Algoritmos e Estrutura de Dados II
(ESTCMP011)
2do PERIODO 2018

UEA
UNIVERSIDADE DO ESTADO DO AMAZONAS

Backtracking

Prof. Luis Cuevas Rodríguez, PhD
E-mail: lcuevasrodriguez@gmail.com / lrodriguez@uea.edu.br
Celular: 9298154648

EST
Escola Superior de Tecnologia da UEA

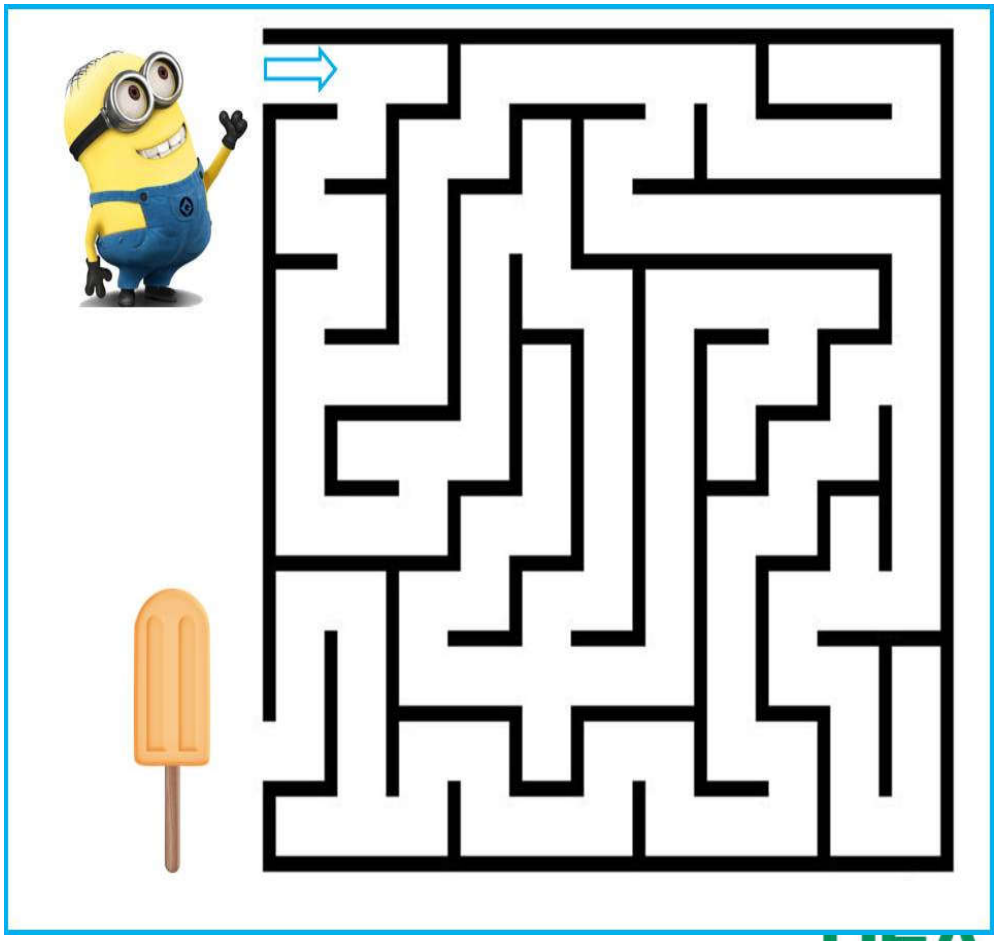
NUCOMP
NÚCLEO DE COMPUTAÇÃO

Conteúdo

- Definição Backtraking (volta atrás)
- Resolver problemas de decisão.
- Implementação

