-se otimizar neste trabalho problemas multimodais, em que na versão de minimização, o objetivo é encontrar uma solução ótima $\boldsymbol{x}^*$ pertencente a uma região de busca S contida em $\mathbb{R}^n$, tal que $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ para todo $\boldsymbol{x}$, com $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$, pertencentes a S, onde S é alguma região de $\mathbb{R}^n$ e a função objetivo f é $f: S \to \mathbb{R}$. As expressões (1)–(4) apresentam um problema multimodal de forma genérica.

$$\min f(\boldsymbol{x}) \tag{1}$$

Sujeito a:

$$g_i(\boldsymbol{x}) \leq 0 \qquad i = 1, \cdots, ng \tag{2}$$

$$h_i(\boldsymbol{x}) = 0 \qquad i = 1, \cdots, nh \tag{3}$$

$$l_i \leq x_i \leq u_i \qquad i = 1, \cdots, n \tag{4}$$

A estrutura genérica de um problema multimodal é dividida em: função objetivo $f(\boldsymbol{x})$ que será minimizada, apresentada em (1), que está sujeita as restrições de desigualdade, $g_i(\boldsymbol{x})$, em (2), e as restrições de desigualdade, $h_i(\boldsymbol{x})$, em (3). Os limites das variáveis canalizadas são mostrados em (4).

3. O Algoritmo de Chu-Beasley

O AGCB é um Algoritmo Genético modificado proposto por Chu e Beasley em 1997, primeiramente para resolver o problema de atribuição generalizado (GAP, do inglês *Generalised Assignment Problem*), que envolve encontrar o custo mínimo de atribuição de t trabalhos para a agentes, de forma que cada trabalho seja atribuído a exatamente um agente, sujeito a restrições de capacidade de cada agente. O GAP é um problema de otimização combinatorial e NP-completo.

O Algoritmo Genético tradicional é um método de otimização que simula o processo de evolução, aplicando operadores genéticos nos indivíduos de uma população. Cada solução da população é avaliada de acordo com uma medida de adaptação (*fitness*). Às soluções altamente adaptadas na população é dada a chance de se reproduzirem. Novas soluções descendentes são geradas, e soluções não adaptadas à população são substituídas. O Algoritmo Genético tradicional tem substituição geracional, isto é, os elementos da população corrente são substituídos por descendentes gerados usando os operadores genéticos. Esta estratégia pode eliminar da população corrente as melhores soluções já encontradas e também pode gerar soluções repetidas. Este processo de avaliação, seleção e reprodução é repetido até que uma solução satisfatória seja encontrada (CHU E BEASLEY, 1997). Além disto, na proposta original do Algoritmo Genético, todas as variáveis do problema devem ser codificadas como variáveis binárias.

A proposta de Chu e Beasley para o um Algoritmo Genético inclui a representação de uma proposta de solução de forma específica para cada tipo de problema, Avaliações de *fitness* e *unfitness* de forma separada e um procedimento de melhoria local. Além disto, propõe-se uma forma diferente de se substituir um indivíduo em uma população.

No passo de geração da população inicial o AGCB, assim como o Algoritmo Genético tradicional, sugere a geração de uma população de indivíduos gerados aleatoriamente. Este processo caracteriza-se pela geração de indivíduos infactíveis e distantes da factibilidade, o que pode ser observado em casos de problemas complexos.

Uma proposta inovadora apresentada pelo AGCB é na manipulação das infactibilidades. Desta maneira, este algoritmo apresenta as infactibilidades e os valores da função objetivo de forma separada, onde as funções objetivo são armazenadas em um vetor chamado *fitness* e as infactibilidades em um vetor chamado *unfitness*. O vetor *fitness* é usado na seleção e o vetor *unfitness*, juntamente com o *fitness*, é usado para substituir um elemento da população, eliminando-se desta forma a necessidade de um parâmetro de penalização, quando as duas informações são juntadas em um único *fitness*. Para o problema (1)–(4), o *k*-ésimo elemento do vetor *fitness* pode ser definido como mostrado em (5), enquanto o *k*-ésimo elemento do vetor *unfitness* pode ser definido como mostrado em (6).

$$fitness_k = f(\boldsymbol{x}_k) \tag{5}$$

$$unfitness_k = \sum_{i=1}^{nh} |h_i(\boldsymbol{x}_k)| + \sum_{i=1}^{ng} \max\big(0, g_i(\boldsymbol{x}_k)\big) \tag{6}$$

Uma diferença significativa entre o AGCB e o Algoritmo Genético tradicional é o procedimento de substituição dos indivíduos na população corrente. No AGCB, pelo processo de seleção, são escolhidas duas soluções geradoras para o processo de recombinação. Além disto, substitui-se apenas um único indivíduo na população corrente a cada geração. Este procedimento de substituição de um único indivíduo na população corrente permite inserir duas estratégias muito importantes no desempenho da meta-heurística. Sendo estas: a melhoria local no descendente gerado a partir da recombinação e o controle absoluto da diversidade.

