RACIOCÍNIO AUTOMÁTICO
INTRODUÇÃO

Luís Morgado

ISEL-DEETC

RESOLUÇÃO AUTOMÁTICA DE PROBLEMAS

- **DECISÃO SEQUENCIAL**
  - Saber o que fazer
  - Resultado: Políticas de acção
- **PLANEAMENTO**
  - Saber como fazer
  - Resultado: Planos
- **OPTIMIZAÇÃO**
  - Saber qual a melhor configuração de parâmetros perante um conjunto de restrições
  - Resultado: Configurações de parâmetros

Representação de conhecimento

Raciocínio automático

- **CLASSIFICAÇÃO**
  - Saber o que é
  - Resultado: Conceitos, Relações

Aprendizagem automática

RACIOCÍNIO AUTOMÁTICO

- Capacidade de um sistema computacional resolver de forma automática um problema com base numa representação de conhecimento do respectivo domínio, produzindo uma solução a partir de diversas alternativas possíveis
- Processo computacional que tendo como entrada uma **representação de conhecimento** de um determinado domínio, **produz como resultado conclusões baseadas nesse conhecimento**
- O processo de manipulação da representação de forma a obter conclusões é normalmente designado ***inferência***

RACIOCÍNIO AUTOMÁTICO

- **Processo cognitivo**
  - Representação de conhecimento
  - Exploração de alternativas
  - Controlo de processamento
- **Tipos de raciocínio**
  - Raciocínio teórico
    - Orientado para o conhecimento
  - Raciocínio prático
    - Orientado para acção
- **Suporte da tomada de decisão**

RACIOCÍNIO AUTOMÁTICO

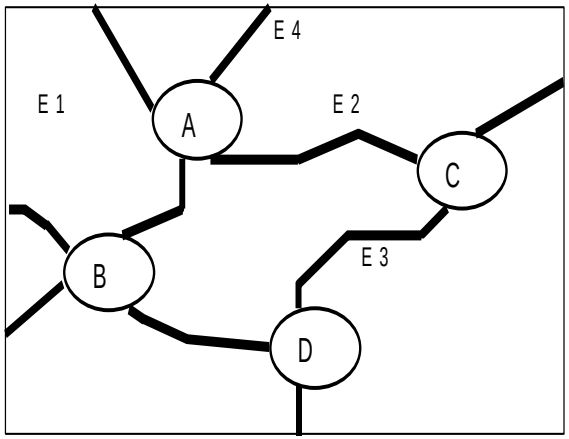
- **Aspectos principais**
  - **Representação do problema**
    - Determinante para o processo de raciocínio
      - **Eficiência**
      - **Eficácia**
  - **Método de raciocínio**
    - Define método de exploração de alternativas (opções)
    - Define estratégia de controlo do raciocínio
      - Critérios de prioridade de exploração de alternativas

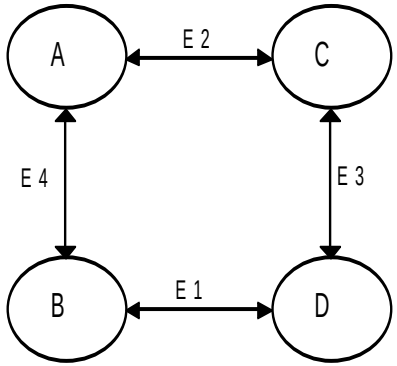
MÉTODOS DE RACIOCÍNIO AUTOMÁTICO

- **Exemplos**
  - Inferência lógica
  - Inferência estatística
  - Planeamento simbólico
  - Procura em espaços de estados
  - Processos de decisão sequencial

