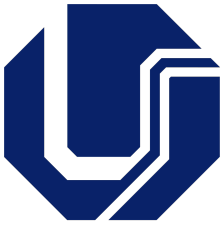


Universidade Federal de Uberlândia
Pós-Graduação em Engenharia Elétrica
Disciplina: Algoritmos Genéticos

LUIZ ARTHUR TARRALO PASSATUTO

TRABALHO 2
ALGORITMOS GENÉTICOS
MINIMIZAÇÃO DE FUNÇÃO PELO MÉTODO DA ROLETA

Uberlândia
2023

1 Introdução

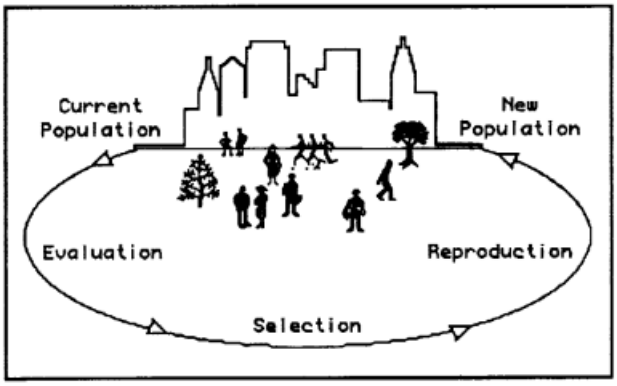
Este trabalho se propõe a fazer uma breve explicação do artigo publicado pela *World Scientific* de autoria de M. Tomassini em 1995 (TOMASSINI, 1995). O artigo trata de um tema relevante a época e ainda muito utilizado nos dias atuais, os *Evolutionary Algorithms* ou também conhecidos como *Genetic Algorithms*, em português, *Algoritmos Genéticos* (AG).

Os primeiros AGs foram descritos por John Holland nos anos 60, sendo melhor desenvolvidos por seus alunos e colegas da Universidade de Michigan nas décadas seguintes (MITCHELL, 1995). Tais métodos envolvem uma procedimentos de busca e otimização de resultados baseados no funcionamento do mundo biológico, ou seja, na teoria da evolução de Darwin. Seria errôneo dizer que as AGs são totalmente fiéis a teoria da evolução, visto que esta levou milhões de anos para se provar algo natural ao mundo, enquanto o desenvolvimento de um algoritmo ou implementação da ideia leva no máximo alguns meses.

Algoritmos Genéticos são um termo generalista que envolve diversas metodologias relacionadas mas não identicas com ideias inspiradas nos mais variados estudos que permeiam a evolução biológica, através dos métodos de evolução e seleção (TOMASSINI, 1995).

O modelo mais básico de AG pode ser visto na Figura 1, a qual demonstra uma população inicial, consistindo de potênciais soluções, geradas de forma randômica ou heurísticamente. No AG clássico, cada membro é representado por uma codificação binária com número fixo de *bits* que formaram o chamado *cromossomo*. Tal codificação precisa ser descodificada para que tenhamos o valor real do indivíduo. A partir disso, teremos a fase de avaliação, selecionando os melhores indivíduos de acordo com seu *fitness* ou, em paralelo biológico, valor de adaptabilidade ao sistema. Os indivíduos mais adaptados terão a maior probabilidade de darem origem a uma nova geração, por meio do *crossover*, ou ao menos continuarem na população da próxima geração (WANG; BAYER, 1991).

Figura 1: Modelo de Algoritmo Genético de Iteração.

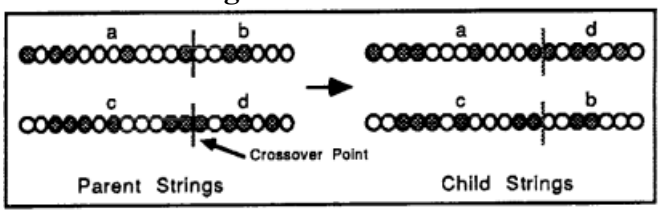


Fonte: Wang e Bayer (1991)

A fase de reprodução consiste na seleção aleatória de dois indivíduos que já foram considerados aptos por meio da avaliação do seu *fitness* e tendo seus cromossomos recombinados de

modo que possam gerar dois novos indivíduos. O processo é repetido até que uma nova população seja gerada. Tal operação pode ser melhor compreendida pela representação na Figura 1.