A melhoria local do descendente gerado sugerida no AGCB pode ser uma melhoria de busca local muito simples ou sofisticada, levando em conta as características específicas do problema. No caso da aplicação em problemas multirrestritos, esta etapa é dividida nas seguintes fases: fase de melhoria da infactibilidade e a fase de melhoria da qualidade.

No processo de substituição, o AGCB sugere substituir apenas um elemento da população corrente pelo descendente gerado, preservando a diversidade completa, ou seja, todos os elementos da população devem ser diferentes. Sendo assim, se o descendente for igual a um elemento da população ele é descartado. Em caso contrário, segue-se os seguintes passos: se o descendente gerado é infactível, verifica-se se esta infactibilidade é menor que a infactibilidade do elemento com maior infactibilidade, caso seja verdadeiro, a substituição procede e em caso contrário o descendente gerado é descartado; se o descendente gerado é factível, substitui-se o elemento da população com maior infactibilidade, mas, se todos os elementos da população forem factíveis, deve-se verificar se o descendente gerado é de melhor qualidade que o indivíduo de pior *fitness* na população corrente para possibilitar a troca. No fluxograma da Figura 1, é apresentado a estrutura de forma simplificada do AGCB original.

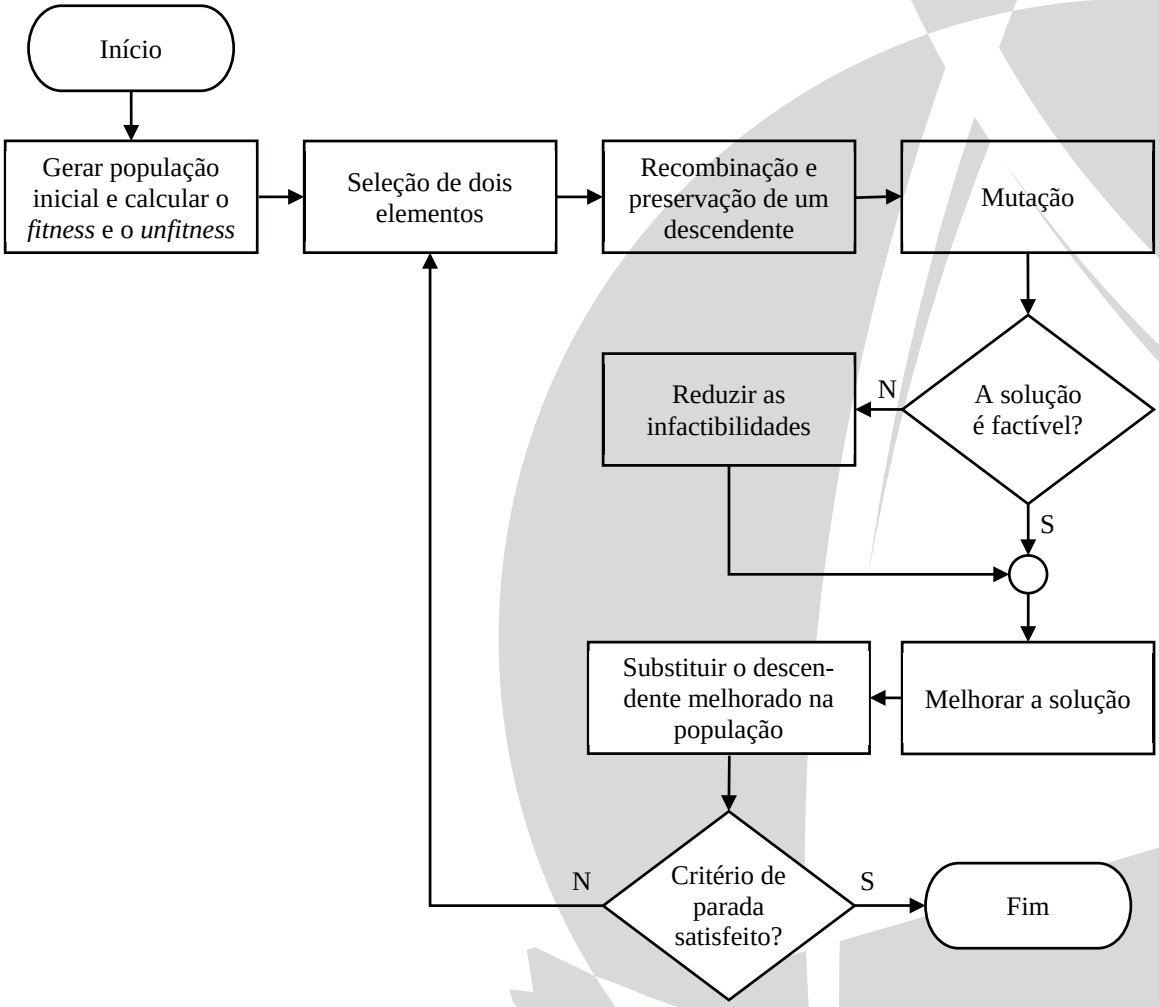

Figura 1: Fluxograma do AGCB.

Para a implementação do AGCB deve-se seguir os seguintes passos apresentados de forma resumida:

1. Especificar parâmetros de controle: tamanho da população inicial, taxa de recombinação, taxa de mutação, etc.;
2. Determinar características gerais do algoritmo: tipo de codificação, população inicial, manipulação das infactibilidades, tipo de seleção, etc.;
