Uma Técnica de Otimização Utilizando Algoritmos Genéticos para Encontrar Extremos de Funções

JOÃO AUGUSTO SOBRAL DA SILVA E LUIZ ANTONÔNIO RIBEIRO JUNIOR

Instituto de Física - Universidade de Brasília - UnB
Campus Universitário Darcy Ribeiro - Asa Norte 70919-970 Brasília / DF

Resumo

Neste trabalho foram estudadas técnicas heurísticas de otimização global utilizando algoritmos genéticos. Populações de indivíduos foram criadas e submetidas a operadores genéticos de forma a simular um processo de evolução natural, gerando indivíduos que se adequassem aos extremos das funções de avaliação.

Palavras-Chave: Algoritmos Genéticos; Otimização.

1 Introdução

Uma busca tem como principal objetivo encontrar a melhor solução dentre todas as soluções possíveis – o chamado espaço de soluções. Algoritmos genéticos são algoritmos de busca baseados nos mecanismos de seleção natural e genética, ou seja, em conceitos como de hereditariedade, reprodução e mutação[1]. Esta técnica quase não necessita de interferência humana e encontra soluções próximas das soluções ótimas, necessitando apenas de uma forma de avaliação do resultado. Além disto, este tipo de algoritmo não possui a necessidade de se utilizar a derivada da função avaliada, o que seria extremamente complicado ao se tratar de funções descontínuas em determinados pontos.

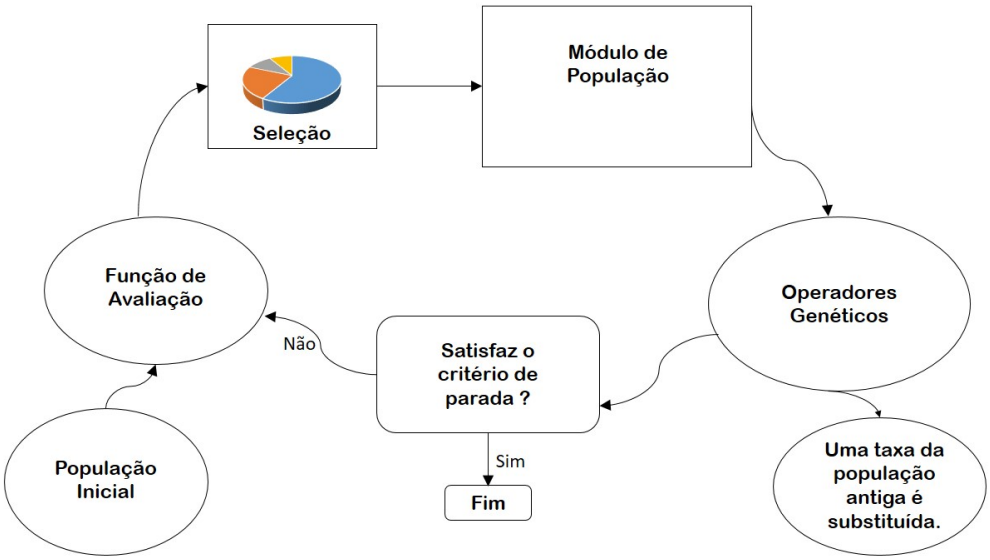

Figura 1: *Estrutura Básica de um Algoritmo Genético*

A figura 1 ilustra o funcionamento básico de um algoritmo genético. Inicialmente é gerada uma população com um determinado número de indivíduos e esses são avaliados através de uma função de avaliação. Em seguida, todos os indivíduos ganham uma probabilidade de serem selecionados para reproduzirem a partir de suas avaliações. Os indivíduos selecionados geram novos por um determinado método de reprodução. Visando uma maior variabilidade genética nas populações, operadores genéticos como os de mutação são aplicados em uma porcentagem da população. Após isto, uma taxa da população antiga é substituída pelos novos indivíduos, desses seleciona-se o melhor, que será utilizado como critério de parada caso seja encontrado, do contrário, todo o processo do loop é repetido.

Sendo assim, este projeto teve como objetivo o desenvolvimento de um algoritmo genético capaz de encontrar os extremos globais e locais de uma determinada função.