REPRESENTAÇÃO DO PROBLEMA

Exemplo: representação de uma rede de estradas

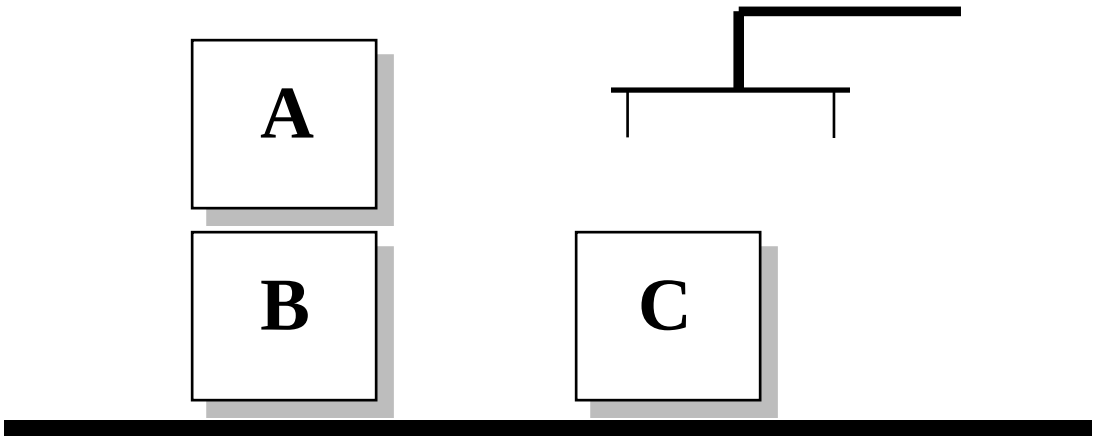




- A forma de representação deve ser adequada
  - Ao domínio do problema
  - Ao método de raciocínio

	A	B	C	D
A		E 4	E 2	-
B	E 4		-	E 1
C	E 2	-		E 3
D	-	E 1	E 3	

EXEMPLO: MUNDO DOS BLOCOS



[Wooldridge, 2002]

- Robot
- Blocos
- Mesa

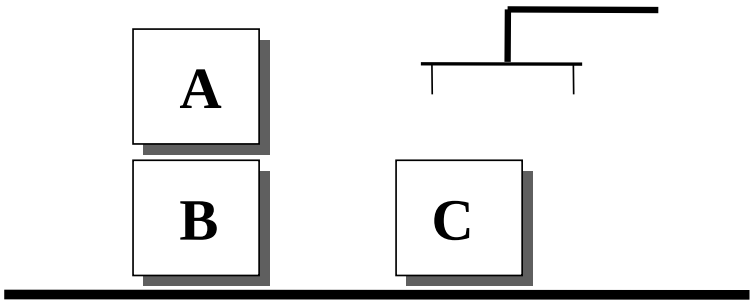
MUNDO DOS BLOCOS

- Uma representação possível (simbólica)
  - on(x, y) Bloco x está sobre bloco y
  - on_table(x) Bloco x está sobre a mesa
  - clear(x) Bloco x está livre
  - holding(x) Braço do robot tem x
  - arm_empty Braço do robot está livre

MUNDO DOS BLOCOS

- Exemplo de **representação da situação** do mundo

clear(A)
on(A, B)
on_table(B)
on_table(C)
arm_empty



MUNDO DOS BLOCOS

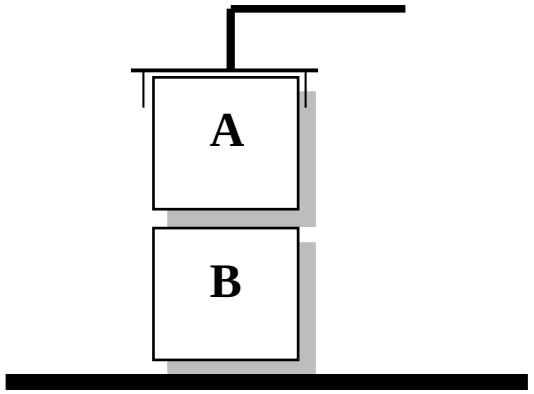
- Exemplo de **representação de acção**

stack(*x*, *y*)

- Pré-condições: clear(*y*), holding(*x*)
- Remover: clear(*y*), holding(*x*)
- Adicionar: arm_empty, on(*x*, *y*)