3. Encontrar uma população inicial de forma aleatória que se transforma na população corrente. Encontrar o *fitness* e *unfitness* da população corrente;
4. Implementar a seleção para escolher apenas duas soluções geradoras;
5. Implementar a recombinação e preservar apenas um descendente;
6. Implementar mutação do descendente preservado;
7. Implementar uma fase de melhoria local;
8. Decidir se o descendente melhorado entra na população corrente;
9. Se o critério de parada não for satisfeito voltar ao passo 4. Caso contrário termina-se o processo.

Nas subseções seguintes são apresentadas em detalhes as informações sobre as etapas de seleção, recombinação, teste de substituição e as modificações consideradas na metodologia proposta.

3.1. Seleção

O AGCB realiza seleção por torneio. Nesta proposta, dois indivíduos são escolhidos aleatoriamente da população. O indivíduo mais adaptado (com menor valor de *fitness*) é então selecionado para a recombinação. Este processo é então repetido para determinar o segundo indivíduo que será utilizado para produzir os descendentes. Na etapa de seleção o valor do *unfitness* não é utilizado. A Figura 2 ilustra a seleção por torneio.

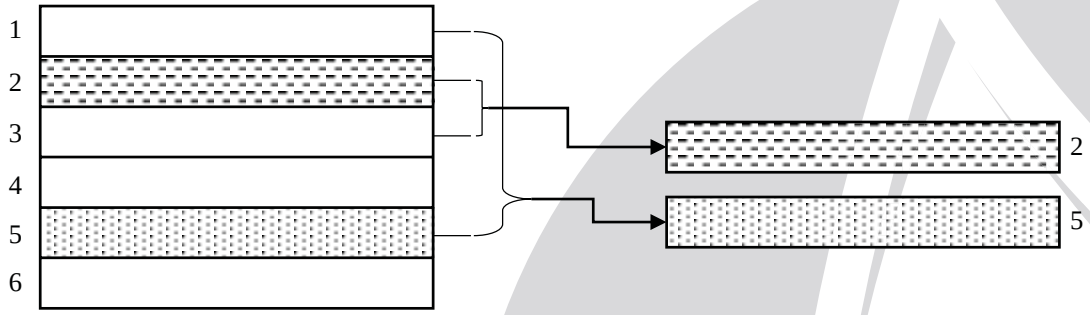

Figura 2: Seleção no AGCB modificado.

Na Figura 2, inicialmente escolhe-se aleatoriamente dois indivíduos da população (no caso, 2 e 3) e verifica-se qual dos dois tem o menor *fitness*. O indivíduo com menor *fitness* (no caso o 2) é o primeiro a ser escolhido para a recombinação. Escolhe-se em seguida outros dois indivíduos (no caso 2 e 5) e escolhe-se, usando o mesmo critério anterior qual será considerado na recombinação (no caso o indivíduo 5). Assim, 2 e 5 terão direito de gerar descendentes.