2 Seção Experimental

A primeira subrotina "População Inicial" foi criada através da função ran1(idum)[2], gerando uma certa quantidade de indivíduos em um intervalo específico. A função ran1 é responsável por gerar um número entre 0 e 1, para os efeitos do algoritmo a seguinte equação foi utilizada:

$$X(i) = ran1(idum) \cdot (IntF - IntI) + IntI \tag{1}$$

Onde IntF é último número e IntI o primeiro número do intervalo escolhido.

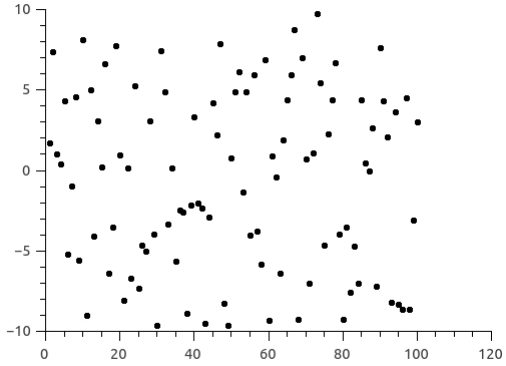

Figura 2: *100 Indivíduos gerados de forma aleatória entre -10 e 10 em uma função teste.*

Todos os indivíduos foram analisados pela função de avaliação. A subrotina "Ordenar" foi estruturada de modo a ordenar todas as avaliações e indivíduos de forma crescente utilizando o método de ordenação Bubblesort[2].

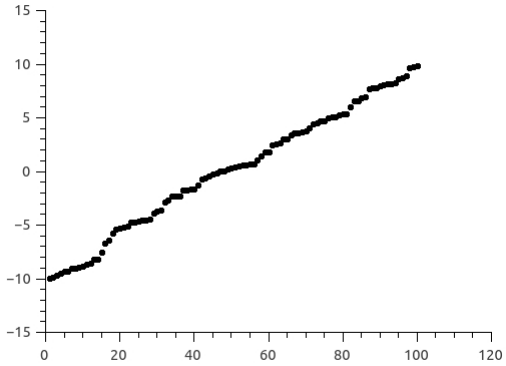

Figura 3: *Indivíduos ordenados pelo método Bubblesort.*

O método da roleta viciada atribui a cada indivíduo uma porcentagem de ser selecionado e parte de uma roleta virtual. Esta probabilidade pode ser obtida através da seguinte expressão:

$$prob(i) = \frac{Y(i)}{\sum_{i=1}^{n} Y(i)} \tag{2}$$

Onde prob(i) indica a probabilidade do indivíduo ser escolhido, Y(i) sua avaliação, $\sum_{i=1}^{n} Y(i)$ a soma de todas as avaliações e "n" o número de indivíduos em cada geração. Para indivíduos com uma avaliação negativa o termo no denominador da expressão em (2) não faria sentido. Uma subrotina chamada "Translação" foi criada com o intuito de evitar esse problema na seleção com a seguinte equação:

$$T(i) = Y(i) - Y(1) \tag{3}$$

Onde T(i) representa a avaliação do indivíduo transladada, Y(i) a avaliação do indivíduo e Y(1) a avaliação do primeiro indivíduo.

Como a subrotina "Ordenar" organizou todos os indivíduos e avaliações de forma crescente, o primeiro indivíduo possui a menor avaliação, portanto, cada indivíduo possui uma avaliação maior ou igual a 0 na subrotina "translação".

Fenômenos de convergência genética, ou convergência prematura, ocorrem quando o algoritmo converge de forma muito rápida para um máximo ou mínimo local. Para se evitar problemas como este, após transladadas, todas as avaliações foram enviadas para uma subrotina chamada "Linearização", onde uma nova espécie de função de avaliação foi criada de modo a evitar que os melhores indivíduos fossem sempre selecionados.

Os indivíduos foram então enviados para a subrotina "seleção" onde o método da roleta viciada [1] foi utilizado.

Os pais escolhidos para reprodução foram enviados para a subrotina "Reprodução" onde o método Gaussian Random Walk[3] foi utilizado. O método random walk consiste em uma caminhada aleatória dentre um intervalo especificado - no caso os dois pais escolhidos - o último indivíduo da caminhada é sempre o escolhido como um dos filhos. Para um conjunto de passos e de indivíduos muito grande na caminhada, o gráfico deste método se aproxima ao de uma distribuição normal. Quanto maior a caminhada, maior a largura da distribuição.

