Comparação entre Algoritmos Genéticos e Estratégias Evolutivas Aplicados na Função de Griewank

Diego Américo Guedes e Marcelo de Castro Cardoso
{diegoamericoguedes,marcelodcc}@gmail.com
Instituto de Informática – Universidade Federal de Goiás

***Resumo* — Este é um relatório técnico que mostra um comparativo entre *Algoritmos Genéticos* e *Estratégias Evolutivas* aplicados *a função de Griewank*. Foi possível avaliar a vantagem das Estratégias Evolutivas quanto a velocidade para encontrar mínimos globais, entretanto, é possível perceber que Algoritmos Genéticos conseguem bons resultados mediante ajustes de parametrização.**

Palavras-chave—metaheurísticas, algoritmos genéticos, estratégias evolutivas, função de griewank

I. Introdução

Algoritmos genéticos (AG) é uma técnica de programação que foi baseada na evolução biológica [1] e são comumente usados para resolver problemas de otimização. Além disso, é uma ótima estratégia para se resolver problemas para o qual pouco se sabe [2]. Algoritmos genéticos usam o princípio da natureza de seleção e evolução para produzir melhores soluções para um dado problema. A entrada do algoritmo é população inicial que resolve o problema e uma função de *fitness* ou aptidão que permite quantificar cada indivíduo da população. Indivíduos dessa população são selecionados seguindo algum critério para então serem submetidos a um processo de cruzamento (*crossover*). Os descendentes desse cruzamento são submetidos a um processo de mutação. Essa nova população é então submetida a um novo processo de avaliação segundo a função de aptidão. Esse processo é repetido até uma condição de parada como, por exemplo, alcance do valor ótimo. Na Figura 1 é mostrado um fluxograma do algoritmo genético.

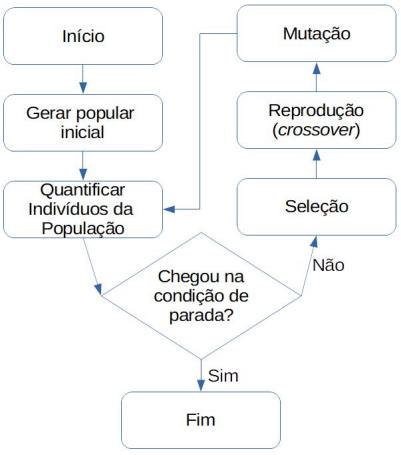

Fig. 1. Fluxograma Algoritmos Genéticos

Os primeiros esforços em Estratégias Evolutivas (EE) iniciaram em 1964 na Universidade de Berlin [3]. Assim como os AGs, as Estratégias Evolutivas [3][4] se baseiam na evolução biológica como método para resolver problemas de otimização. A primeira estratégia que foi utilizada é a chamada Estratégia Evolutiva de dois membros (1+1)-EE, que consiste numa população de um único indivíduo e neste é aplicado o operador genético de mutação. Assim, haverá um único pai e um único filho, por isso (1+1). Para introduzir o conceito de mais indivíduos, Rechenberg [3] propôs a Estratégia Evolutiva com multimembros, onde $\mu > 1$ pais podem participar na geração de um único indivíduo filho, denotado por (μ+1)-ES. Uma extensão é a (μ,λ)-ES [5], onde μ pais produzem λ filhos, onde somente os filhos são submetidos a seleção, ou seja, o tempo de vida de cada indivíduo é limitado a 1 geração.

Uma função de *benchmark* que é comumente utilizada para avaliar a performance de diferentes metaheurísticas para encontrar ótimos locais é a função de Griewank [6] definida na Tabela 1.

Tabela 1. Definição da função de Griewank [7]

Tipo	Minimização
Intervalo	$x_i \in [-600, 600]$
Ótimo Global	$x_i = 0, \forall i \in \{1 \dots N\},\ f(\mathbf{x}) = 0$
Função	$f(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$

A Figura 2 apresenta graficamente a função de Griewank com valores de $20 \le x_i \le 20$. Observa-se que a função tem um aspecto bastante interessante por possuir diversos máximos e mínimos locais distribuídos de maneira quase uniforme. Ademais, os mínimos locais e o mínimo global tem valores muito próximos, o que dificulta a estratégia de busca do mínimo global.

