

Trabalho Computacional 1: Algoritmos Genéticos

Pedro Vinícius A. B. Venâncio

Resumo—Algoritmos Evolutivos (AE) foram propostos na literatura ao longo dos anos com o intuito de simplificar a formulação e solução de problemas de otimização a partir de técnicas inspiradas na biologia evolutiva. Uma classe popular desses algoritmos é composta pelos Algoritmos Genéticos (AG), cujo processo de evolução corresponde a uma busca no espaço de soluções do problema. Em dicotomia com os algoritmos tradicionais de otimização, os algoritmos genéticos trabalham com uma população de soluções e não uma única solução, viabilizando *trade-offs* importantes aos tomares de decisão em problemas de alta complexidade. Este trabalho propõe algoritmos genéticos para a resolução de dois problemas clássicos: (i) N-Rainhas e (ii) Minimização da Função Rastrigin. O primeiro consiste em um problema de otimização combinatória e o segundo em um problema contínuo com restrições de caixa, o que corrobora a versatilidade dos algoritmos genéticos.

Palavras-chave—Algoritmos genéticos, otimização combinatória, otimização contínua.

I. INTRODUÇÃO

Algoritmos Genéticos, em essência, partem de um princípio similar à Seleção Natural proposta por Charles Darwin e Alfred Wallace [1], onde indivíduos de uma população competem entre si por recursos limitados e os mais aptos sobrevivem. Por consequência, a reprodução dos sobreviventes tende a promover um desenvolvimento na população ao longo das gerações, fato que instigou diversos pesquisadores a utilizar tais conceitos de melhorias gradativas em algoritmos de otimização, bem como estabelecer analogias entre a biologia evolutiva e a computação.

Tendo em vista uma função (*fitness*) que permite mensurar a aptidão de uma solução (indivíduo), pode-se criar aleatoriamente, a priori, um conjunto de soluções candidatas (população). Conforme os valores de aptidão de cada indivíduo, os melhores são selecionados para semear a próxima geração segundo operadores de variação, dentre eles o cruzamento (*crossover*) e a mutação (*mutation*). O cruzamento, em geral, é conduzido a partir de dois indivíduos selecionados (pais) e produz outros dois novos indivíduos (filhos). A mutação, por sua vez, é aplicada a um candidato por vez, produzindo um indivíduo anômalo.

Por fim, esse conjunto de novos indivíduos têm sua aptidão avaliada para competir com os demais indivíduos por uma vaga na próxima geração mediante um mecanismo de seleção. Normalmente, esse processo iterativo é mantido até que uma solução candidata tenha qualidade próxima da desejada ou que um limite computacional previamente definido seja alcançado, por exemplo, número máximo de gerações ou de avaliações na função objetivo [2].

Este trabalho tem como finalidade desenvolver algoritmos genéticos para tratar o Problema das N-Rainhas e o Problema de Minimização da Função Rastrigin. Visto que ambos problemas são clássicos na literatura da Computação Evolucionária,

espera-se compreender todos os componentes necessários para composição de um algoritmo genético, tal como assimilar suas variantes e possibilidades de integração.

II. PROBLEMA DAS N-RAINHAS

O Problema das N-Rainhas consiste em dispor N rainhas em um tabuleiro de xadrez de dimensão $N \times N$ de modo que tais rainhas não sejam capazes de atacar umas às outras. Em outras palavras, deve haver apenas uma única rainha em cada linha, coluna ou diagonal do tabuleiro [3].

Devido à sua natureza combinatória, o Problema das N-Rainhas foi largamente utilizado como *benchmark* para novas técnicas de Inteligência Artificial (IA). Sua solução requer um algoritmo que seja tanto eficiente na exploração do espaço de decisão quanto capaz de satisfazer um conjunto de restrições, o que acabou fomentando a elaboração de soluções práticas nas áreas de controle de tráfego e criação de circuitos integrados [4].

O presente trabalho tem como objetivo implementar um algoritmo genético para resolução do Problema das N-Rainhas com $N = 8$, isto é, considerando a disposição de 8 rainhas em um tabuleiro de dimensão fixa 8×8 . Apesar da solução proposta ser generalizável para qualquer valor de N , a complexidade do problema é $O(n!)$ e, portanto, valores de N suficientemente grandes exigem um alto custo computacional.

III. ALGORITMO GENÉTICO PROPOSTO

O algoritmo proposto para o Problema das N-Rainhas é baseado em um algoritmo genético clássico para a resolução do *Travelling Salesman Problem* (TSP) [5], uma vez que ambos os problemas podem ser modelados como permutação de inteiros. Em sua configuração de modelo *Steady-State*, o algoritmo contempla os operadores de variação *cut-and-crossfill crossover* e *swap mutation*. Tanto a seleção dos indivíduos para a próxima geração quanto a seleção dos pais para cruzamento são proporcionais ao *fitness* das soluções, no entanto, a segunda ocorre apenas sobre uma amostra aleatória da população.

