Algoritmos Evolutivos para Problemas de Otimização Numérica com Restrições

Alexandre César Muniz de Oliveira[*1], Luiz Antonio de Nogueira Lorena[**1]

(1)Área de Pesquisa Operacional
Laboratório Associado de Computação e Matemática Aplicada
Instituto Nacional de Pesquisas Espaciais (INPE)

(*)Doutorado, Bolsa CAPES/PICDT, e-mail: acmo@lac.inpe.br; (**) Orientador

Resumo

Este artigo é uma resenha sobre aplicação de algoritmos evolutivos para problemas de otimização numérica sujeita a restrições. Várias abordagens são apresentadas que ilustram bem a problemática de construção de algoritmos evolutivos para minimização ou maximização de uma função objetivo, considerando restrições do tipo equações ou inequações, sejam estas lineares ou não. Este estudo é a primeira etapa de um trabalho que visa o desenvolvimento de um *framework* completo que possibilitará uma gama significativa de aplicações na área de otimização.

Palavras-Chave: algoritmos evolutivos, otimização, otimização numérica com restrições.

Introdução

Problemas de otimização numérica podem ser formulados genericamente como:

$$\begin{aligned} &\text{Otimize } f(x), \quad x=(x_1, x_2, x_3, .., x_n)^T \in \Re^n \\ &\text{Sujeito a } p \geq 0 \text{ equações: } c_i(x) = 0,\ i=0,..,p \\ &\text{e } m-p \geq 0 \text{ inequações: } \quad c_i(x) \leq 0,\ i=p+1, ..., m \end{aligned} \qquad (1)$$

onde a função objetivo, bem como o conjunto de restrições do problema em questão podem ser lineares ou não-lineares. Em otimização numérica, as variáveis de controle podem assumir qualquer valor (inteiro ou real) que minimize ou maximize o valor da função objetivo.

Os chamados algoritmos evolutivos têm sido bastante utilizados por possuírem uma série de características que os tornam bem atraentes para este tipo de aplicação. Algumas delas: são métodos de otimização global, robustos, facilmente adaptáveis; realizam buscas em paralelo, sem necessidade de derivadas; podem ser eficientemente combinados com heurísticas de busca local. Naturalmente, também possuem algumas desvantagens. Por exemplo, podem ser mais lentos que outras alternativas e possuem parâmetros que devem ser bem ajustados para se obter eficácia.

A idéia básica dos algoritmos evolutivos é a de manter uma população de indivíduos, representando soluções candidatas para problemas concretos, que evolui ao longo de gerações através de um processo de competição, onde os mais aptos (melhores *fitness*) têm maiores chances de sobreviver e se reproduzir. A reprodução se baseia em um processo de seleção de indivíduos e modificação das soluções candidatas que eles representam, através de operadores como cruzamento (ou *crossover*) e mutação.

Algoritmos Evolutivos para Problemas de Otimização com Restrições

O principal problema para construção de algoritmos evolutivos eficientes para problemas de otimização com restrições reside na forma de se tratar (evitar, manter e avaliar) as soluções não viáveis. Quando se trabalha apenas com restrições de intervalo constantes, tem-se efetivamente um espaço de busca convexo. Entretanto, restrições não-lineares podem formar espaços de busca razoavelmente complexos.

Há duas abordagens básicas para manipulação de soluções não viáveis: (a) modificar o conjunto de operadores evolutivos para não haver violação de restrição; (b) penalizar soluções que violem alguma restrição. A seguir algumas das recentes abordagens são apresentadas.

Abordagens envolvendo penalidades

Penalizar um indivíduo não viável significa acrescentar pesos a cada restrição que for violada pela solução que ele representa. Isto tende a reduzir a aptidão desse indivíduo e, consequentemente, sua probabilidade de participar do processo de evolução (seleções e cruzamentos). Descartar soluções não viáveis pode ser considerado como impor uma "penalidade da morte", ou seja, o indivíduo assim penalizado não participa mais do processo de evolução da população. O descarte de um indivíduo não viável é um método popular usado por muitas técnicas, como estratégias de evolução [1] e tende a funcionar razoavelmente bem, quando o espaço viável de busca V é convexo e se constitui em uma razoável parte do espaço de busca total ® [2].

Em espaços de viabilidade não convexos, restringir o acesso a regiões não viáveis que poderiam funcionar como "corredores" para regiões viáveis, não produz bons resultados [2]. Da mesma forma, é mais eficiente melhorar um conjunto de soluções não viáveis que rejeitá-las, quando a razão |V|/|®| é pequena [2]. Sabe-se que

os algoritmos evolutivos otimizam por combinar informações parciais de toda a população. Dessa forma, uma solução não viável também pode prover informação essencial para o processo de evolução e não deve ser simplesmente descartada [3] .