3.2. Recombinação

Os indivíduos escolhidos na seleção participam da etapa de recombinação. Neste processo o número de pontos de recombinação é definido de acordo com a dimensão do problema e os pontos são selecionados aleatoriamente. A Figura 3 ilustra o processo de recombinação.

Na proposta original de Chu e Beasley, o operador de recombinação utilizado foi o simples, com apenas um ponto, isto é, determina-se um ponto de recombinação p, $0 < p < n$, e a solução descendente consistirá de p genes do primeiro pai e $n - p$ genes do segundo pai, ou vice-versa, com a mesma probabilidade.

Após a recombinação, dois novos indivíduos são obtidos. A proposta de Chu e Beasley consiste em preservar apenas um destes indivíduos, e descartar o outro. Desta forma apenas o des-

cendente com melhor *fitness* é mantido e uma melhoria local é então realizada no melhor descendente. A etapa da melhoria local proposta será descrita em detalhes na seção 4.

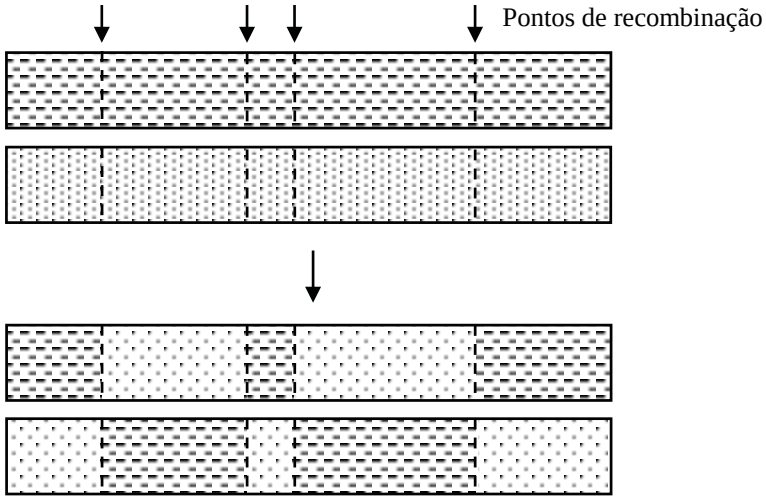

Figura 3: Recombinação no AGCB modificado.

3.3. Teste de Substituição

Para que um descendente entre na população corrente, este deve satisfazer o teste de substituição. Para executar o teste, inicialmente percorre-se o vetor *fitness* e *unfitness* e armazena-se os seguintes parâmetros de controle: $n_{unf} = \{0, -1, n^*_{unf}\}$, em que $n_{unf} = 0$ indica que não existem soluções infactíveis na população corrente, $n_{unf} = -1$ indica que existem soluções infactíveis na população corrente, mas o descendente tem maior infactibilidade que todas elas e $n_{unf} = n^*_{unf}$ indica a posição da solução com maior infactibilidade no vetor *unfitness* (maior que a infactibilidade do descendente), $n_{fit} = \{-1, n^*_{fit}\}$, em que $n_{fit} = -1$ indica que todas as soluções da população são melhores que a descendente e $n_{fit} = n^*_{fit}$ indica a posição da solução de pior qualidade na população e também de menor qualidade que o descendente e, portanto, candidato a ser retirado. Também, deve-se usar um vetor auxiliar n_{aux} para armazenar a posição das soluções da população que não satisfazem o critério de diversidade com o descendente, assim como o tamanho kn que define o número de soluções que não satisfazem o critério de diversidade. A preservação da diversidade pode, ocasionalmente, mudar o tamanho da população. Assim, n_{pop} define o tamanho da população ideal (tamanho da população inicial) e n_{pc} o tamanho da população corrente.