Backtracking – Volta trás

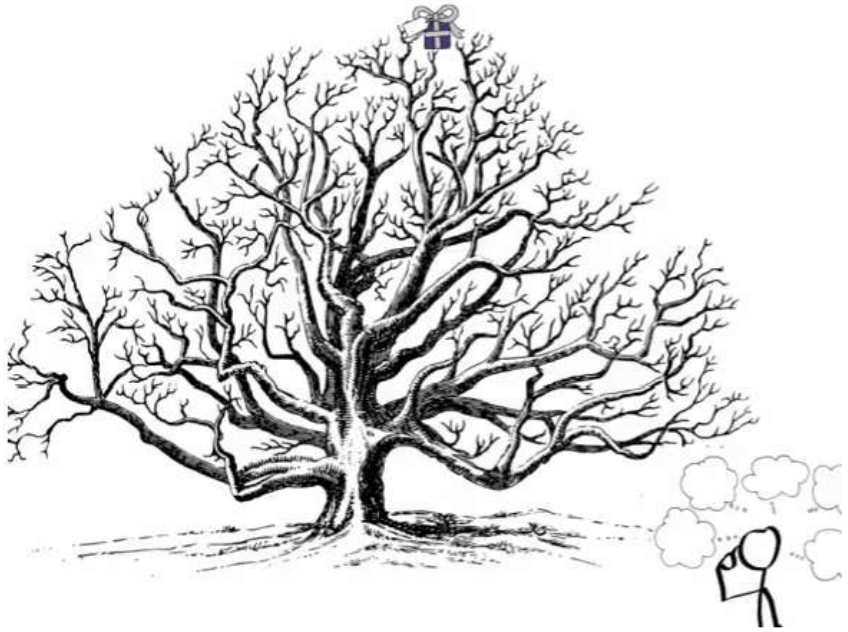
- É uma técnica de solução de problemas ou (desenho de algoritmos) que examina o espaço de soluções de forma exaustiva, procurando uma solução.
- Com o menor esforço possível, com eficiência.

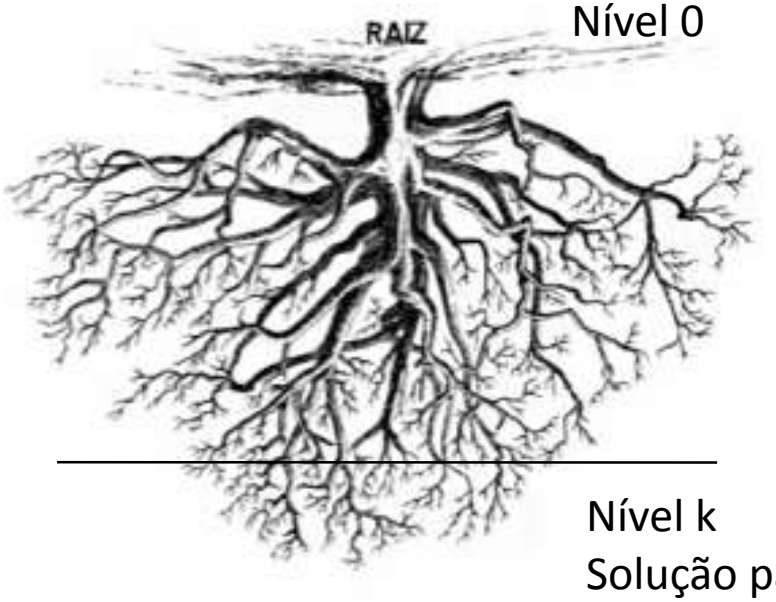


Backtracking – Volta trás

- Resolver o problema por força bruta (complexidade exponencial ou fatorial)
- Usando backtracking se pode resolver o problema com uma complexidade menor, porque são consideradas as características do problema.

Backtracking – Volta trás





Folha do arvore
-Temos uma solução que é única
-Têm mais de uma solução, buscar as outras
-Não é uma solução

Backtracking – Volta trás

- Se podem abandonando famílias de caminhos por alguma inviabilidade seja determinada (exaustão inteligente).
- É particularmente aplicável a problemas NP-Completos.