Outra forma de impor penalidades é criar um *ranking,* onde as avaliações das soluções viáveis sejam sempre mapeadas para valores melhores que as soluções não viáveis. Ou seja, a pior solução viável é melhor que a melhor solução não viável. Basicamente, para uma constante *r* a ser determinada, tem-se [4]:

$$aval(x) = \begin{cases} f(x), x \in V \\ f(x) + r.p_j(x,t), x \in \beta - V \end{cases} \quad (2)$$

Algumas abordagens que envolvem penalidades avaliam todos os indivíduos (viáveis ou não) da mesma forma, através da função objetivo modificada, onde cada restrição $p_j(x)$ entra na avaliação associada a um peso. Esses pesos, nesse caso, servem para ponderar as penalidades, inclusive, definindo as mais importantes para o processo de otimização e devem ser corretamente sintonizados para dar um comportamento suave a cada uma das componentes (restrições) da função objetivo modificada. Neste caso, também existem várias estratégias associadas: penalidade uniforme, variável, adaptativa e dupla.

O método de Homaifar trabalha com uma matriz fixa R_{ij} de níveis de violação *i* para cada restrição *j*[5]. O desempenho deste método depende fortemente de R_{ij} [2].

$$aval(x) = f(x) + \sum_{j=1}^{m} R_{ij} p_j^{\,2}(x) \quad (3)$$

Joines & Houck propuseram penalidade variável, reduzindo também a quantidade de parâmetros a serem ajustados. Apenas C, α e β que, em geral, recebem valores pequenos [6].

$$eval(x) = f(x) + (Ct)^{\alpha} \sum_{j=1}^{m} p_j^{\beta}(x) \quad (4)$$

A idéia por trás desse controle determinístico de pesos está em negligenciar as violações dos indivíduos não viáveis, a princípio, e no decorrer das gerações, ser mais rigoroso, direcionando o processo de busca progressivamente para os subespaços de busca viáveis.

A penalidade também pode variar em função do desempenho do algoritmo evolutivo e não simplesmente em função do número de gerações. A penalidade adaptativa, proposta por Bean & Hadj-Alouane, por exemplo, aumenta ou diminui dependendo da tendência momentânea do algoritmo de estar gerando mais ou menos indivíduos viáveis[7]. A penalidade dupla, proposta por Le Riche, balanceia a escolha de indivíduos viáveis ou não, trabalhando com dois *ranking's* de avaliação: um que utiliza penalidade alta e outro, penalidade baixa. A cada geração é usado um *ranking* diferente para selecionar indivíduos para cruzamentos e mutações[8].

Memória de Comportamento

Schoenauer & Xanthakis propuseram a memória de comportamentos que difere um pouco das outras abordagens[9]. A principal motivação para esta nova proposta é que a imposição de penalidades não é eficaz para muitos tipos de problemas. O fundamento principal consiste em tratar as restrições em uma ordem particular. Primeiramente, evolui-se uma população aleatória inicial (através de algum algoritmo evolutivo padrão), usando uma avaliação de aptidão que relacione a função objetivo com a primeira restrição, até que um certo percentual da população (*flip threshold*) seja viável para esta restrição. Depois, repete-se esse processo para cada uma das restrições restantes, sempre usando a população final da evolução anterior como ponto de partida, eliminando-se os indivíduos que violem a restrição anterior. Quando não houver mais restrições, o processo é executado mais uma vez sobre os indivíduos viáveis, utilizando, desta vez, apenas a função objetivo na avaliação desses indivíduos[9].

Sistemas GENOCOP

Genocop (*GEnetic algorithm for Numerical Optimization for COnstraits Problems*) é um sistema baseado em algoritmo genético para otimização de funções não-lineares com ou sem restrições, desenvolvido por Michalewicz[10]. Existem três versões disponíveis que possuem filosofias de trabalho diferentes. O GENOCOP (posteriormente re-batizado de GENOCOP I) pode minimizar ou maximizar funções não-lineares com restrições de igualdade ou desigualdade lineares.

O GENOCOP I é serve de base aos demais GENOCOP's e tem sido utilizado como modelo-teste para pesquisadores no mundo inteiro. O primeiro passo do GENOCOP I é remover as igualdades, eliminando a mesma quantidade de variáveis e, ao mesmo tempo, reduzindo o espaço de busca. Após a eliminação de igualdades, as restrições restantes, na forma de desigualdades lineares, formam um conjunto convexo que garante que combinações lineares de soluções geram soluções viáveis sem a necessidade de verificação de

restrições. O GENOCOP I provê uma série de operadores evolutivos, tais como: as mutações uniforme e não-uniforme, e os cruzamentos aritmético, heurístico e simples[10].

O GENOCOP II, por sua vez, é uma versão híbrida desenvolvida para trabalhar com restrições não-lineares. A função objetivo é modificada pela inclusão das restrições não-lineares sujeitas a uma penalidade variável [10]:

$$F(x, \tau) = f(x) + 1/(2\tau)\, A^T A \qquad (5)$$

onde $1/(2\tau)$ é o fator crescente de penalidade, A é o conjunto de todas as equações não-lineares acrescido das inequações não-lineares que estiverem sendo violadas por uma solução x^* encontrada até então, usando o GENOCOP I. Este último, operando apenas com restrições lineares.