Com os parâmetros especificados, o teste de diversidade, que determina o processo de substituição de um elemento na população por um descendente assume a seguinte forma:

1. Percorrer os vetores *fitness* e *unfitness*, encontrar o valor dos parâmetros de controle e verificar se o descendente satisfaz o critério de diversidade;
2. Se o descendente não satisfaz o critério de diversidade ($kn \neq 0$) fazer seguinte:
    a. Se o descendente gerado é factível ($n_{unf} = 0$) e a função objetivo deste descendente é melhor que todas as soluções que não satisfazem o critério de diversidade, então incorporar o descendente na população e eliminar todas as kn soluções que não satisfazem o critério de diversidade. Fazer $n_{pc} = n_{pc} - kn + 1$ e ir ao passo 6;
    b. Em caso contrário eliminar o descendente gerado e ir ao passo 6.
3. Se o descendente satisfaz o critério de diversidade fazer seguinte:
    a. Se $n_{pc} < n_{pop}$ então incorporar o descendente na população, fazer $n_{pc} = n_{pc} + 1$ e ir ao passo 6;
    b. Em caso contrário ir ao passo 4.

4. Se o descendente for infactível fazer seguinte:
    a. Se não existem soluções infactíveis ($n_{unf} = 0$), eliminar o descendente gerado e ir ao passo 6. Em caso contrário ir ao passo 4(b);
    b. Se existem soluções infactíveis ($n_{unf} \neq 0$) mas todas têm menor infactibilidade que do descendente gerado ($n_{unf} = -1$) eliminar o descendente gerado e ir ao passo 6. Em caso contrário ir ao passo 4(c);
    c. Neste caso, $n_{unf} = n^*_{unf}$ e existe uma solução com maior infactibilidade que do descendente gerado e localizado na posição n^*_{unf}. Incorporar o descendente gerado na população, retirando a solução que se encontra na posição n^*_{unf} e ir ao passo 6.
5. Se o descendente gerado for factível fazer seguinte:
    a. Se existem soluções infactíveis então $n_{unf} = n^*_{unf}$ e neste caso substituir a solução infactível localizada na posição n^*_{unf} pelo descendente gerado e ir ao passo 6. Em caso contrário ir ao passo 5(b);
    b. Se todas as soluções factíveis são de melhor qualidade, $n_{fit} = -1$, então eliminar o descendente gerado e ir ao passo 6. Em caso contrário ir ao passo 5(c);
    c. Existe uma solução factível de pior qualidade localizada na posição n^*_{fit}. Incorporar o descendente gerado na população trocando com o solução armazenada na posição n^*_{fit} da população e ir ao passo 6.
6. Terminar o processo de substituição atualizando alguns parâmetros e a solução incumbente, se for o caso.

3.4. Modificações Propostas no AGCB Original

Na metodologia proposta, sugere-se as seguintes modificações ao AGCB, com as respectivas justificativas:

1. Implementar uma etapa de melhoria da população inicial: verifica-se experimentalmente que, aplicando um algoritmo de busca local aos indivíduos da população inicial, reduz-se o esforço computacional na resolução do problema
2. Utilizar uma população com tamanho reduzido: já que nos problemas abordados existe a presença de variáveis contínuas, a etapa de melhoria local pode demandar um alto esforço computacional. Utilizar populações de tamanho elevado pode fazer com que a metodologia leve muito tempo para convergir;
3. Recombinação de vários pontos, em vez de um único ponto: opta-se em utilizar uma estratégia mais sofisticada de recombinação, de forma a obter maior diversificação nesta etapa;
4. Manter apenas o descendente com o melhor valor de *fitness*: na proposta original de Chu e Beasley, a escolha do descendente que será melhorado é aleatória. A proposta deste trabalho considera manter o descendente com o melhor valor de *fitness* esperando que este seja o mais promissor para encontrar soluções de melhor qualidade;
5. Eliminar a etapa de mutação: a mutação é utilizada no Algoritmo Genético tradicional para que características que não estão presentes na população possam aparecer. Entretanto, a etapa de melhoria local elimina esta necessidade.

Na seção seguinte é apresentada em detalhes a metodologia de busca local utilizada neste trabalho.

4. Busca Local Baseada em Sensibilidades

Apresenta-se nesta seção o algoritmo de busca local desenvolvido para melhorar as solu-

ções obtidas na etapa de recombinação do AGCB modificado. A estratégia consiste-se em discretizar o espaço de busca e realizar modificações aleatórias nas variáveis do problema. Estas modificações, entretanto, consideram a variação obtida na função objetivo na última vez que a variável foi alterada. Também, estas variações devem ser pequenas.

