ALGORITMO GENÉTICO APLICADO AO PROBLEMA DE OTIMIZAÇÃO DE FUNÇÕES

BEZERRA, Lucas M. C[1]; LOPES, Lais C. R. S.[2]; SILVA, Wilton R.[3]; LOPES, Allan K.[4]

[1] Estudante de Iniciação Científica – Instituto Federal Goiano – Câmpus Posse - GO. lucasbezerra.ks@hotmail.com; [2] Orientador – Instituto Federal Goiano – Câmpus Posse - GO. lais.lopes@ifgoiano.edu.br; [3] Colaborador – Instituto Federal Goiano – Câmpus Iporá - GO. wilton.silva@ifgoiano.edu.br; [4] Colaborador – Instituto Federal Goiano – Câmpus Posse - GO. allankardec.ti@gmail.com.

RESUMO: O presente trabalho de pesquisa tem por objetivo demonstrar a implementação de um algoritmo genético para otimização de funções. Pretende-se conhecer como estes algoritmos trabalham e para isso todos os passos de sua construção serão explicados detalhadamente. Para verificar o comportamento do algoritmo genético aplicado a esta problemática, alguns resultados foram gerados demonstrando a eficiência do método proposto. Futuramente será implementado um algoritmo genético para resolver uma instância do problema de escalonamento de tarefas conhecida na literatura como Gauss 18. Serão realizados experimentos para demonstrar a eficiência, ou não, do algoritmo genético na busca por soluções ótimas ou sub-ótimas para esta instância do problema que reduzam o tempo de execução, conhecido como *makespan*.

Palavras-chave: Algoritmo Genético. Otimização de Função. *Makespan*.

INTRODUÇÃO

O algoritmo genético é um mecanismo de busca baseados nos processos de seleção natural da luta pela vida e da genética de populações descrito por Darwin (1982). Trata-se de um método pseudoaleatório. Portanto, pode-se dizer que é um procedimento de exploração inteligente no espaço de parâmetros codificados.

John Holland (1985), na Universidade de Michigan, iniciou o estudo do algoritmo genético. Este procedimento pode ser aplicado a diversos tipos de problemas como, por exemplo, otimização de funções, roteamento, otimização combinatória, escalonamento de tarefas, *timetabling*, etc. A otimização é utilizada quando se deseja maximizar, ou minimizar, uma função. Isto é, encontrar os valores estacionários da função. A busca pelo máximo ou mínimo deve ser feita no domínio da função que representa o conjunto que contém todos os elementos para os quais a função é definida.

Técnicas clássicas de otimização são confiáveis e possuem aplicações nos mais diferentes campos de engenharia e de outras ciências. Porém, estas técnicas podem apresentar algumas dificuldades numéricas e problemas de robustez. Assim, os estudos de métodos heurísticos, com busca randômica controlada por critérios probabilísticos, reapareceram como uma forte tendência nos últimos anos, principalmente devido ao avanço dos recursos computacionais, pois um fator limitante destes métodos é a necessidade de um número elevado de avaliações da função objetivo (MICHELI, 1994).

O objetivo neste trabalho é implementar e utilizar um algoritmo genético para otimizar funções.

MATERIAIS E MÉTODOS

A função a ser otimizada é a seguinte: $f(x)=-|x\sin(\sqrt{|x|})|$. O problema de minimização de função pode ser modelado da seguinte maneira: dada a função $f(x)$ e o conjunto $D \in \mathbb{R}^n$ encontrar x' de modo que $f(x')=min\{f(x) \mid \forall x \in D\}$ onde $x=(x_1,x_2,\ldots,x_n)^T$. Pretende-se encontrar x' no intervalo [0,512] que minimize f. Como $f(x)$ é simétrico, estudar a porção positiva do eixo x será suficiente.

A população inicial foi formada aleatoriamente, por 50 pontos no intervalo [0,512]. Um indivíduo nada mais é do que um valor real da variável x. *Strings* binárias de 10 bits foram utilizadas para representar os valores de x. As strings (0000000000) e (1111111111) representam os limites do espaço de busca. Todas as outras strings foram mapeadas para o interior do espaço de busca. O mapeamento de uma *string* binária para um número real é feito da seguinte maneira, primeiro a *string* é convertida para um número decimal e só depois para o correspondente real x.

