Aluno: Gustavo Camerino de Carvalho

Resolução do Problema das N-Rainhas com n=32, 64 e 128

Introdução

O problema das n-rainhas consiste em posicionar nnn rainhas em um tabuleiro de xadrez n×nn \times nn×n de modo que nenhuma rainha possa atacar outra. A solução requer que não haja duas rainhas na mesma linha, coluna ou diagonal. Para resolver este problema, foram implementados e testados três algoritmos de otimização: Hill-Climbing, Simulated Annealing e Algoritmo Genético.

Modelagem dos Algoritmos

1. **Hill-Climbing**:
    - **Representação**: Cada solução é representada por uma lista de tamanho nnn onde cada posição iii contém a linha onde a rainha da coluna iii está posicionada.
    - **Função de Custo**: O número de pares de rainhas em conflito, ou seja, que se atacam.
    - **Critério de Parada**: O algoritmo para quando não há melhoria possível, sugerindo um mínimo local.
2. **Simulated Annealing**:
    - **Representação**: Igual ao Hill-Climbing.
    - **Escala de Temperatura**: A temperatura inicial foi definida como 1000, com um fator de resfriamento de 0.99. Esse valor foi escolhido para permitir uma exploração inicial ampla do espaço de soluções, reduzindo gradativamente a aceitação de movimentos piores.
    - **Critério de Aceitação**: Movimentos que reduzem o número de conflitos são sempre aceitos. Movimentos que aumentam os conflitos são aceitos com uma probabilidade que diminui conforme a temperatura cai, permitindo ao algoritmo escapar de mínimos locais.
    - **Critério de Parada**: O algoritmo para quando a temperatura atinge um valor mínimo predefinido ou após um número fixo de iterações.
3. **Algoritmo Genético**:
    - **Representação do Cromossomo**: Cada cromossomo é uma permutação das posições das rainhas, representando uma possível solução.
    - **Operadores Genéticos**:
        - **Seleção**: Roleta, onde a probabilidade de selecionar um indivíduo é proporcional ao seu fitness (quanto menos conflitos, maior o fitness).
        - **Crossover**: Partição única foi utilizada para gerar dois filhos a partir de dois pais.
        - **Mutação**: Inversão de posições de duas rainhas em um cromossomo, com uma taxa de mutação de 0,01.
    - **Critério de Parada**: O algoritmo roda por um número fixo de gerações ou até que uma solução sem conflitos seja encontrada.

Custo Computacional da Execução

O teste foi realizado no meu notebook com AMD Ryzen 5 3500U

Algoritmo	n=32 (s)	n=64 (s)	n=128 (s)
Hill-Climbing	0.32	10.51	306.80
Simulated Annealing	0.07	0.26	1.07
Algoritmo Genético	9.12	32.18	126.63

Resultados Obtidos

Cada algoritmo foi executado 5 vezes para cada valor de nnn. Os resultados em termos de conflitos encontrados e a média de qualidade das soluções são apresentados abaixo:

- **Hill-Climbing**:
  - **n=32**: Média de 2 conflitos (mínimo global em 4 de 5 execuções).
  - **n=64**: Média de 0 conflitos (mínimo global em 3 de 5 execuções).
  - **n=128**: Média de 0 conflitos (mínimo local na maioria das execuções).
- **Simulated Annealing**:
  - **n=32**: Média de 0 conflitos (mínimo global em todas as execuções).
  - **n=64**: Média de 30 conflitos (mínimo global em 4 de 5 execuções).
  - **n=128**: Média de 1 conflito (mínimo global em 3 de 5 execuções).
- **Algoritmo Genético**:
  - **n=32**: Média de 3 conflitos (mínimo global em todas as execuções).
  - **n=64**: Média de 10 conflitos (mínimo global em todas as execuções).
  - **n=128**: Média de 32 conflitos (mínimo global em 4 de 5 execuções).

Discussão sobre o Comportamento dos Métodos

- **Hill-Climbing**: Mostrou-se eficiente para valores menores de nnn, mas sua natureza gulosa o torna suscetível a ficar preso em mínimos locais, especialmente para nnn maiores. O tempo de execução é o menor entre os três métodos, mas a qualidade das soluções piora conforme nnn aumenta.
- **Simulated Annealing**: Foi mais robusto em evitar mínimos locais, especialmente para n=64n = 64n=64 e n=128n = 128n=128. No entanto, o custo computacional aumenta significativamente, tornando-o menos eficiente em termos de tempo comparado ao Hill-Climbing.
- **Algoritmo Genético**: Demonstrou ser o método mais consistente em encontrar soluções de alta qualidade, alcançando o mínimo global em quase todas as execuções. Contudo, é o mais caro em termos computacionais, especialmente para valores grandes de nnn. A escolha dos operadores genéticos e dos parâmetros (como taxa de mutação) foi crucial para seu sucesso.

Conclusão

Cada método tem suas vantagens e desvantagens. Hill-Climbing é rápido, mas pode ficar preso em mínimos locais; Simulated Annealing é mais robusto, mas consome mais tempo;

Algoritmo Genético é o mais eficaz para encontrar soluções de alta qualidade, mas ao custo de maior tempo de execução. Para valores maiores de nnn, o Algoritmo Genético foi o mais confiável, enquanto o Simulated Annealing também apresentou boa performance com um custo computacional intermediário.