Trabalho 2 - Implementação de um algoritmos genético

1. Introdução

Este relatório descreve a implementação de um Algoritmo Genético (AG) para resolver o problema das 8 rainhas, utilizado a biblioteca PyGAD. O problema das 8 rainhas consiste em posicionar 8 rainhas em um tabuleiro de xadrez 8x8 sem que nenhuma ataque outra (mesma linha, coluna ou diagonal). As decisões de projeto, operadores genéticos e resultados são detalhados a seguir

2. Estrutura do Cromossomo e População Inicial

Em nossa codificação, cada indivíduo (cromossomo) é representado por um vetor de 8 inteiros. A posição no vetor (índice) corresponde à coluna do tabuleiro (0 a 7) e o valor armazenado indica a linha da rainha naquela coluna (0 a 7). Por exemplo, o cromossomo [3, 0, 4, 7, 1, 6, 2, 5] colocar as rainhas nas posições (0, 3), (1, 0), (2, 4), etc. Essa representação garante que sempre haverá exatamente uma rainha por coluna.

A população inicial consiste em 50 cromossomos gerados aleatoriamente, atribuindo a cada gene um valor uniforme de 0 a 7.

3. Função de Fitness

Definimos a função de fitness como o número de pares de rainhas que não se atacam, cujo máximo possível é 28. Começamos com fitness inicial igual a 28 e, para cada par de colunas, subtraímos do fitness caso as rainhas estivessem na mesma linha ou na mesma diagonal.

4. Seleção, Cruzamento e Mutação

Para escolher os pais a cada geração, adotamos seleção por torneio, onde a cada geração são selecionados aleatoriamente N indivíduos da população (por padrão é sol_per_pop // 10), e dentro do grupo selecionado, o indivíduo com maior fitness é escolhido para ser um dos pais. Para cada geração são mantidos 5 pais da geração anterior.

No cruzamento, usamos crossover single_point, que funciona dividindo os cromossomos dos pais em um único ponto de corte e troca as partes subsequentes para gerar os filhos.

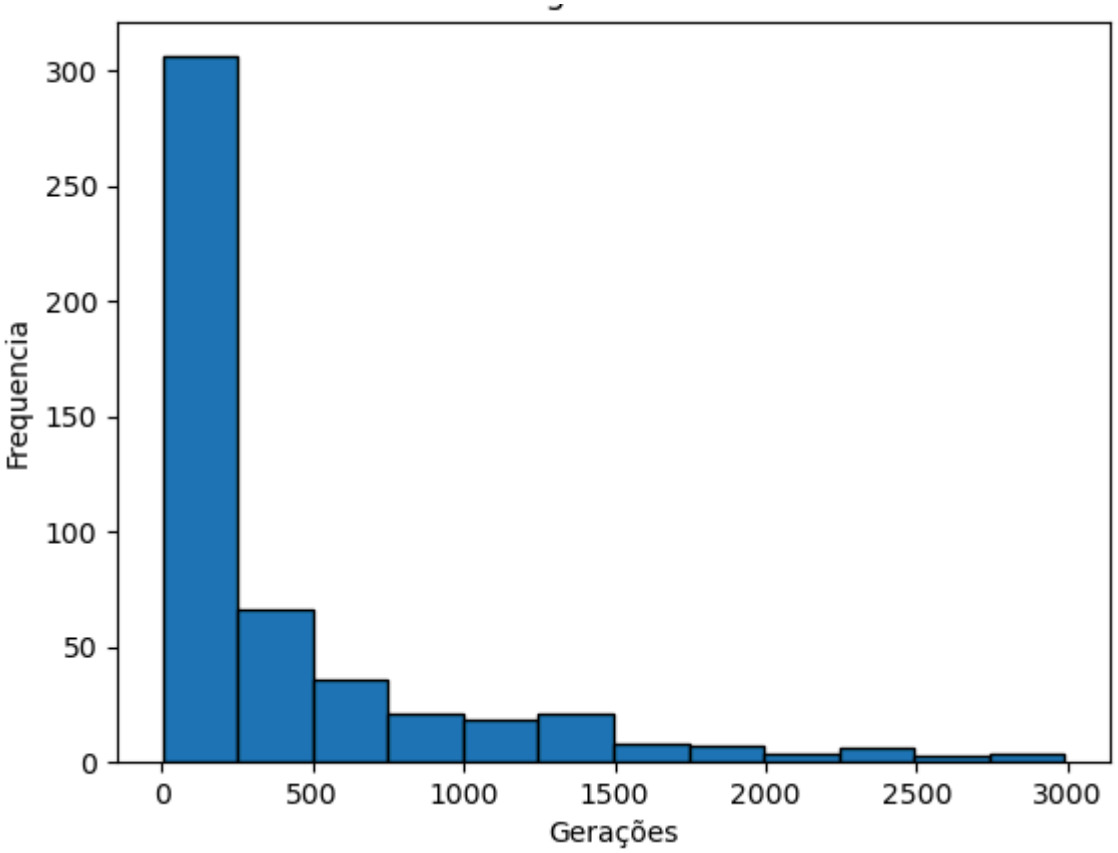
A mutação é aplicada com probabilidade de 5% para cada gene de um cromossomo, sendo do tipo random que consiste apenas na troca de seu valor.

5. Critério de Parada

Para o critério de parada, foi definido que cada instância de simulação seria rodada por 3000 gerações e, se chegasse no resultado final, a execução iria parar antes de todas gerações terem sido simuladas.

Com os parâmetros escolhidos e esse número de gerações, conseguimos convergir para um resultado na grande maioria dos casos.

6. Convergência e Análise de Resultados



Este histograma mostra a quantidade de gerações que levou para que se chegasse ao resultado esperado.

Pode-se ver que a maior parte das instâncias levou poucas gerações para convergir, enquanto algumas outras, demoraram mais de 1000 gerações para atingir o resultado esperado. Isso provavelmente se deve ao fato da solução ideal após um certo número de gerações ser dependente do fator de mutação (5% para cada gene), e, até o gene certo sofrer a mutação, a simulação fica num estado de deadlock.

7. Exemplo de Melhor Solução

Abaixo está uma das melhores soluções encontradas, com fitness 28, representada como vetor e em tabuleiro:

```
Cromossomo: [0, 4, 7, 5, 2, 6, 1, 3]
Fitness: 28

Tabuleiro (Q = rainha):
Q . . . . . . .
. . . . . . Q .
. . . . Q . . .
. . . . . . . Q
. Q . . . . . .
. . . Q . . . .
. . . . . Q . .
. . Q . . . . .
```

8. Conclusões Finais e Desafios

Em conclusão, a implementação do Algoritmo Genético para o problema das Oito Rainhas foi bastante direta, pois a biblioteca PyGAD abstrai e automatiza grande parte do trabalho.

O maior desafio, na prática, esteve no ajuste correto dos parâmetros (tamanho da população, número de gerações, probabilidade de mutação, etc) de modo a garantir que o algoritmo convirja para solução desejada dentro de um tempo de execução aceitável.