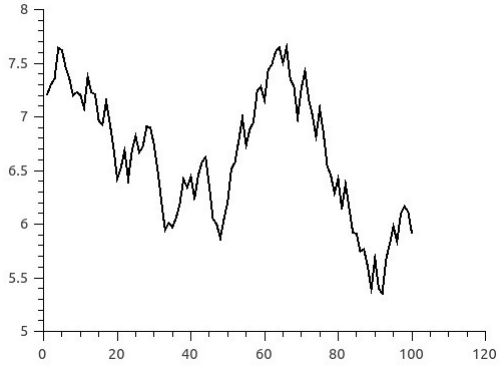

Figura 4: *Random Walk para 100 passos*

Na subrotina "Atribuição" os filhos gerados substituíram metade dos indivíduos da geração de seus pais, os indivíduos com a avaliação mais baixa. Utilizando a técnica de elitismo[1], o melhor indivíduo de cada geração permaneceu nas próximas gerações, tendo mais chances de ser escolhido para reproduzir.

Por fim, todos estes indivíduos sofreram uma espécie de mutação: 5 por cento dos indivíduos foram selecionados para sofrerem um acréscimo ou decréscimo infinitesimal utilizando o método random walk[3].

A figura 5 ilustra o funcionamento do programa como um todo, os loops foram efetuados até o critério de parada ser alcançado.

Função Teste	Máx. (Esperado)	Máx. (Encontrado)	Mín. (Esperado)	Mín.o (Encontrado)
x^2	-	-	0	0
$x^3 - 9x$	-1,73205080756	-1,73205081221	1,73205080756	1,73205080605
$e^{cos(x)}$	6,28318530717	6,28318530307	3,14159265358	3,14159265207

Tabela 1: *Resultados do Algoritmo para algumas funções teste*

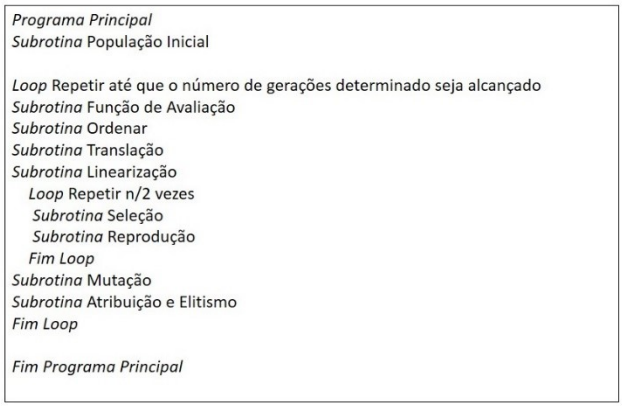

Figura 5: *Estrutura Principal do Algoritmo*

3 Resultados e Discussão

O algoritmo teve resultados satisfatórios para as 3 funções teste, como demonstra a tabela 1. Os valores encontrados para os extremos nas funções teste 2 e 3 tiveram uma precisão de pelo menos 7 casas decimais. O tempo de execução do pior caso foi extremamente baixo, portanto, o algoritmo consegue desempenhar o papel pelo qual foi desenvolvido de forma tangível.

A função teste 2 possui um máximo em -1,73 e um mínimo em 1,73. Após este ponto a função cresce para infinito positivo. Ao encontrar o ponto de máximo da função o algoritmo continuou procurando por outros candidatos a máximo, ou seja, os valores de x cresceram cada vez mais, tendendo ao infinito. Em casos de convergência genética o algoritmo ficaria preso nesse máximo local.

A função teste 3 apresenta um comportamento periódico. Os valores apresentados foram calculados somente para valores maiores que 0. Em outros testes, máximos e mínimos para x negativo foram encontrados da mesma forma.

4 Conclusão

Todas as soluções encontradas são satisfatórias ou estão muito perto das soluções ótimas. Para todos os resultados o tempo de execução se encontrou dentro do intervalo esperado. Os objetivos do projeto

foram alcançados tendo em vista que o algoritmo respondeu bem a diversas funções teste. Após mais testes e aperfeiçoamentos o algoritmo será utilizado para estudar espectros de sistemas diatômicos.

Agradecimentos

Os autores são gratos pelo suporte dado pelo CNPq e pela FAP-DF.

5 Referências

[1] R. Linden, Algoritmos Genéticos, 2006.

[2] S. Teukolsky, W. Vetterling, W. Press, B. Flan-nery, Numerical Recipes: Example Book (FORTRAN), 1986.

[3] S. Luke, Essentials of Metaheuristics, 2015.