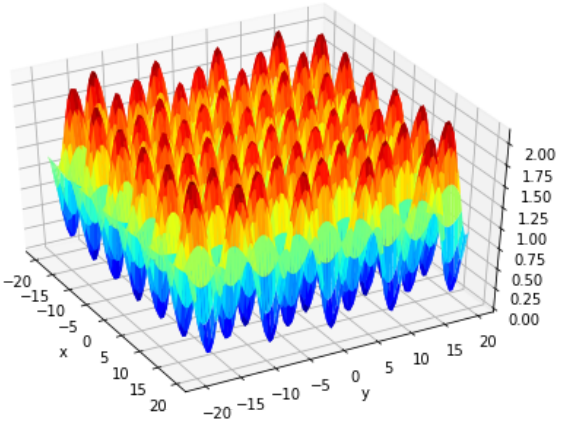

Fig. 2. Visão em 3 dimensões da função de Griewank [8]

Com o objetivo de comparar o comportamento de algoritmos genéticos e estratégias evolutivas, realizou-se a análise desses dois métodos aplicados a função de Griewank. Para os algoritmos genéticos, houve dois experimentos, um com indivíduos modelados com valores binários e outro com indivíduos modelados com valores reais.

II. Metodologia dos Experimentos

Os códigos fontes encontram-se no github [8] e foram escritos na linguagem de programação python utilizando a plataforma Colab do Google [9]. A quantidade de variáveis escolhida como parâmetro para a função de Griewank é 10. Pode-se definir os parâmetros x na função de Griewank como:

$$x = [x_1, .., x_i],\ tal\ que,\ 1 \leq i \leq 10\ e\ -20 \leq x_i \leq 20$$

Por se tratarem de metaheurísticas, algoritmos não determinísticos, os testes foram executados em 30 rodadas. Os resultados que serão apresentados na próxima seção são resultantes da execução dessas 30 rodadas para cada algoritmo.

Em cada rodada foram produzidas 300 gerações. Em cada geração, foi gerada uma população com 300 indivíduos. Assim, em cada rodada, foram analisados 90000 indivíduos. Esses mesmos parâmetros foram utilizados em todos os experimentos e com todos os algoritmos avaliados.

III. Experimentação e Análise de Resultados

Para a técnica de algoritmos genéticos, foi utilizado a representação binária e a representação de ponto flutuante ou real. Para a representação binária, como o valor máximo de x_i utilizando na função de Griewank é 20 (20 = 10100 em binário), foi utilizado um binário com 11 bits, sendo 1 bit para sinal (positivo ou negativo), 5 bits para a parte inteira e 5 bits para a casa decimal. O *Algoritmo 1* foi o algoritmo utilizado na conversão do array binário em ponto flutuante. Por exemplo, para o indivíduo 1 00011 00110, o primeiro bit 1 é de sinal negativo, os bits 00011 representam 3 em decimal e os bits 00110 representam 6 em decimal. Logo, os 11 bits representam -3.06 em decimal.

Algoritmo 1. Converter array binário em Ponto Flutuante [8]

```
# CONVERTE ARRAY BINARIO EM INTEIRO
DEF BIN_TO_INT(B):
 RETURN INT(B, 2)

# CONVERTE ARRAY BINARIO EM PONTO FLUTUANTE
DEF BIN_TO_FLOAT(B, S):
 IF LEN(B) < S*2+1:
  RETURN 0
 RETURN FLOAT(
   ('-' IF B[0] == 1 ELSE '') +
   STR(BIN_TO_INT(''.JOIN(STR(E) FOR E IN B[1:S+1]))) +
   '.' +
   STR(BIN_TO_INT(''.JOIN(STR(E) FOR E IN
B[S+1:2*S+1]))).ZFILL(LEN(STR(POW(2,S-1)))) )
```

Nos três métodos comparados, foi utilizado a biblioteca deap [10] para o cálculo da função de Griewank. Além disso, a biblioteca foi usada para gerar os indivíduos e população, além de fazer a seleção, cruzamento e mutação.

Dentre as 30 rodadas do algoritmo genético com representação binária, houve um indivíduo que foi o melhor de todos, ou seja, 1 entre 2700000 indivíduos. Nessa execução, foram feitas 300 gerações. A Figura 3 apresenta o valor do *fitness* do melhor indivíduo em cada uma das 300 gerações. É possível observar que o *fitness* decai ao longo das gerações até chegar no valor ótimo de 0.0 .