A. Representação da solução

O Problema das N-Rainhas pode ser formulado como um Problema de Satisfação de Restrições, onde cada uma das N variáveis assume um único valor e o número de variáveis é igual ao número de valores possíveis. Dessa forma, qualquer solução pode ser considerada como atribuir uma permutação às variáveis [3]. Quando uma permutação satisfaz todas as restrições, ela é considerada uma solução factível.

A representação do fenótipo pode ser dada trivialmente por uma configuração do tabuleiro de xadrez com N rainhas, ou seja, uma matriz de adjacências binária $\mathbf{f} \in \mathbb{B}^{N \times N}$,

onde $f_{ij} = 1$ indica que há uma rainha na posição (i, j) . A representação do genótipo, por sua vez, corresponde a permutação de inteiros $g = \{i_1, \dots, i_k, \dots, i_N\}$, onde o k -ésimo elemento de valor i_k indica que há uma rainha na posição (i_k, k) (Figura 1). É importante ressaltar que não houve a necessidade de implementação do fenótipo, uma vez que todos os operadores de variação e mecanismos de seleção são efetuados sobre o genótipo.

1	0	0	1	0	0	0	0	0
2	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0
4	0	0	0	0	1	0	0	0
5	0	0	0	0	0	0	0	1
6	0	1	0	0	0	0	0	0
7	0	0	0	1	0	0	0	0
8	0	0	0	0	0	1	0	0

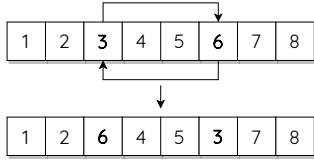


Figura 3. Exemplo de uma mutação pelo operador *swap mutation*. Nesse caso, as posições sorteadas são 3 e 6, cujos valores por coincidência são, respectivamente, 3 e 6. A mutação é consolidada após o valor 3 tomar a posição 6 e o valor 6 tomar a posição 3. Fonte: Elaborado pelo autor.

F. Critérios de parada

O processo iterativo do algoritmo é interrompido após um número máximo de gerações g_{max} ser atingido ou após uma solução ótima ser encontrada, isto é, uma solução cujo valor da função objetivo é $f(g) = 0$.

IV. EXPERIMENTOS E RESULTADOS

A proposta de avaliação do algoritmo se baseia no número de gerações necessárias, em média, para se obter uma solução ótima do problema. Os experimentos foram conduzidos a partir de ajustes dinâmicos dos hiperparâmetros p_c e p_m com os demais fixos (Tabela I). De forma a evocar o Teorema do Limite Central (TLC), foram realizadas 100 execuções de cada uma das configurações do algoritmo.

N	m	k	g_{max}
8	100	10	1.000

TABELA I

HIPERPARÂMETROS DO ALGORITMO GENÉTICO.

Inicialmente, foi realizado o ajuste da probabilidade de cruzamento no intervalo $0,6 \leq p_c \leq 1,0$ em conjunto com uma probabilidade de mutação fixa em $p_m = 0,8$ [2]. Dado o melhor p_c , pode-se ajustar o valor de p_m no mesmo domínio. A ordem de ajuste foi assumida tendo em vista que o operador de mutação causa uma menor perturbação na solução e, portanto, pode ser ajustado a posteriori.

O algoritmo genético proposto apresentou desempenhos médios relativamente similares para as diferentes probabilidades p_c analisadas. No entanto, a configuração $p_c = 0,6$ destacou-se tanto na média (54,24) quanto na dispersão em torno da média (65,24), além de ser a única configuração a não exceder 300 gerações para convergência (Figura 4).

No que se refere ao ajuste subsequente de p_m , os desempenhos médios entre as configurações apresentaram maior discrepância (Figura 5). Em associação com a melhor probabilidade de cruzamento ($p_c = 0,6$), a probabilidade $p_m = 0,9$ propiciou ao algoritmo genético atingir sua melhor performance. As soluções ótimas foram obtidas com aproximadamente 48 gerações, cerca de 24 gerações, em média, a menos que a segunda melhor configuração investigada ($p_m = 1,0$). No pior caso, foram necessárias apenas 213 gerações, valor que, mesmo sendo um *outlier* dessa distribuição, está substancialmente abaixo do número máximo de gerações g_{max} definido a priori.