Considere que a solução $\boldsymbol{x}^r$ seja a preservada após a operação de recombinação. Define-se então a função objetivo penalizada de $\boldsymbol{x}^r$, $F(\boldsymbol{x}^r)$, como mostrado em (7).

$$F(\boldsymbol{x}^r) = f(\boldsymbol{x}^r) + \rho\left\{\sum_{i=1}^{nh}[h_i(\boldsymbol{x}^r)]^2 + \sum_{i=1}^{ng}\beta_i[g_i(\boldsymbol{x}^r)]^2\right\} \quad (7)$$

Em (7), a função objetivo penalizada $F(\boldsymbol{x}^r)$, é composta pela função objetivo $f(\boldsymbol{x}^r)$ do problema, somada ao somatório do quadrado das funções de infactibilidades (restrições violadas): $h(\boldsymbol{x}^r)$ (restrições de igualdade), $g(\boldsymbol{x}^r)$ (restrições de desigualdade). Além disto, em (7), $\rho \gg 1$ é um parâmetro de penalização. O parâmetro β_i é definido como, $\beta_i = 0$ se $g_i(\boldsymbol{x}) \leq 0$ e $\beta_i = 0$ se $g_i(\boldsymbol{x}^r) > 0$, nh é o número de restrições de igualdade e ng o número de restrições de desigualdade.

Define-se em seguida um vetor de sensibilidades, $\boldsymbol{s}$, de dimensão n (onde n é o número de variáveis do problema), inicialmente com todos os elementos, s_i, iguais a zero. Determina-se então, quais são os valores de cada passo de discretização das variáveis, Δ_i, como mostrado em (8), onde u_i e l_i são o limite superior e inferior da variável x_i^r, respectivamente, Γ é o número (inteiro) de discretizações e μ é um fator de ajuste dinâmico, inicializado em $\mu = 1$.

$$\Delta_i = \mu \cdot \frac{(u_i - l_i)}{\Gamma} \qquad i = 1, \cdots, n \quad (8)$$

Procede-se escolhendo aleatoriamente uma variável do problema para ter o valor modificado. Suponha que a variável x_k^r seja escolhida. A Figura 4 ilustra como x_k^r é discretizada.

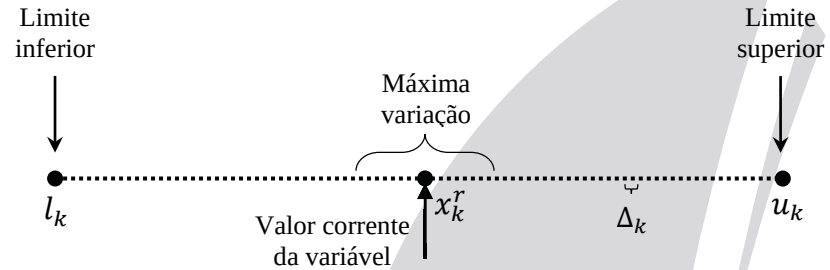

Figura 4: Discretização de uma variável na melhoria local.

Em seguida determina-se o valor da variação δ_k, como mostrado em (9), onde φ é um número aleatório gerado entre zero e um ($0 \leq \varphi \leq 1$), γ é um fator de escala e $\lceil\cdot\rceil$ representa a função teto, que arredonda o valor de $\gamma \cdot \varphi$ para o número inteiro imediatamente superior.

$$\delta_k = \lceil \gamma \cdot \varphi \rceil \cdot \Delta_k \quad (9)$$

No passo seguinte, deve-se determinar o novo valor de x_k^r, como mostrado em (10).

$$\tilde{x}_k^r = x_k^r \pm \delta_k \quad (10)$$

Em (10), como inicialmente $s_k = 0$, escolhe-se, com igual probabilidade, aumentar ou diminuir o valor da variável x_k^r. Note que em (10), $\tilde{x}_k^r$ pode violar o limite inferior l_k ou superior u_k. Neste caso considere a correção (11).

$$\hat{x}_k^r = \begin{cases} \tilde{x}_k^r, & \text{se } l_k \leq \tilde{x}_k^r \leq u_k \\ l_k, & \text{se } \tilde{x}_k^r < l_k \\ u_k, & \text{se } \tilde{x}_k^r > u_k \end{cases} \quad (11)$$

