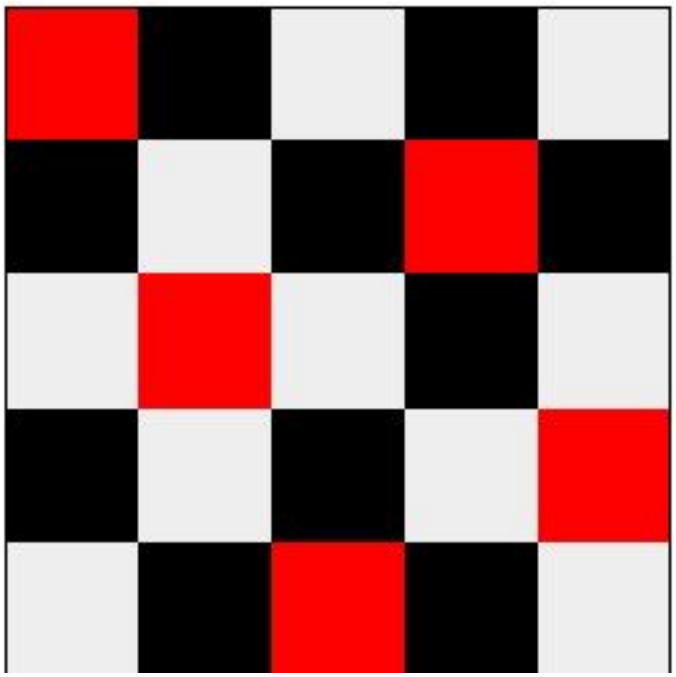
Alunos: Lucas Jacques e Lucas Vieira

Relatório sobre as soluções do Problema das N-Rainhas

O que é?

O problema das N-Rainhas consiste no posicionamento de um número N de rainhas proporcional ao tamanho NxN do tabuleiro, de forma que nenhuma rainha consiga atingir outra em apenas um passo, por exemplo: em um tabuleiro 5x5 é necessário posicionar 5 rainhas, uma das possíveis soluções – em vermelho, as rainhas – seria:



Soluções

Backtracking:

Consiste em um algoritmo de busca exaustiva que constrói candidatos-solução do problema proposto em uma árvore com todas as possibilidades de caminhos gerados pelo ambiente em que se encontra o problema e segue, como qualquer busca exaustiva, todos os caminhos até encontrar uma solução, porém ao chegar em um nodo da árvore em que não foi encontrado uma solução válida, o algoritmo em questão volta para um nível de profundidade acima do atual e examina os demais caminhos-candidatos a solução, por isso recebe o nome de “backtracking”.

A implementação para o problema das n rainhas consiste em ir posicionando as rainhas coluna a coluna na primeira linha que não estiver ameaçada. Assim que não existir nenhuma linha não ameaçada, volta-se uma coluna e posiciona-se a rainha desta coluna anterior no próximo local não ameaçado, modificando a configuração para a próxima coluna que agora, poderá haver possibilidades de locais não ameaçados.

Iterative Repair:

As rainhas são inicialmente colocadas aleatóriamente em linhas e colunas de maneira que cada rainha fique em uma linha diferente e uma coluna diferente. A cada iteração a próxima rainha que está sendo ameaçada por qualquer outra rainha é movida dentro de sua linha para o local em que há menos rainhas ameaçadoras. Se existirem vários destes pontos, um é escolhido de forma aleatória. Eventualmente os locais com menos rainhas ameaçadoras serão os mesmos que os locais com nenhuma rainha ameaçadora, ou seja, nem uma rainha estará ameaçando ou sendo ameaçada.

Greedy:

As rainhas são inicialmente colocadas aleatóriamente em linhas e colunas de maneira que cada rainha fique em uma linha diferente e uma coluna diferente. A cada iteração, é escolhido um movimento que reduz o número de rainhas atacadas no tabuleiro pela maior quantidade possível, caso existam mais de um movimento com o mesmo resultado, um é escolhido aleatóriamente, dando preferência para os movimentos que não envolvam a mesma rainha que foi movida na ultima iteração.

Soluções implementadas

Escolhemos implementar o algoríitimo backtracking e o iterative-repair, apesar de saber que o backtracking seria muito mais lento, optamos por ele, exatamente para mostrar a diferença de performance de uma busca exaustiva para algorítimos que utilizam de conhecimentos específicos do problema para resolve-lo (heurísticas).

É possível observar que o backtracking sempre encontrará a mesma solução e com a mesma quantidade de passos. Já o iterative-repair pelo seu fator aleatório é capaz de encontrar diferentes soluções com uma quantidade de passos variada porém sempre menor que o backtracking. Também observamos que a medida que o tabuleiro cresce, rapidamente o backtracking se torna inviável para encontrar a solução.

Em um tabuleiro 4x4 ele sempre vai levar 12 iterações para descobrir a solução, parece pouco, porém o iterative-repair leva em média 5 iterações para resolver esse tabuleiro, além disso é possivel que ele encontre a solução apenas ao executar seu primeiro passo, (colocar as rainhas aleatoriamente), já que o tabuleiro 4x4 é pequeno.

Em um tabuleiro maior, 12x12 por exemplo, começamos a notar o verdadeiro brilho da busca com heuristica em relação a exaustão, o backtracking começa a levar 510 iterações, enquanto o iterative repair é capaz de resolver com apenas 30 em média (conseguimos chegar a solução em apenas 6 iterações).

Como executar o código

Escolhemos implementar na linguagem javascript, principalmente por ser fácil montar a interface em html, dessa maneira basta abrir o index.html com um navegador qualquer, no caso do Internet Explorer, pedimos que seja aberto a partir da versão 9 (versões anteriores podem não suportar novas features do javascript que foram utilizadas).

Como alternativa deixamos o código aberto aqui:
https://github.com/vieiralucas/n-queen
Que pode ser visto em execução aqui:
http://vieiralucas.github.io/n-queen/

Bibliografias:

- https://pt.wikipedia.org/wiki/Backtracking
- http://pushedpawn.org/chesspedia/Eight_queens_puzzle.htm
- http://yuval.bar-or.org/index.php?item=9