Backtracking – Volta trás

- Para problemas modelado por médio de componentes da solução. Que podem ser acrescentados á solução passo a passo.
- Sistemática e exaustiva
- Busca em profundidade.

Aplicação

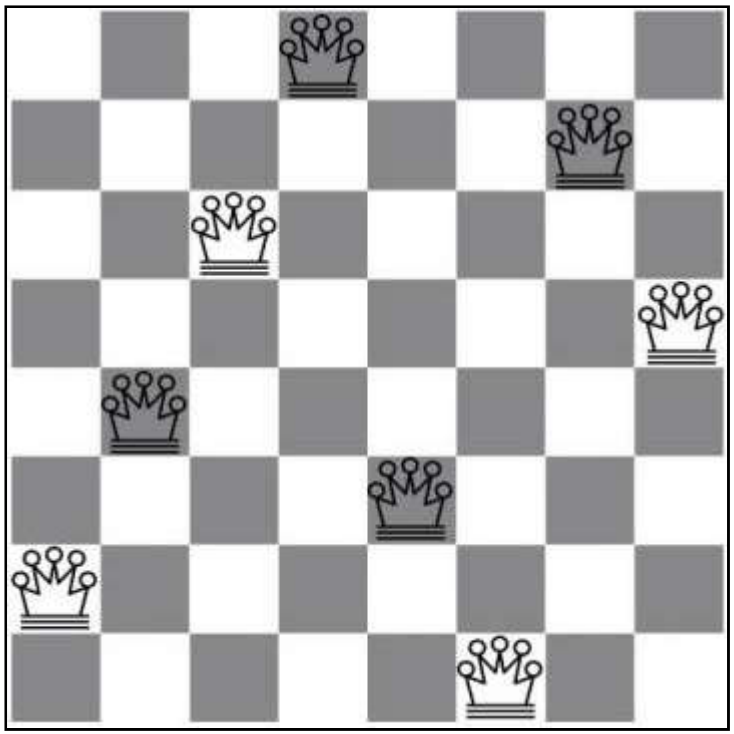
- Aplicar backtracking a problemas de decisão
  - Procurar **só uma solução**, qualquer uma, não têm que ser a melhor solução.
    - Não usa função objetivo (Mínimo custo ou Maximo beneficio) → problema de otimização

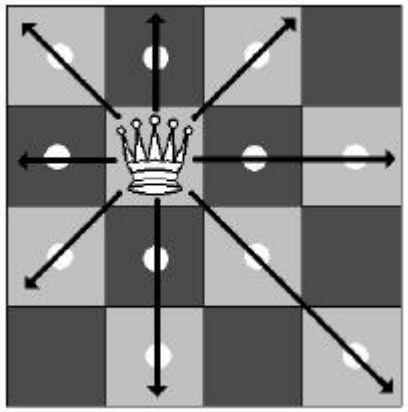
Passos

1. Representar a solução do problema.
2. Representar a arvore de busca.
3. Codificar (recursividade)

Exemplo

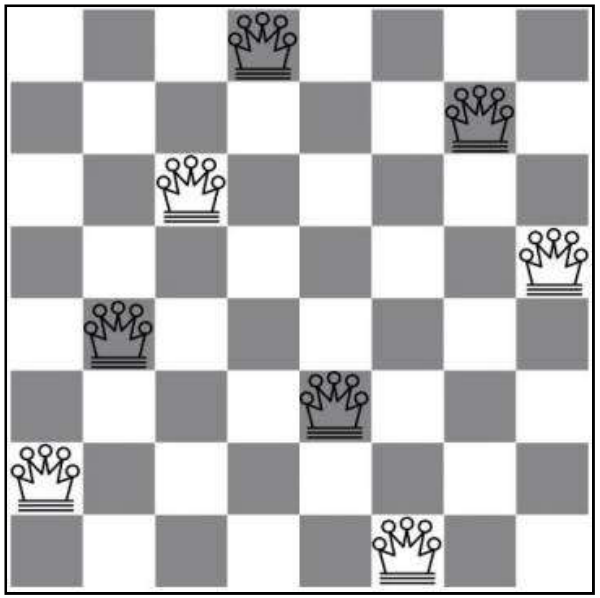
- Exemplos de problemas
  - Problema das oito rainhas.