Figura 2: *Crossover*.



Fonte: Wang e Bayer (1991)

Por fim, antes de darmos seguimento a uma nova geração, pode-se determinar uma taxa de mutação que irá afetar esses novos indivíduos. A mutação afeta diretamente os genes do indivíduo, trocando aleatoriamente valores que compõem o cromossomo do mesmo. Tal operação garante maior variabilidade as possíveis soluções, mas pode afetar indivíduos com ótimo *fitness* os afastando do resultado esperado, por isso ela deve ser manipulada de maneira atenta.

Com isso temos uma explicação breve sobre os AGs e podemos introduzir o problema aqui proposto.

2 *A Simple Example*: o problema proposto

O exemplo apresentado por Tomassini (1995) como exemplo básico a apresentação do funcionamento das AGs é uma função matemática simples, onde devemos determinar seu mínimo global em um dado intervalo definido para x. Logicamente tal problema pode ser resolvido de forma manual e ter resultados satisfatórios em minutos, mas exatamente por isso fica fácil comprovar a aplicabilidade de um AG.

Considere a Equação 1. O exemplo envolve determinar um valor de x^* para o intervalo [0,512] que minimiza a função f. Como se trata de uma função simétrica, o autor propõe o estudo apenas do intervalo positivo.

$$f(x) = -|x \cdot sin(\sqrt{|x|})| \tag{1}$$

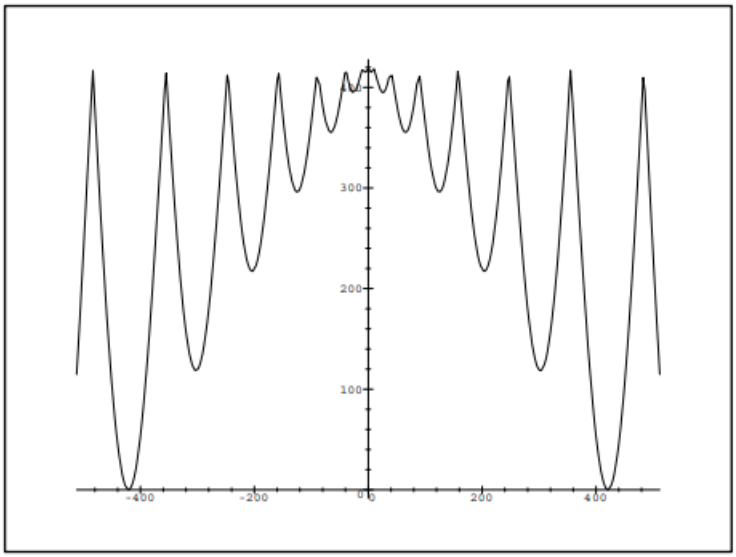
Para desenvolvimento deste AG alguns outros pontos devem ser destacados. A população inicial é formada por 50 indivíduos randômicos presentes no intervalo de [0,512]. A representação de cada possível solução, ou seja x, é feita por meio de uma sequência de binários formados em 10 *bits* compondo cerca de 1024 valores diferentes no intervalo descrito, nos garantindo assim um intervalo de 0,5 entre cada possível valor de x.

Logo a sequência (0000000000) e (1111111111) representam, respectivamente, o mínimo e máximo do intervalo de busca. A decodificação é feita do binário ao inteiro, para então

verificação de x na função. O *fitness* de cada indivíduo é basicamente o retorno da função f para aquele determinado x, ou seja, quanto menor o retorno da função mais adaptado será o indivíduo.

A saída da Equação 1 pode ser visto na Figura 2.