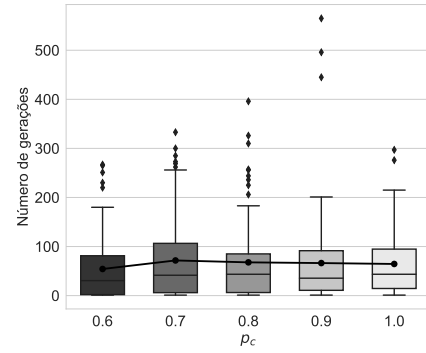


Figura 4. Otimização da probabilidade de cruzamento p_c dados os hiperparâmetros da Tabela I e a probabilidade de mutação previamente estipulada em $p_m = 0,8$. O gráfico de linha contém o número médio de gerações necessárias para convergência e o *boxplot* compreende a distribuição do número de gerações, bem como os *outliers*.

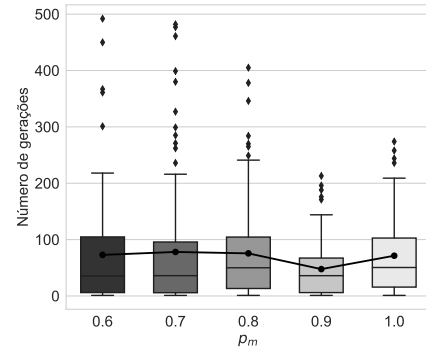


Figura 5. Otimização da probabilidade de mutação p_m dados os hiperparâmetros da Tabela I e a melhor probabilidade de cruzamento encontrada anteriormente ($p_c = 0,6$), conforme sugere a Figura 4.

V. PROBLEMA DE MINIMIZAÇÃO DA FUNÇÃO RASTRIGIN

A função Rastrigin, proposta inicialmente por Rastrigin [7] como uma função bidimensional e posteriormente generalizada por Muhlenbein [8], é uma função não-linear, não-convexa e multimodal frequentemente utilizada como função de *benchmark* para algoritmos genéticos. Diante do grande número de mínimos locais regularmente distribuídos (Figura 6), realizar uma busca no seu espaço de soluções é uma tarefa bastante difícil.

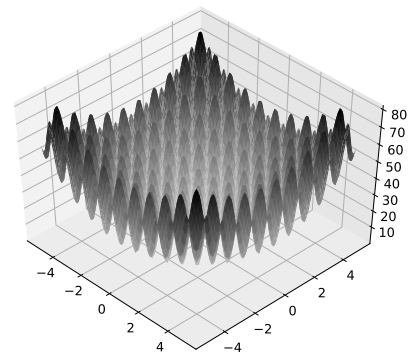


Figura 6. Função Rastrigin bidimensional ($n = 2$). Fonte: Elaborado pelo autor.

A complexidade do problema de minimização da função Rastrigin é da ordem de $\mathcal{O}(n \ln(n))$, onde n é o número de variáveis de decisão. Além disso, sua superfície é modelada por uma amplitude e uma frequência, cujo hiperparâmetro A controla. Matematicamente, a função Rastrigin multidimensional pode ser definida como (1):

$$\min_{\mathbf{x}} f(\mathbf{x}) = An + \sum_{i=1}^n (x_i^2 - A \cos(2\pi x_i)) \quad (1)$$

com $\{x_i \in \mathbb{R} \mid -5,12 \leq x_i \leq 5,12\}$. O mínimo global ocorre em $\mathbf{x}^* = 0$, onde $f(\mathbf{x}^*) = 0$.

Neste trabalho propõe-se um algoritmo genético para minimização da função Rastrigin com $n = 10$ dimensões e $A = 10$. No entanto, a solução é altamente generalizável para quaisquer valores de n e A .

VI. ALGORITMO GENÉTICO PROPOSTO

O algoritmo genético implementado para a minimização da função Rastrigin trata-se de uma adaptação do *Simple Genetic Algorithm* (SGA) proposto por Holland [9]. Em conformidade com a versão original, a representação das variáveis de decisão é feita por codificação binária, a seleção dos pais é proporcional ao valor da função objetivo e o operador de mutação é o *bit-flip* com baixa probabilidade. Entretanto, o operador de cruzamento considerado não atribui um ponto de corte por cromossomo, mas sim por variável de decisão, bem como a seleção dos sobreviventes não provém do modelo geracional clássico, ou seja, um indivíduo pode perpetuar por várias gerações.

A. Representação da solução

Sejam todas as variáveis x_i contínuas e pertencentes ao domínio $[x_{i,\min}, x_{i,\max}]$, deseja-se representar cada uma delas por sua correspondente binária \mathbf{b}_i com L_i bits, de forma a representar as soluções candidatas por uma sequência de $\sum_{i=1}^n L_i$ bits. Como a função Rastrigin é restrita ao intervalo $x_{i,\min} = -5,12$ e $x_{i,\max} = 5,12$ para qualquer $i = \{1, \dots, n\}$, todas as variáveis serão codificadas igualmente por L bits, resultando em uma solução composta por $n \times L$ bits. À medida que L aumenta, melhor a precisão da representação e, consequentemente, maior o custo computacional do algoritmo genético.

