Solução das oito rainhas genéticas

Pedro de Souza Queiroga,
UFPE, CIn

Tópicos avançados em
inteligência artificial – Bioinspirada
Professor: Paulo Salgado
2019.1

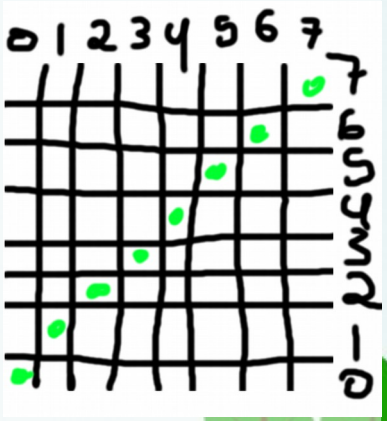
Estrutura geral do algoritmo

- Inicializa população com permutações aleatórias do arranjo [0,1,…,7]
- Repete enquanto não encontrou solução*:
  - Seleciona 2 pais da população
  - Recombina os 2 pais, gerando 2 filhos
  - Os 2 filhos sofrem mutação
  - Seleciona sobreviventes

*ou por, no máximo, vinte mil gerações

Representação

- Um indivíduo da forma [0,1,2,3,4,5,6,7] representa a diagonal de rainhas.
- Cada índice representa uma coluna, e cada número representa uma linha.
- Cada elemento representa um par (coluna,linha)



Fitness

- Indica se uma solução foi encontrada, ou quão “longe” de uma se está.
- Função do tipo fitness = erro, a busca pela solução é um problema de minimização.
- Checa quantas rainhas estão se batendo.

Seleção

- Por torneio, ou por roleta.
- Torneio:
  - Escolhe 5* indivíduos aleatórios dentro da população.
  - Seleciona os dois melhores entre os 5.
- Roleta:
  - Atribui um peso (1/1+erro) para cada indivíduo, e escolhe 2 aleatoriamente.
  - Quanto menor o fitness, maiores as chances de ser selecionado.

Recombinação

- Recombinação comum de 1 ponto
  - Não pareceu muito adequada para o problema.
  - Outras recombinações comuns (uniforme, n-pontos) também não pareceram adequadas.
  - Geram não-permutações.

Recombinação

- Recombinação de permutações
  - Ideia: gerar novas permutações.
  - Troca de informação: índice das rainhas
  - Escolhe o i-ésimo elemento do pai, descobre onde ele está na mãe (j).
  - Um filho terá o i-ésimo elemento do pai na posição j, e o outro terá o i-ésimo da mãe.
  - Faz isso para vários elementos.
  - É uma mutação guiada pelo outro integrante do casal.

Mutação

- Mutação comum
  - Para cada elemento de um indivíduo, tem uma chance de trocar um número por outro.
  - [0,1,2,3,4,5,6,7] → [0,1,2,3,4,1,6,7]
  - Mutação troca um bit do código de gray do número.
  - Gera não-permutações.

Mutação

- Mutação de permutação
  - Para cada elemento do arranjo-indivíduo, troca ele de lugar com um outro aleatório.
  - Para cada elemento, um dado é rolado para decidir se deve sofrer mutação ou não.
  - Gera novas permutações.

Seleção de sobreviventes

- Remove os dois piores indivíduos.
- Morte por idade:
  - Piorou os testes iniciais.

População

- Controle de população
  - O normal é manter uma população fixa, mas foram testadas configurações em que a população começava menor e ia crescendo até um certo limitante.
  - Não melhorou nenhuma configuração, usada apenas no começo do desenvolvimento do projeto.

Perguntas exploratórias e resultados

- Recombinação guiada por permutação melhora mesmo?
  - Sim. permutação vs 1-ponto…
  - Torneio:
    - 100, 30, 30, 133.77, 0, 449, 2.49, 4.37, 42.28
    - 100, 30, 30, 607.70, 0, 1760, 2.36, 4.03, 124.11

popSize, nRuns, nOk, avgGen, leastGen, largestGen, avgIndie, avgWorstIndie, time

Perguntas exploratórias e resultados

- Mutação de permutação é melhor do que de bit?
  - Sim. permutação vs bit:
    - 100, 30, 30, 133.77, 0, 449, 2.49, 4.37, 42.28
    - 100, 30, 19, 10020.03, 0, 20000, 2.52, 6.83, 2408.20

popSize, nRuns, nOk, avgGen, leastGen, largestGen, avgIndie, avgWorstIndie, time

Perguntas exploratórias e resultados

- Roleta é mais elitista. É melhor para este problema?
  - Não parece ser melhor.
  - Torneio vs roleta, com recombinação de permutação:
    - 100, 30, 30, 133.77, 0, 449, 2.49, 4.37, 42.28
    - 100, 30, 30, 148.73, 0, 557, 2.39, 3.73, 38.10
  - Com recombinação de 1 ponto:
    - 100, 30, 30, 607.70, 0, 1760, 2.36, 4.03, 124.11
    - 100, 30, 30, 748.43, 0, 3358, 2.44, 4.37, 150.05

popSize, nRuns, nOk, avgGen, leastGen, largestGen, avgIndie, avgWorstIndie, time

Mais resultados interessantes

- Encontrar solução vs convergir (permutações e torneio):
  - 100, 30, 30, 133.77, 0, 449, 2.49, 4.37, 42.28
  - 100, 30, 30, 2327.30, 1454, 3066, 0.00, 0.00, 560.44
- Encontrar solução vs convergir (permutações e roleta)
  - 100, 30, 30, 148.73, 0, 557, 2.39, 3.73, 38.10
  - 100, 30, 30, 2972.43, 1703, 4673, 0.00, 0.00, 698.17

popSize, nRuns, nOk, avgGen, leastGen, largestGen, avgIndie, avgWorstIndie, time

Mais resultados interessantes

- Encontrar solução vs convergir (1-ponto, bit, torneio):
  - 100, 30, 24, 7541.50, 0, 20000, 2.68, 6.50, 1554.79
  - 100, 30, 15, 20000.00, 20000, 20000, 1.93, 7.63, 4160.98
- Encontrar solução vs convergir (1-ponto, bit, roleta):
  - 100, 30, 15, 12890.80, 0, 20000, 2.42, 6.83, 2612.52
  - 100, 30, 21, 20000.00, 20000, 20000, 1.93, 7.77, 4078.75

popSize, nRuns, nOk, avgGen, leastGen, largestGen, avgIndie, avgWorstIndie, time

Mais resultados interessantes

- Influência do tamanho da população:
  - 10, 30, 30, 85.43, 6, 393, 1.47, 3.23, 2.27
  - 50, 30, 30, 78.43, 0, 206, 2.27, 3.67, 8.63
  - 100, 30, 30, 133.77, 0, 449, 2.49, 4.37, 42.28

  Se não mantermos permutações:
  - 10, 30, 17, 13497.87, 1382, 20000, 1.91, 7.17, 362.42
  - 50, 30, 13, 14649.23, 0, 20000, 2.04, 6.90, 1626.80
  - 100, 30, 24, 7541.50, 0, 20000, 2.68, 6.50, 1554.79

popSize, nRuns, nOk, avgGen, leastGen, largestGen, avgIndie, avgWorstIndie, time

Conclusão

- Manter permutações é **bastante** interessante.
- A representação é boa parte do problema.
- Seria bom experimentar com uma inicialização estruturada.

- 30 execuções não é suficiente para convergir a média

mas já faz com que alguns casos demorem bastante.