Figura 3: *Gráfico de f(x).*



Fonte: Tomassini (1995)

Para seleção dos indivíduos para reprodução foi utilizada a seleção *fitness-proportionate*. O primieiro passo é realizar a soma do *fitness* dos indivíduos da geração atual, conforme a Equação 2, onde S é o *fitness* total da população. Então a probabilidade de um indivíduo ser selecionado é dada pela Equação 3. Por fim temos a definição de uma probabilidade cumulativa para indivíduo, onde o *fitness* é somado ao resultado obtido pelos membros da geração anterior, conforme a Equação 4.

$$S = \sum_{i=1}^{popsize} f_i \tag{2}$$

$$p_i = \frac{f_i}{S} \tag{3}$$

$$c_i = \sum_{k=1}^{i} p_k, i = 1,2,...,popsize \tag{4}$$

2.1 Seleção pelo Método da Roleta

O método de seleção utilizado para este problema é conhecido como *Seleção pela Roleta*. Basicamente a probabilidade cumulativa vista na Equação 4 é utilizada como um identificador para uma das possíveis soluções.

Então um número gerado randomicante, entre 0 e 1, sendo comparado a probabilidade do indivíduo estudado para seleção. Se o número estiver entre o valor obtido da geração anterior e da atual, então o indivíduo da geração atual é selecionado e continuará para geração posterior (YU et al., 2016).

O uso da seleção pela roleta limita o AG a um máximo, já que a função de avaliação sempre mapeia as soluções dentro de conjunto determinado (PENCHEVA; ATANASSOV; SHANNON, 2009).

Citando um exemplo simples, imagine que há apenas três indivíduos onde o valor de probabilidade p_i para cada um é respectivamente: $p_1 = 0.2$, $p_2 = 0.1$ e $p_3 = 0.3$. E temos que: $c_1 = 0.5$, $c_2 = 1.0$ e $c_3 = 0.3$. Caso o número aleatório gerado entre [0,1] seja 0,25, como $0{,}25 < c_3$, o indivíduo 3 será selecionado. Caso o número fosse maior, como 0,9, então seria o indivíduo 2, já que $c_2 < 0.9$.

Em resumo, quanto mais apto o indivíduo, proporcionalmente maiores as chances de ele se reproduzir e suas sequências serão selecionadas mais de uma vez. Lembrando que para tal método funcionar, os valores de *fitness* precisam ser positivos como estamos trabalhando com probabilidade.

2.2 Cruzamento e Mutação

Assim que uma nova geração nasce, os indivíduos foram pares e poderão ser recombinados por meio do *crossover*, como já explicado anteriormente. Os invidíduos gerados a partir daí substituirão seus pais na nova geração. A frequência de ocorrência do *crossover* é dada pela *taxa de crossover* p_c, um valor compreendido no invervalo [0,1], onde comumente se usa o valor 0,6.

Após o *crossover* os indivíduos podem sofrer também, como já explicado, a mutação, numa frequência dada por p_m que geralmente fica entre 0,01. Conforme Tomassini (1995) a taxa de mutação dos genes (ou bits) de um indivíduo é dada da seguinte forma: para cada membro da população é gerado um número aleatório r no intervalo [0,1], se $r \leq p_m$ o gene/bit será modificado.

3 Desenvolvimento e Resultados

3.1 Desenvolvimento do Algoritmo Genético

O código foi desenvolvido na linguagem *Python 3*, utilizando a IDE *Visual Studio Code*. A linguagem *python* é um método de programação amplamente conhecido e de fácil compreensão, sendo utilizado para diversas aplicações graças as suas bibliotecas e a facilidade em aplica-las.

O código desenvolvido para o AG de Seleção por Roleta contém comentários em todas as linhas, visando que qualquer pessoa possa interpretar o código facilmente. Para melhor funcionamento, o código foi desenvolvido por funções, mantendo uma estrutura coesa onde é possível manipular o passo a passo do algoritmo e podendo aplicar diferentes funções, ou alterando as variáveis de modo a aumentar a aplicabilidade do algoritmo desenvolvido.

O código em *.py* está contido em anexo junto a este relatório. Após sua execução, dois arquivos serão gerados. Sendo eles:

- um arquivo dados_geracao.xlsx contendo os dados populacionais por geração, com o melhor indivíduo da população e o *fitness* corresponde;
- um arquivo graf_f(x).png contendo o *plot* do gráfico da função f(x), indicando o mínimo global encontrado pelo algoritmo.