A codificação de cada variável de ponto flutuante x_i para sua representação binária \mathbf{b}_i é feita pela função $\delta^{-1} : [x_{i,\min}, x_{i,\max}] \rightarrow \mathbb{B}^L$ (2):

$$\delta^{-1}(x_i) = \varphi \left(\frac{x_i - x_{i,\min}}{\Delta_i} \right) \quad (2)$$

onde $\varphi(\cdot)$ é a função tradicional de conversão inteiro para binário e Δ_i é o intervalo entre dois valores binários adjacentes, isto é, (3):

$$\Delta_i = \frac{x_{i,\max} - x_{i,\min}}{2^L - 1} \quad (3)$$

Uma vez que a função de fitness é contínua e a representação é binária, torna-se necessário converter toda

solução binária para real antes de avaliá-la. A decodificação de cada variável binária \mathbf{b}_i de volta para sua respectiva representação real x_i é feita através da função de decodificação $\delta : \mathbb{B}^L \rightarrow [x_{i,\min}, x_{i,\max}]$ (4) [6]:

$$\delta(\mathbf{b}_i) = x_{i,\min} + \Delta_i \left(\sum_{k=0}^{L-1} b_{i(L-k)} 2^k \right) \quad (4)$$

B. Inicialização da população

Em busca de uma população inicial com indivíduos moderadamente melhores em aptidão e também de uma convergência mais rápida devido ao número limitado de avaliações da função objetivo ($n_{eval} = 10.000$), o método de inicialização utilizado foi o *Opposition-Based Population Initialization* [10].

Suponha uma população aleatória uniformemente distribuída $P(m)$ com m soluções candidatas. A população oposta à $P(m)$, denotada por $O(m)$, tem cada um de seus indivíduos calculados por (5):

$$O_{k,i} = x_{i,\min} + x_{i,\max} - P_{k,i} \quad (5)$$

onde $k = \{1, \dots, m\}$ e $i = \{1, \dots, n\}$. Como o domínio da função Rastrigin é simétrico, a Equação (5) é simplificada para $O_{k,i} = -P_{k,i}$. É interessante ressaltar ainda que a construção da população oposta é feita sobre as soluções reais e só no final da avaliação de cada indivíduo que é efetuada a codificação binária pela Equação (2).

Assim, a população inicial é definida como os m indivíduos mais aptos do conjunto $\{P(m) \cup O(m)\}$ [10].

C. Mecanismos de seleção

D. Operadores de variação

E. Critérios de parada

VII. EXPERIMENTOS E RESULTADOS

VIII. CONCLUSÕES

REFERÊNCIAS

- [1] C. Darwin and A. Wallace, "On the Tendency of Species to form Varieties; and on the Perpetuation of Varieties and Species by Natural Means of Selection," *Journal of the proceedings of the Linnean Society of London. Zoology*, vol. 3, no. 9, pp. 45–62, 1858.
- [2] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Springer, 2015.
- [3] X. Hu, R. C. Eberhart, and Y. Shi, "Swarm Intelligence for Permutation Optimization: A Case Study of N-Queens Problem," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No. 03EX706)*. IEEE, 2003, pp. 243–246.
- [4] A. Homaifar, J. Turner, and S. Ali, "The N-Queens Problem and Genetic Algorithms," in *Proceedings IEEE Southeastcon'92*. IEEE, 1992, pp. 262–267.
- [5] D. Whitley and N.-W. Yoo, "Modeling Simple Genetic Algorithms for Permutation Problems," in *Foundations of genetic algorithms*. Elsevier, 1995, vol. 3, pp. 163–184.
- [6] F. G. Guimarães, "Lecture Notes on Evolutionary Computation," 2019, Accessed on August 6, 2020.
- [7] L. A. Rastrigin, "Systems of Extremal Control," *Nauka*, 1974.
- [8] H. Mühlenbein, M. Schomisch, and J. Born, "The Parallel Genetic Algorithm as Function Optimizer," *Parallel Computing*, vol. 17, no. 6-7, pp. 619–632, 1991.
- [9] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, 1975.

- [10] S. Rahnamayan, H. R. Tizhoosh, and M. M. Salama, “A Novel Population Initialization Method for Accelerating Evolutionary Algorithms,” *Computers & Mathematics with Applications*, vol. 53, no. 10, pp. 1605–1614, 2007.