Universidade Federal de Ouro Preto
Departamento de Computação

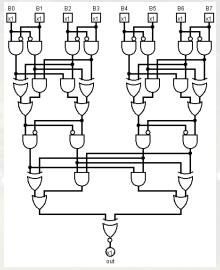
Backtracking: Busca com Retrocesso

decom
departamento de computação

Prof. Haroldo Gambini Santos

Busca Combinatória

- Para vários problemas estamos dispostos a gastar muito processamento na obtenção da solução exata
  - Ex.: testar um programa ou circuito para todas as entradas possíveis e verificar sua corretude



Aproveitando sua CPU

- Computadores com ciclos em Gigahertz são baratos
- Em que isso ajuda:
  - bilhões de operações por segundo
  - computação de uma solução toma centenas/milhares de ciclos
  - milhões de soluções:
    - permutações de 10, 11 elementos
    - combinações de 20 elementos
  - solucionar problemas muito maiores que isso envolve a poda cuidadosa do espaço de soluções

Modelagem

- Solução, vetor:
  - $\mathbf{a}=(a_1, a_2,\ldots,a_n)$
  - Cada elemento a_i tem um conjunto S_i de valores possíveis
- O método trabalha com ***soluções parciais*** que são estendidas
- A cada passo:
  - checar se a solução está completa
  - checar as maneiras de estender a solução no passo k seguinte (construir S_k)

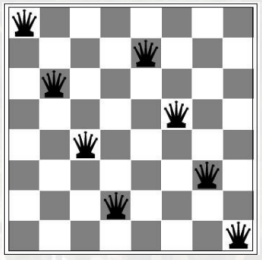
Backtracking com Busca em Profundidade

```
procedure backtracking(a, k)
begin
    if (a = (a_1, a_2, ..., a_n)) is a solution then
        report it
    else
    begin
        k + 1
        compute S_k
        while (S_k ≠ ∅) do
        begin
            a_k ← element from S_k
            S_k ← S_k \ {a_k}
            backtracking(a, k)
        end
    end
end
```


Backtracking

- Busca em Profundidade – DFS
  - Solução representada pelo caminho entre raiz e nó folha
  - Requerimento de espaço: altura da árvore
- Busca em Largura
  - Espaço: largura da árvore !

Problema das 8 Rainhas



Kakuro