O GENOCOP III é a última e, segundo Michalewicz, "a mais promissora" versão para operar com restrições não-lineares [10]. Também incorpora o GENOCOP I, mas acrescenta duas populações separadas, onde o desenvolvimento dos indivíduos de uma afeta a avaliação dos indivíduos da outra. A primeira população Ps consiste dos chamados pontos de busca S que satisfazem, a princípio, as restrições lineares (obtido via GENOCOP I). A segunda população Pr consiste dos chamados pontos de referência R que satisfazem todas as restrições (inclusive as não-lineares). Os pontos de R são avaliados diretamente pela função objetivo, mas os pontos de S são reparados para efeito de avaliação da seguinte forma: para avaliar $s \in S$, é selecionado aleatoriamente, um indivíduo $r \in R$, e são gerados pontos $z = a\, s + (1 - a)\, r$ (a = {números reais consecutivos $\in (0,1)$}) até que um deles não viole nenhuma restrição. Uma vez viável, o ponto z é avaliado via função objetivo e, se for melhor que o ponto de referência r, este último é substituído por z [10].

Em outras palavras, o GENOCOP III cria linhas de busca entre regiões viáveis e não viáveis, com intensidade de busca maior na vizinhança dos melhores pontos de referência. Em um segundo estágio alguns pontos de referência são movidos para Ps e o processo se repete. O GENOCOP III foi submetido a uma bancada significativa de problemas-teste e teve um desempenho bom, segundo o próprio autor[10] e segundo outros autores que também têm usado-o para efeito de comparação com suas propostas [11].

Conclusão

Este trabalho focalizou os algoritmos evolutivos para problemas de otimização numérica com condições de restrição. Foram apresentadas abordagens para tratar (evitar, manter e avaliar) as soluções não viáveis baseadas em penalidades, em operadores evolutivos especialmente construídos, e em reparação de soluções não viáveis. São citados trabalhos que mostram um desempenho satisfatório para o GENOCOPIII. O estudo desenvolvido neste trabalho abre perspectivas concretas para a construção de um *framework* para otimização numérica competitivo, baseado no Algoritmo Genético Construtivo proposto por Lorena e Furtado [12]. Uma versão expandida deste trabalho foi apresentada como monografia de exame de qualificação do curso CAP/INPE [13].

Referências

1 Bäck, T. Hoffmeister, F. and Schwefel, H.P. A survey of evolution strategies. In Lashon B. Belew, Richard K.; Booker, editor, In Proc. 4th Internat. Conf. on Genetic Algorithms, pages 2--9, San Diego, CA, July 1991. Morgan Kaufmann.

2 Michalewicz, Z. and Schoenauer, M. Evolutionary algorithms for constrained parameter optimization problems. Evolutionary Computation, 1996.

3 Richardson, J.T., M.R. Palmer, G. Liepins and M. Hilliard. Some Guidelines for Genetic Algorithms with Penalty Functions. In Proc. 3tr Internat. Conf. on Genetic Algorithms, Los Altos, CA, Morgan Kaufmann Publishers. 1989.

4 Powell, D. and M.M. Skolnick. Using Genetic Algorithms in Engineering Design Optimization with Non-linear Constraints. In Proc. 5th Internat. Conf. on Genetic Algorithms, Los Altos, CA, Morgan Kaufmann Publishers, 1993.

5 Homaifar, A., S. H.-Y. Lai and X. Qi. Constrained Optimization via Genetic Algorithms. Simulation, 62: 242--254. 1994.

6 Joines, J.A. and C.R. Houck. On the Use of Non-Stationary Penalty Functions to Solve Nonlinear Constrained Optimization Problems with GAs. In Proc. Evolutionary Computation Conference, Orlando, June 1994.

7 Bean, J.C. and Hadj-Alouane. A dual genetic algorithm for bounded integer programs. Tech Rep. TR92-53. Dep of Industrial and Operations Engineering, The University of Michigan. 1992.

8 Le Riche, R. G., et al. A segregate genetic algorithm for constrained structural optimization. In L.J.Eshelman (ED.), Proc. 6th Internet. Conf. On Genetic Algorithms, pp. 558-565, 1995.

9 Schoenauer, M., and S. Xanthakis (1993). Constrained GA Optimization. In Proc. 5th Internat. Conf. on Genetic Algorithms, Los Altos, CA, Morgan Kaufmann Publishers, 1993.

10 Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs. 3rd Ed. Springer-Verlag, New York. 1996.

11 Laguna, M. Metaheuristic Optimization with Evolver, Genocop and OptQuest EURO/INFORMS Joint International Meeting Plenaries and Tutorials, J. Barcelo (Ed.), pp. 141-150. 1997.

12 Lorena, L. A. N. and Furtado, J. C. Constructive Genetic Algorithm for Clustering Problems. Evolutionary Computation 9(3):309-327, 2001.

13 Oliveira, A. C. M. Algoritmos Evolutivos para Problemas de Otimização Numérica com Variáveis Reais – Monografia de Exame de qualificação – CAP/INPE, 2001. Disponível em http://www.lac.inpe.br/~lorena/monografia-alexandre.pdf