Backtracking Recursivo

Daniel Oliveira

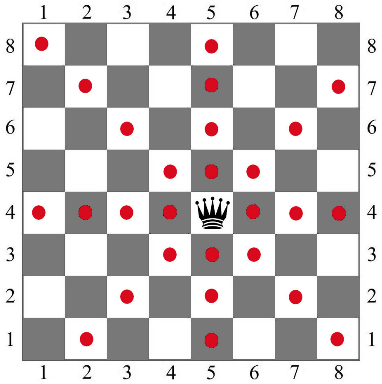
Departamento de Informática - UFPR

Janeiro 2021

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

8-rainhas

- Como colocar 8 rainhas num tabuleiro de xadrez de forma que nenhuma rainha ataque outra rainha?
  - Rainhas se movimentam (e capturam peças) em toda a linha, coluna, diagonal e anti-diagonal

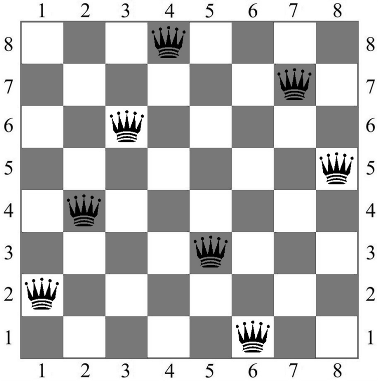


- Verificar se uma rainha na posição (i, j) ataca outra é trivial:
  - linha: i
  - coluna: j
  - diagonal: $i + j$
  - anti-diagonal: $i - j$

n-rainhas

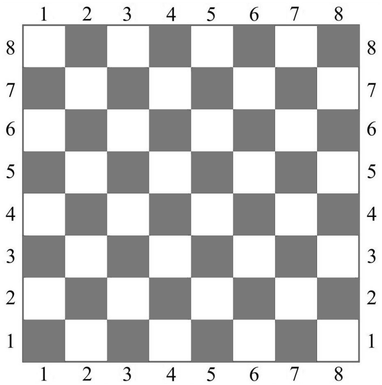
n-rainhas
Instância: n, onde n é a quantidade de rainhas colocadas em um tabuleiro de $n \times n$.
Resposta: Todas as soluções onde as n rainhas não se atacam

8-rainhas - uma solução



8-rainhas - Proposta #1

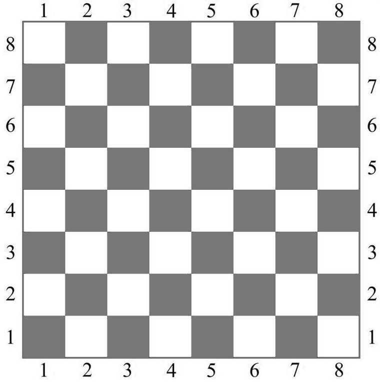
- Testar todas as combinações possíveis e verificar quais respeitam as condições



- Quantas combinações diferentes existem?
  - $\binom{64}{8} = 4.426.165.368$

8-rainhas - Proposta #2

- Por que não caminhar apenas por linhas, ou colunas?
  - Nenhuma solução pode ter mais de uma rainha na mesma linha
  - Testar todas as combinações com uma rainha por linha



8-rainhas(v, l)

```
Se l > 8
    testa_e_imprime(v)
Senão
    c ← 1
    Enquanto c > 8
        v[l] ← c
        8-rainhas(v, l + 1)
        c ← c + 1
```

- Possibilidades (#chamadas recursivas): $8^8 = 16.777.216$, equivale a $0,38\%$ de $\binom{64}{8}$

Backtracking

- Uma maneira de examinar todas as possibilidades **produtivas**, abandonando situações completamente exploradas
  - Método batizado de *backtrack* por R. J. Walker por volta de 1950
  - 'possibilidades' se referem a diferentes configurações num espaço de busca (8 rainhas no tabuleiro)
    - Todas as configurações devem ser geradas de maneira sistemática
  - Normalmente o algoritmo termina na primeira solução encontrada para o problema
- Backtracking, basicamente, faz uma exploração parcial em vez de exaustiva do espaço de busca

Backtracking - Framework geral

- Buscamos todas as sequências $x_1, x_2, ..., x_n$ tal que uma propriedade $P_n(x_1, x_2, ..., x_n)$ é verdadeira
  - onde x_n pertence a um domínio D_k de inteiros

- Backtracking, na sua forma básica, envolve o uso de uma propriedade de 'corte' ou 'poda' $P_l(x_1, ..., x_l)$ para $1 \leq l < n$ tal que:
  1. $P_l(x_1, ..., x_l)$ é verdadeiro sempre que $P_{l+1}(x_1, ..., x_{l+1})$ é verdadeiro
  2. $P_l(x_1, ..., x_l)$ é fácil de testar se $P_{l-1}(x_1, ..., x_{l-1})$ é verdadeiro

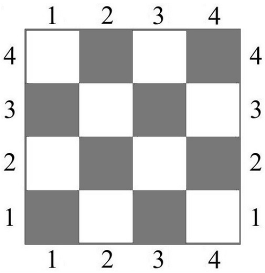
Backtracking - Algoritmo geral

Solucao()

Inicializa P, l e n
busca_solucao(P, l, n)

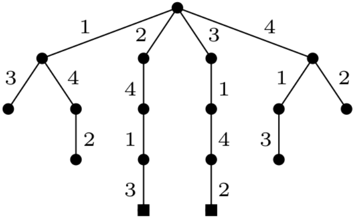
busca_solucao(P, l, n)

Se $l > n$
    Uma solução P encontrada
Senão
    Para *todo* x_l *pertencente ao domínio* D_k
        Insere x_l em P
        Se P *é verdade*
            busca_solucao($P, l + 1, n$)
        Remove x_l de P {Passo de backtrack}



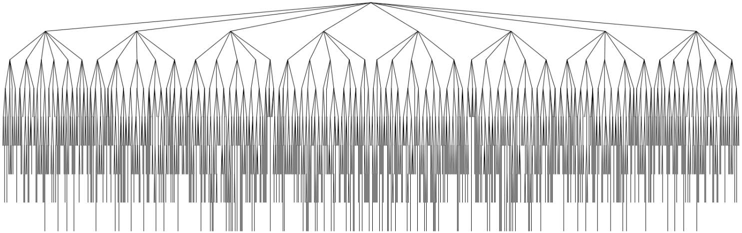
Backtracking - 4-rainhas

- Árvore de busca, ou árvore de backtrack, para 4-rainhas:
  - 17 nós
  - número de soluções ($P_n = P_4$) é 2



Backtracking - 8-rainhas

- Árvore de busca, ou árvore de backtrack, para 8-rainhas:
  - 2057 nós
    - 2057 é apenas $0,01\%$ de $8^8 = 16.777.216$
    - 8^8 é apenas $0,38\%$ de $\binom{64}{8} = 4.426.165.368$
  - número de soluções $(P_n = P_8)$ é 92



Backtracking - 27-rainhas

- Em 2016 o problema de 27 rainhas foi resolvido (tabuleiro de 27x27)
  - Um cluster de FPGAs foi usado por 383 dias
  - 234.907.967.154.122.528 soluções foram encontradas

- Sendo Q(n) o número de soluções para o problema de n-rainhas, temos:

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Q(n)	1	1	0	0	2	10	4	40	92	352	724	2680	14200	73712	365596	2279184	14772512

Fim

- O conteúdo desta aula está em Knuth, Donald E. Art of Computer Programming, Volume 4, Fascicle 5: Mathematical Preliminaries Redux; Introduction to Backtracking; Dancing Links. Addison-Wesley Professional