Finalmente deve-se atualizar a solução com $\hat{x}_k^r$. Denomine a nova solução obtida de $\hat{\boldsymbol{x}}^r$. A função objetivo penalizada, $F(\hat{\boldsymbol{x}}^r)$, é então calculada para verificar a qualidade da nova proposta de solução. Duas situações podem então ocorrer: 1) $F(\hat{\boldsymbol{x}}^r) \leq F(\boldsymbol{x}^r)$, neste caso define-se $s_k = 1$,

se em (10) x_k^r foi incrementada, ou $s_k = -1$, se em (10) x_k^r foi decrementada; além disto, deve-se atualizar a solução corrente $\boldsymbol{x}^r \leftarrow \hat{\boldsymbol{x}}^r$. 2) $F(\hat{\boldsymbol{x}}^r) > F(\boldsymbol{x}^r)$, neste caso define-se $s_k = -1$ se em (10) x_k^r foi incrementada ou $s_k = 1$ se em (10) x_k^r foi decrementada; neste caso a solução corrente não é atualizada.

O procedimento descrito é então repetido, isto é, escolhe-se outra variável para ter o valor modificado. No passo (10) entretanto, deve-se considerar o valor de s_k, ou seja, se $l_k \leq x_k^r \leq u_k$ e $s_k = 1$, então deve-se incrementar o valor de x_k^r, se $s_k = -1$ decrementa-se o valor de x_k^r, e se $s_k = 0$ escolhe-se aleatoriamente entre incrementar ou decrementar x_k^r. Agora, para as situações em que $x_k^r = l_k$ ou $x_k^r = u_k$, deve-se considerar incrementar (se $x_k^r = l_k$) ou decrementar (se $x_k^r = u_k$) x_k^r, independente do valor de s_k.

Este processo é então repetido até que nenhuma melhoria seja obtida ou até um número máximo de iterações Υ. O fator μ deve ser ajustado dinamicamente durante o processo, ou seja, a partir de τ iterações, $\tau < \Upsilon$, sem melhoria da função objetivo penalizada, atualiza-se $\mu \leftarrow \mu/2$ e recalcula-se os valores de cada passo de discretização das variáveis, Δ_i, como mostrado em (8).

Esta estratégia de busca local pode ser facilmente estendida para problemas com variáveis inteiras. No caso, se as variáveis são inteiras pode-se modifica-las aumentando ou diminuindo-se um número inteiro pequeno de unidades, dependendo do estado de sensibilidade anterior. As variáveis binárias mudam de 0 para 1 ou 1 para 0 se elas são escolhidas.

A melhoria local é aplicada tanto nos indivíduos da população inicial quanto na busca local no indivíduo recombinado. Neste trabalho, adota-se os seguintes valores para os parâmetros apresentados: $\rho = 10^7$, $\Gamma = 1000$ e $\gamma = 100$.

5. Resultados

Para avaliar a eficiência da meta-heurística proposta, são apresentados, na Tabela 1, os resultados obtidos para cinco problemas teste. Os resultados obtidos são os mesmos e que os melhores resultados disponíveis na literatura. Os cinco problemas utilizados na análise de desempenho do AGCB modificado estão presentes em g1: Floudas e Pardalos (1990), g2: Koziel e Michalewicz (1999), g3: Michalewicz, Nazhiyath e Michalewicz (1996), g4: Himmelblau (1972) e g5: Hock e Schittkowski (1981), sendo que em todos os casos é considerada a versão de minimização do problema. Todos os problemas considerados também podem ser encontrados em Liang *et al*. (2006).

Os testes foram realizados em um computador com processador Intel Core i5 de 2,50 GHz, com 6 GB de RAM e rodando Windows 7. A implementação do AGCB foi realizada em Matlab versão 2013a. Na Tabela 1 são apresentados os resultados obtidos para cada um dos problemas resolvidos, onde n é o número de variáveis de cada problema, e o número de restrições é dado por EL, IL, EN e IN, que são, respectivamente, o número de equações lineares, inequações lineares, equações não lineares e inequações não lineares.

Tabela 1: Resultados da meta-heurística AGCB proposta.

Problema	Tipo	n	EL	IL	EN	IN	Valor ótimo
g1	quadrático	13	0	9	0	0	-15,0000
g2	não linear	20	0	1	0	1	-0,8036
g3	não linear	10	0	0	1	0	-1,0005
g4	não linear	5	0	0	0	6	-30665,5387
g5	não linear	4	0	2	3	0	5126,4967

Os resultados mostrados na Tabela 1 indicam que a metodologia foi capaz de encontrar as melhores soluções conhecidas para os cinco problemas testados.