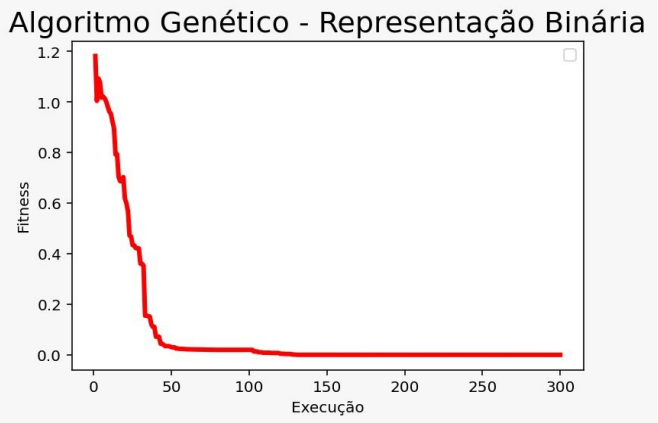

Fig. 3. *Fitness* do melhor indivíduo nas 300 gerações [8]

Para o algoritmo genético com representação real, também é possível analisar, conforme Figura 4, o valor do *fitness* do melhor indivíduo em cada uma das 300 gerações. Assim como a representação binária, na representação real o valor do *fitness* também decai ao longo das gerações. No entanto, o algoritmo não consegue chegar no valor ótimo de 0.0 no *fitness*, mas chega em um valor bem próximo de 0.00740 .

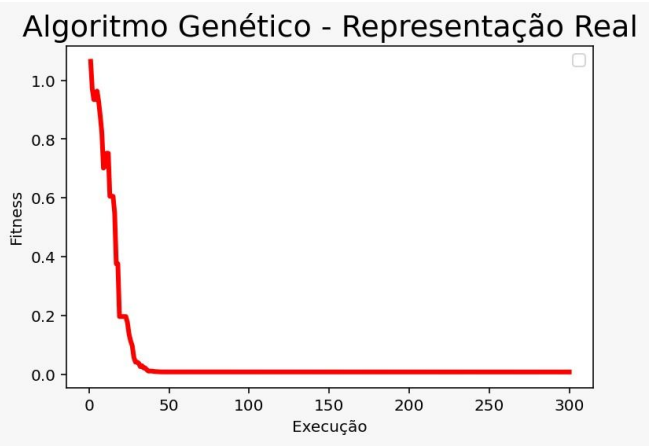

Fig. 4. *Fitness* do melhor indivíduo nas 300 gerações [8]

Para a Estratégia Evolutiva, foi utilizada a Estratégia Evolutiva Multimembro (μ,λ). Conforme recomendação dada em sala de aula, foram utilizados os valores de μ=35 e λ=265, a fim de gerar uma população de 300 indivíduos. A Figura 5 apresenta o valor do *fitness* do melhor indivíduo em cada uma das 300 gerações. Assim como os métodos de algoritmos genéticos, o valor do *fitness* decai ao longo das gerações. No entanto, observa-se que o decaimento é mais acelerado e chega-se no valor ótimo de 0.0 por volta da 20º geração, enquanto a representação binária e representação real chegaram por volta da 130º e 40º geração, respectivamente.

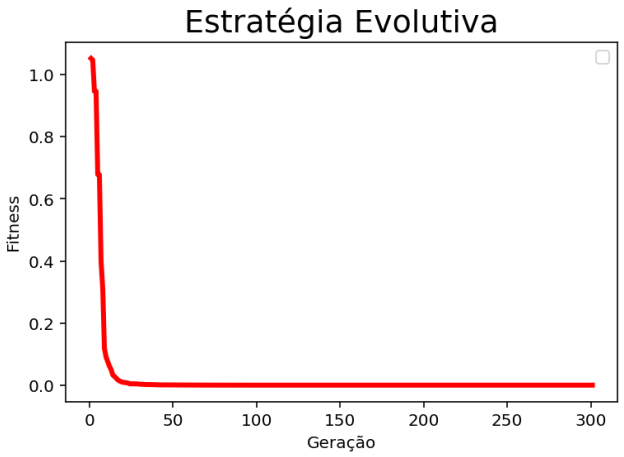

Fig. 5. *Fitness* do melhor indivíduo nas 300 gerações [8]

A Figura 6 apresenta o comparativo entre os 3 métodos utilizando o menor *fitness* de cada uma das 30 execuções. Observa-se que os Algoritmos Genéticos foram os que apresentaram a menor média e menor desvio padrão. Além disso, a maior diferença entre os *fitness* mínimo e máximo foi na Estratégia Evolutiva. Por fim, o AG Binário e a Estratégia Evolutiva conseguiram atingir o mínimo global de 0.0 e o AG Real não.