Como toda a explicação do código está presente dentro do próprio arquivo *.py* com comentários linha por linha, será destacado dentro deste relatório apenas os resultados obtidos.

AS variáveis de entrada foram todas baseadas no apresentado por Tomassini (1995), logo:

- quantidade de indivíduos na população: *pop_length = 50*;
- quantidade de bits em cada indivíduo: *ind_length = 10*;
- número de gerações/iterações: *num_gen = 100*;
- probabilidade de cruzamento: *pc = 0,6*;
- probabilidade de mutação: *pm = 0,01*;
- valor mínimo do indivíduo: *x_min = 0*;
- valor máximo do indivíduo: *x_max = 512*.

As bibliotecas utilizadas aqui não reduzem o código para resultados mais diretos, sem atalhos, na verdade elas foram implementadas para gerar resultados mais visuais ou construir variáveis como *dataframes* para melhor compreender as variáveis e ter saídas mais concisas.

3.2 Resultados obtidos

Lembrando que o AG varia os resultados para cada execução, mesmo que mantidas as variáveis. Primeiro iremos apresentar os melhor indivíduos por geração e seu respectivo *fitness*. Entretanto, os resultados apresentados por Tomassini (1995) estão incorretos, a Equação 1 deveria ter um retorno negativo, diferente do apresentado em seu texto.

A Tabela 1 apresenta os dados obtidos das 20 primeiras gerações para uma execução do algoritmo. A primeira nota a ser feita é a convergência do resultado, tendo que estamos tratando apenas de $\frac{1}{5}$ do total de gerações do algoritmo, podemos notar que o resultado converge a partir da 17ª geração, mesmo que o resultado já tenha sido obtido na 6ª geração.

Tabela 1: Dados obtidos por geração.

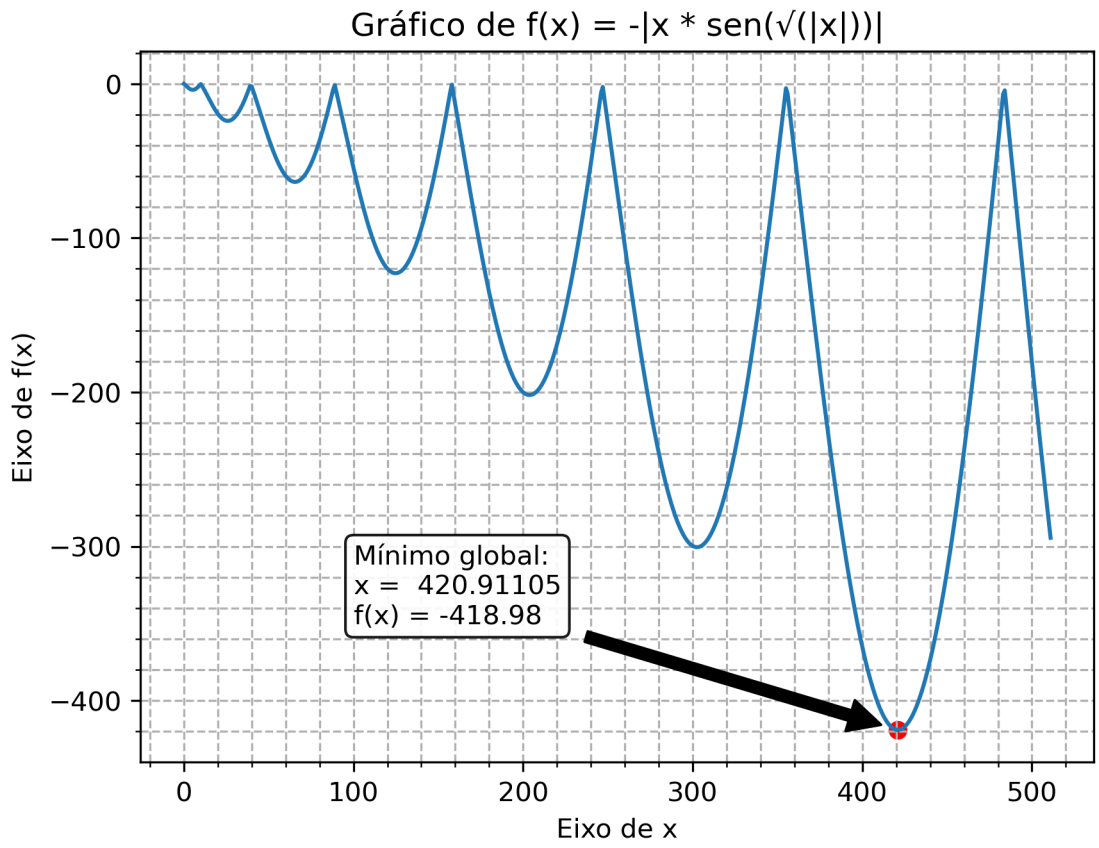
Geração	Melhor Indivíduo	*Fitness*
1	427,4174	-413,7334
2	426,9169	-414,5161
3	426,9169	-414,5161
4	426,9169	-414,5161
5	419,4096	-418,6764
6	420,911	-418,9825
7	420,911	-418,9825
8	420,911	-418,9825
9	420,911	-418,9825
10	420,911	-418,9825
11	420,911	-418,9825
12	419,9101	-418,8415
13	422,913	-418,5056
14	420,911	-418,9825
15	420,911	-418,9825
16	422,913	-418,5056
17	420,911	-418,9825
18	420,911	-418,9825
19	420,911	-418,9825
20	420,911	-418,9825