Para cada problema foram realizados 10 testes, sendo que em cada teste considera-se como critério de parada um número máximo de iterações, ou se a convergência desejada já foi obtida. Considera-se que a convergência foi alcançada quando o valor da função objetivo (5) mais as infactibilidades (6), da melhor solução, diferem do melhor valor conhecido da função objetivo conhecida para o problema de um parâmetro $\varepsilon \ll 1$. No caso considera-se $\varepsilon = 10^{-4}$. Adota-se para os testes uma população com 10 indivíduos.

A Tabela 2 mostra o tempo médio que o algoritmo proposto levou para resolver cada um

dos problemas analisados, considerando 10 testes para cada um deles. Define-se um número máximo de gerações igual a 200. Em todos os casos o algoritmo parou após obter a convergência estabelecida.

Tabela 2: Desempenho da metodologia proposta na resolução dos problemas teste.

Problema	Tempo médio [s]
g1	15,20
g2	145,55
g3	0,75
g4	96,77
g5	11,34

Os resultados obtidos indicam que a metodologia proposta é capaz de encontrar soluções de ótima qualidade para problemas altamente complexos, com tempos computacionais aceitáveis.

6. Conclusões

Neste trabalho, o objetivo foi apresentar um algoritmo genético de Chu-Beasley modificado, com uma estratégia de busca local eficiente, baseada em sensibilidades, para a resolução de problemas de otimização multimodais, com restrições não lineares e variáveis canalizadas.

Os problemas considerados nos testes possuem regiões complexas, e são de difícil convergência para a solução ótima global. A meta-heurística em questão apresenta características com diferenças significativas do algoritmo genético tradicional, desde a geração da população inicial até sua convergência para a solução ótima. Desta forma, não foram necessárias muitas iterações para encontrar a solução próxima da ótima ou a ótima.

A busca local proposta mostrou-se capaz de melhorar a qualidade dos indivíduos recombinados de forma eficiente. Este processo em geral se caracteriza por uma rápida convergência para as soluções ótimas locais.

Desta forma, por ser definida para aplicações gerais, a metodologia mostra-se promissora para ser utilizada em problemas de diversas áreas.

7. Agradecimentos

Os autores agradecem à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo n° 2014/23741-9, pelo apoio financeiro.

Referências

Chu, P. C. e Beasley, J. E. (1997), A genetic algorithm for the generalised assignment problem, *Computers & Operations Research*, v. 24, n. 1, p. 17–23.

Deb, K. e Saha, A. (2010), Finding multiple solutions for multimodal optimization problems using a multi-objective evolutionary approach, *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, New York, p. 447–454.

Floudas, C. A., *Deterministic global optimization: theory, methods and applications*, Nonconvex optimization and its applications, vol. 37, Kluwer Academic Publishers, Dordrecht, 2000.

Floudas, C. A. e Pardalos, P. M., *A Collection of Test Problems for Constrained Global Optimization Algorithms*, Lecture Notes in Computer Science, vol. 455, Springer-Verlag, Berlin, 1990.

Glover, F. e Kochenberger, G., *Handbook of metaheuristics*, Kluwer Academic, Boston, 2003.

Himmelblau, D. M., *Applied nonlinear programming*, McGraw-Hill, New York, 1972.

Hock, W. e Schittkowski, K., *Test examples for nonlinear programming codes*, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, Berlin, 1981.

Koziel, S. e Michalewicz, Z. (1999), Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, vol. 7, n. 1, p. 19–44.

Liang, J. J., Runarsson, T. P., Mezura-Montes, E., Clerc, M., Suganthan, P. N., Coello, C. A. C. e Deb, K. (2006), Problem definitions and evaluation criteria for the CEC 2006, Special Session on Constrained Real-Parameter Optimization.

Michalewicz, Z., Nazhiyath, G. e Michalewicz, M. (1996), A note on usefulness of geometrical crossover for numerical optimization problems, *Proc. of the 5th Annual Conference on Evolutionary Programming*, San Diego, p. 305–312.

Zini, É. de O. C. (2009), Algoritmo Genético especializado na resolução de problemas com variáveis contínuas e altamente restritos. Dissertação (Mestrado em Engenharia Elétrica) – Faculdade de Engenharia de Ilha Solteira – FEIS – UNESP, Ilha Solteira-SP.

Zhigljavsky, A. e Žilinskas, A., *Stochastic global optimization*, Springer optimization and its applications, vol. 9, Springer, Berlin; 2008.