O *fitness* de cada ponto x é simplesmente o valor da função naquele ponto, ou seja, para encontrar o *fitness* de um ponto x qualquer, basta substituir na função a variável x pelo seu valor. O método de seleção utilizado é o da roleta, em que os melhores indivíduos têm mais alta probabilidade de serem selecionados para constituírem a nova geração. Uma vez encontrado o *fitness* fi de cada indivíduo de uma população, realizou-se a soma do *fitness* de todos os indivíduo da seguinte forma:

$$S = \sum_{i=1}^{popsize} fi,$$

Para encontrar a probabilidade pi de cada indivíduo vir a ser selecionado o *fitness* do indivíduo foi dividido por S :

$$pi = \frac{fi}{S}$$

Finalmente, uma probabilidade cumulativa para cada membro da população foi obtida da seguinte forma:

$$ci = \sum_{k=1}^{i} pk,\ i = 1,2,\ldots, popsize$$

A escolha de cada elemento para formar a nova população se deu obedecendo a seguinte condição: $c_i - 1 < r \leqslant c_i$, onde r é um número aleatório no intervalo [0,1]. Uma vez que a nova população foi produzida, as novas *strings* são aleatoriamente selecionadas em duplas e recombinadas através do *crossover* descrito em (TOMASSINI et al., 1995). Os dois novos indivíduos gerados compõem a nova população no lugar de seus parentes. O *crossover* deve ser aplicado de acordo com a taxa de probabilidade de cruzamento pc , onde $0 \leq pc \leq 1$. Após a aplicação do operador de cruzamento, foi aplicado o de mutação com frequência pm aos membros da população, onde $0.001 \leq pm \leq 0.1$. Esse operador foi aplicado a todos os bits de um indivíduo e a todos os indivíduos da população. Em ambos os operadores (*crossover* e mutação), foi gerado um número aleatório no intervalo [0,1] que foi comparado com pc ou pm para decidir se o operador seria aplicado ou não. O critério de parada adotado neste algoritmo genético é uma quantidade z de gerações que é definido pelo usuário durante a execução do sistema, conforme Figura 1.

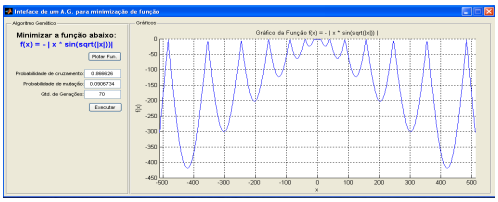

Figura 1 – Interface de utilização do algoritmo genético para minimização de função.

RESULTADOS E DISCUSSÃO

Como medida de qualidade da busca, foi utilizada a média do *fitness* da população durante a geração e o *fitness* do melhor indivíduo da geração. Observou-se que rapidamente o algoritmo se aproxima do mínimo, por volta da 10ª geração conforme Tabela 1 a).

Notou-se, após algumas execuções do algoritmo, que a melhor solução nem sempre estava na última geração, conforme Tabela 1 b). Observou-se que o melhor indivíduo estava na geração 60. Notou-se ainda que todos os pontos nesta tabela estão em torno do mínimo absoluto $(x = 421.0, f(x) = 0.041898276401614)$.

Tabela 1 – a) Qualidade da busca b) Melhores resultados.

a)

Geração	Melhor	Média
1	0.0202	0.0072
3	0.0367	0.0144
10	0.0419	0.0300
18	0.0419	0.0268
26	0.0419	0.0308
36	0.0419	0.0314
50	0.0419	0.0255
70	0.0419	0.0292

b)

X	F(X)	Geração
420.911	0.0419	34
420.911	0.0419	35
420.911	0.0419	40
420.911	0.0419	41
420.911	0.0419	48
420.911	0.0419	50
420.911	0.0419	53
420.911	0.0419	54
420.911	0.0419	58
420.911	0.0419	60

CONCLUSÃO

O presente artigo apresentou uma forma alternativa de encontrar o ponto mínimo de uma função. O algoritmo genético demonstrou ser uma poderosa ferramenta e apresentou ótimos resultados para o problema estudado. Por ser um método estocástico a sua performance varia de execução para execução, a menos que os mesmos números aleatórios gerados sejam usados.

REFERÊNCIAS BIBLIOGRÁFICAS

DARWIN, C. **The origin of species by means of natural selection**. 1872.

HOLLAND, J. H. **Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence**. U Michigan Press. 1975.

MICHELI, Giovanni de. **Synthesis and optimization of digital circuits**. McGraw-Hill Higher Education. 1994.

TOMASSINI, M., CALCOLO, C. S. D. **A Survey of Genetic Algorithms**. Vol. 3. Annual Reviews of Computational Physics: World Scientific, 1995.