Fonte: o Autor

Podemos analisar que os resultados obtidos indicam a precisão do uso do AG pelo método da seleção da roleta, indicando uma convergência boa, mas que poderia ser melhorada com uso de outro método de seleção ou implementando métodos que não façam perder resultados ótimos como o obtido já na 6ª geração e depois voltando a outro x na 12ª geração. Mas, a partir da 17ª geração isso não ocorre mais, onde todos os resultados após são todos do mínimo global.

Já na Figura 3.2 temos o gráfico gerado pelo AG escrito. Primeiro já podemos destacar o resultado correto da função estudada, visto que os resultados de $f(x)$ são negativos de acordo com a equação apresentada e com destaque ao mínimo global de x, obtendo um resultado até mais preciso que o apresentado por Tomassini (1995).

Figura 4: ***Gráfico de f(x)*** **gerado pelo Algoritmo.**



Fonte: o Autor

4 Conclusão

A proposta deste trabalho foi estudar e desenvolver um AG pelo método de seleção da roleta para estudo do caso apresentado na seção *Simple Example* do artigo de Tomassini (1995), onde se desenvolve um algoritmo básico para encontrar o mínimo global para a Equação 1.

Com os resultados obtidos vê-se a qualidade em implementar um AG mesmo para casos simples, porém não podemos esquecer que a linguagem *Python* dispõe de bibliotecas e funções prontas disponibilizadas na internet que aplicam diretamente uma AG para solução de um problema. O código desenvolvido e anexado junto a este trabalho visa, de forma didática, indicar como um AG pode ser desenvolvido de maneira mais ”manual”.

Referências

MITCHELL, M. Genetic algorithms: An overview. In: CITESEER. *Complex.* [S.l.], 1995. v. 1, n. 1, p. 31–39.

PENCHEVA, T.; ATANASSOV, K.; SHANNON, A. Modelling of a roulette wheel selection operator in genetic algorithms using generalized nets. *International Journal Bioautomation*, Bulgarska Akademiya na Naukite/Bulgarian Academy of Sciences, v. 13, n. 4, p. 257, 2009.

TOMASSINI, M. A survey of genetic algorithms. *Annual reviews of computational physics III*, World Scientific, p. 87–118, 1995.

WANG, L.; BAYER, S. E. Genetic algorithms. *NASA, Washington, Technology 2000, Volume 2*, 1991.

YU, F. et al. Improved roulette wheel selection-based genetic algorithm for tsp. In: IEEE. *2016 International conference on network and information systems for computers (ICNISC)*. [S.l.], 2016. p. 151–154.