Exemplo

- Passo 1:
  - Quero retornar onde está cada rainha no tabuleiro?

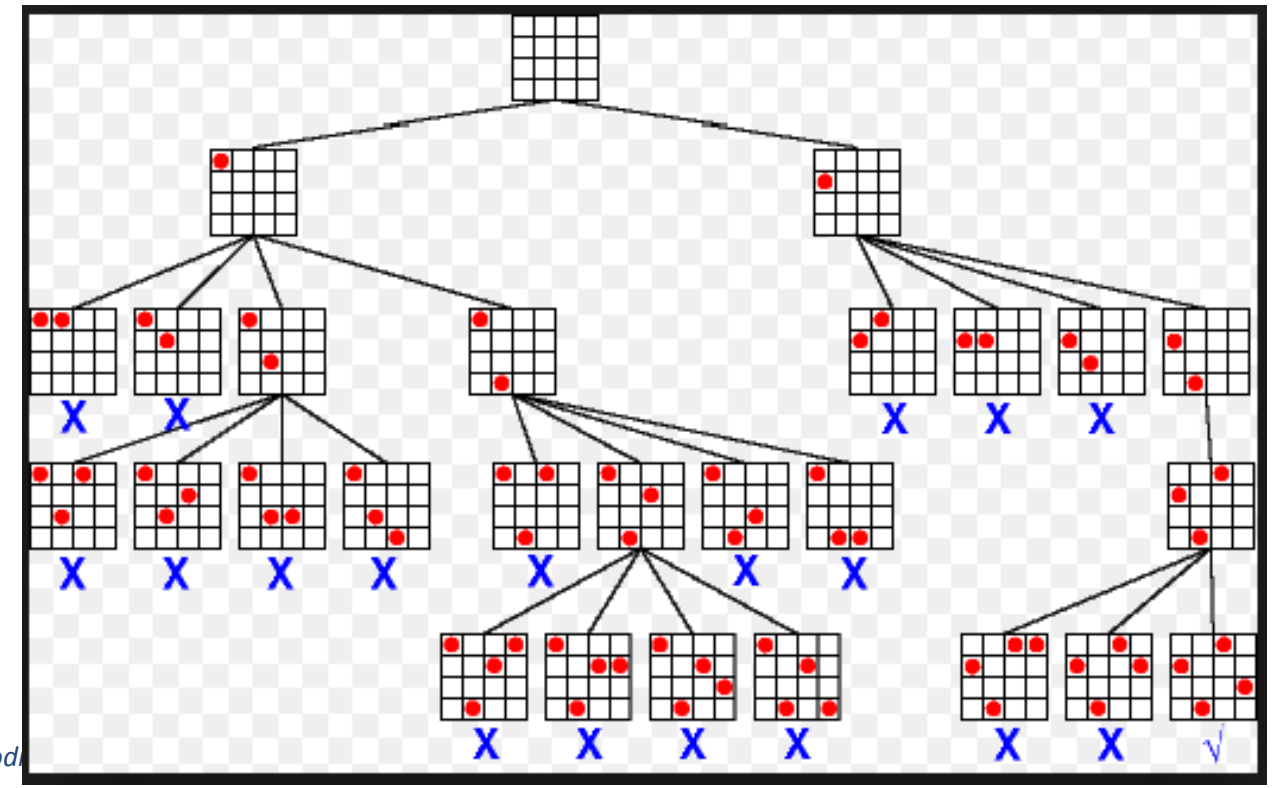


Vetor

6	4	2	0	5	7	1	3

Exemplo

- Passo 2:
  - Árvore de solução



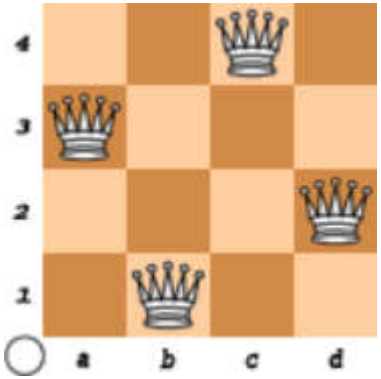
Implementação

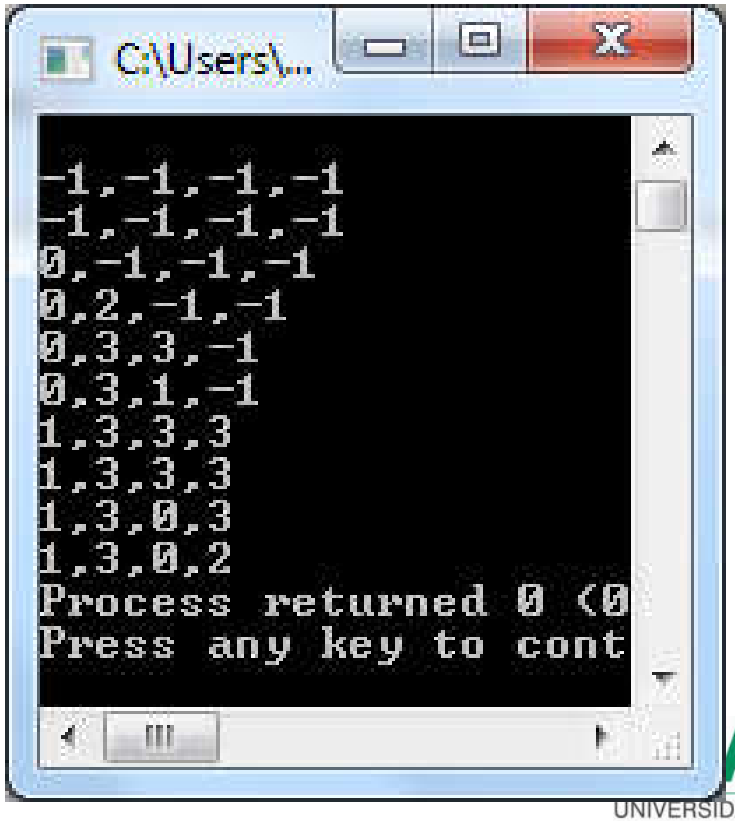
```
bool rainhas (int solucao[], int columna){
    bool sucesso;
    imprime_vetor(solucao);
    if (columna > n-1)
        return false;
    sucesso = false;
    solucao[columna]=-1;
    while (!sucesso && solucao[columna] < n-1){
        solucao[columna]=solucao[columna]+1;
        if (valido(solucao,columna)){
            if (columna != n-1)
                sucesso = rainhas(solucao,columna+1);
            else
                sucesso = true;
        }
    }
    return sucesso;
}
```

Implementação

```
bool valido(int solucao[],int colum){
    for (int i=0; i<=colum-1;i++){
        if ((solucao[i]==solucao[colum])|| (abs(solucao[i]-solucao[colum])==abs(i-colum)))
            return false;
    }
    return true;
}
```

```
int main()
{
    int pos_rainhas[n];
    inicializalista(pos_rainhas);
    imprime_vetor(pos_rainhas);
    rainhas(pos_rainhas,0);
    imprime_vetor(pos_rainhas);
    return 0;
}
```





Exercícios

1. Construir a árvore de solução que mostre todas a soluções para o seguinte Sodoku.

		2	
4	2		
2			3

Exercícios

2. Simular para o problemas das rainhas as 3 primeiras soluções para um tabuleiro de 5x5 ou 6x6.
3. Procurar em Internet outro problemas que possam ser resolvido utilizando a técnica backtraking.