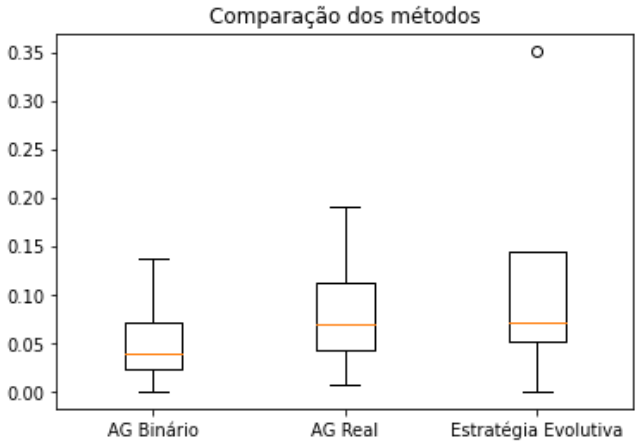

Fig. 6. Comparativo dos métodos [8]

IV. Conclusão e Considerações Finais

Os resultados encontrados mostram que, apesar da função de Griewank apresentar diversos ótimos e mínimos globais, é possível resolvê-lo de forma satisfatória em um tempo computacional aceitável utilizando as metaheurísticas Algoritmos Genéticos e Estratégias Evolutivas. Para os parâmetros utilizados nos experimentos, a Estratégia Evolutiva mostrou-se uma melhor metaheurística no caso geral, pois possui bons resultados em sua função de *fitness* e um tempo melhor de convergência para o valor ótimo.

Em termos de confiança na convergência, para os parâmetros utilizados nos experimentos, o GA Binário apresentou um resultado geral melhor, se aproximando do ótimo mais facilmente.

Alguns experimentos, que não fazem parte da metodologia de análise, foram realizados variando alguns parâmetros pré-definidos como, por exemplo, diminuindo o número de variáveis de 10 para 1 nos AGs. Estes passam a convergir rapidamente para o ótimo global 0.0 da função em poucas gerações, mesmo aumentando os valores limites da função de Griewank para o máximo (-600,600) e a quantidade bits da representação de 11 para 21.

Com esses experimentos, foi possível demonstrar que as Estratégias Evolutivas apresentam uma convergência mais rápida para o ótimo global da função avaliada, assim como apontado nos textos de referência [3][4], entretanto, há métodos, como os AGs, que podem ter convergências mais confiáveis ajustando os parâmetros.

Avaliações com outras funções de *benchmark* se fazem necessárias para comprovar as conclusões evidenciadas por estes experimentos.

Referências

[1] Mitchell, Melanie. An Introduction to Genetic Algorithms. MIT Press, 1996.

[2] M. Srinivas and L. M. Patnaik, "Genetic algorithms: a survey," in Computer, vol. 27, no. 6, pp. 17-26, June 1994, doi: 10.1109/2.294849.

[3] Ingo Rechenber. Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog Verlag, Stuttgart, 1973

[4] Hans-Paul Schwefel, Numerische Optimierung von Computer-Modellen, PhD thesis, 1974. Reprinted by Birkhäuser, 1977.

[5] SCHWEFEL, H.-P. (1995), Evolution and Optimum Seeking, Wiley, New York.

[6] Huidae Cho, Francisco Olivera, Seth D. Guikema. A derivation of the number of minima of the Griewank function. Applied Mathematics and Computation Volume 204, Issue 2, 15 October 2008, Pages 694-701.

[7] Deap, Grieank function. Acessado em: Agosto 16. 2020. [Online]. Disponível: https://deap.readthedocs.io/en/master/api/benchmarks.html#deap.benchmarks.griewank

[8] Diego Américo Guedes, Github – Hospedagem de código-fonte, Agosto 2020. Acessado em: Agosto 14, 2020. [Online]. Disponível: https://github.com/diegoguedes/metaheuristicas/tree/master/algoritmos_geneticos_x_estrategia_evolutiva

[9] Google, Colab – Texto com códigos executáveis, Jul. 2020. Acessado em: Jul. 10, 2020. [Online]. Disponível: https://colab.research.google.com/

[10] DEAP – Distributed Evolutionary Algorithm in Python, Agosto 2020. Acessado em: Agosto 14, 2020. [Online]. Disponível: https://deap.readthedocs.io/