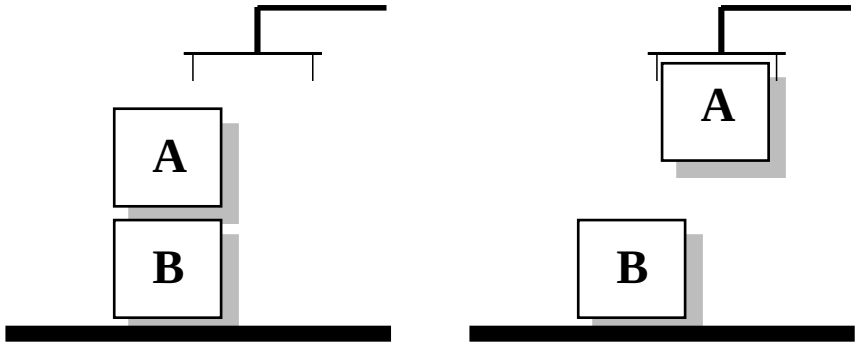
unstack(*x*, *y*)

- Pré-condições: on(*x*, *y*), clear(*x*), arm_empty
- Remover: on(*x*, *y*), arm_empty
- Adicionar: clear(*y*), holding(*x*)



- **Representação relacional**

- **Raciocínio simbólico**

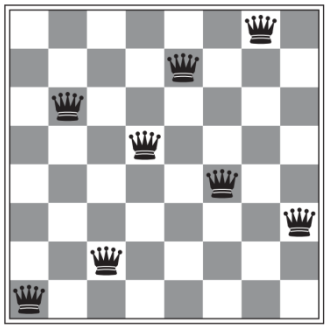


on_table(B)
on(A, B)
clear(A)
arm_empty

on_table(B)
clear(B)
holding(A)

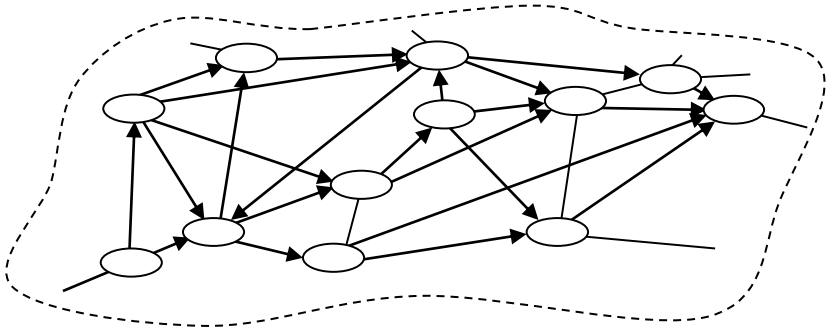
ESPAÇO DE CONFIGURAÇÕES

- **Configuração** (problema, sistema)
  - Simbólica
  - Numérica
- **Transformação** de estado
  - Definição funcional
  - Definição procedimental
  - Definição relacional (simbólica)
- **Exemplo**: Problema das rainhas



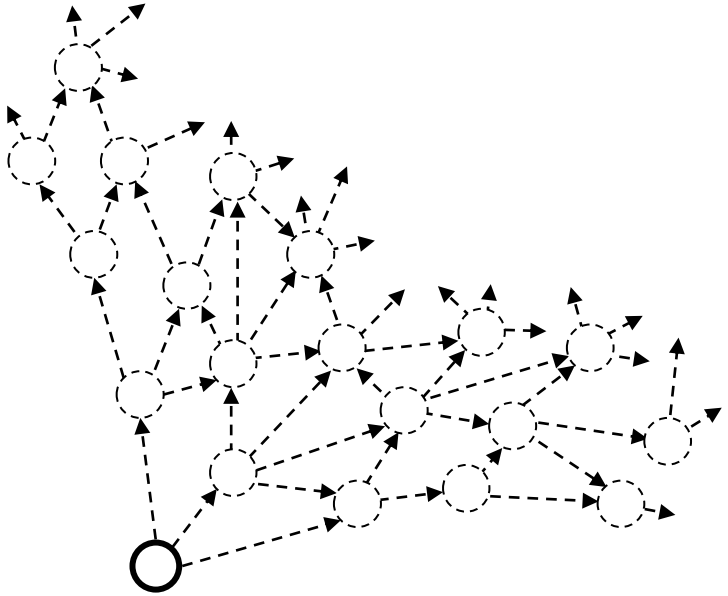
RACIOCÍNIO ATRAVÉS DE PROCURA

SIMULAÇÃO DE HIPÓTESES DE EVOLUÇÃO DE ESTADO



Estado

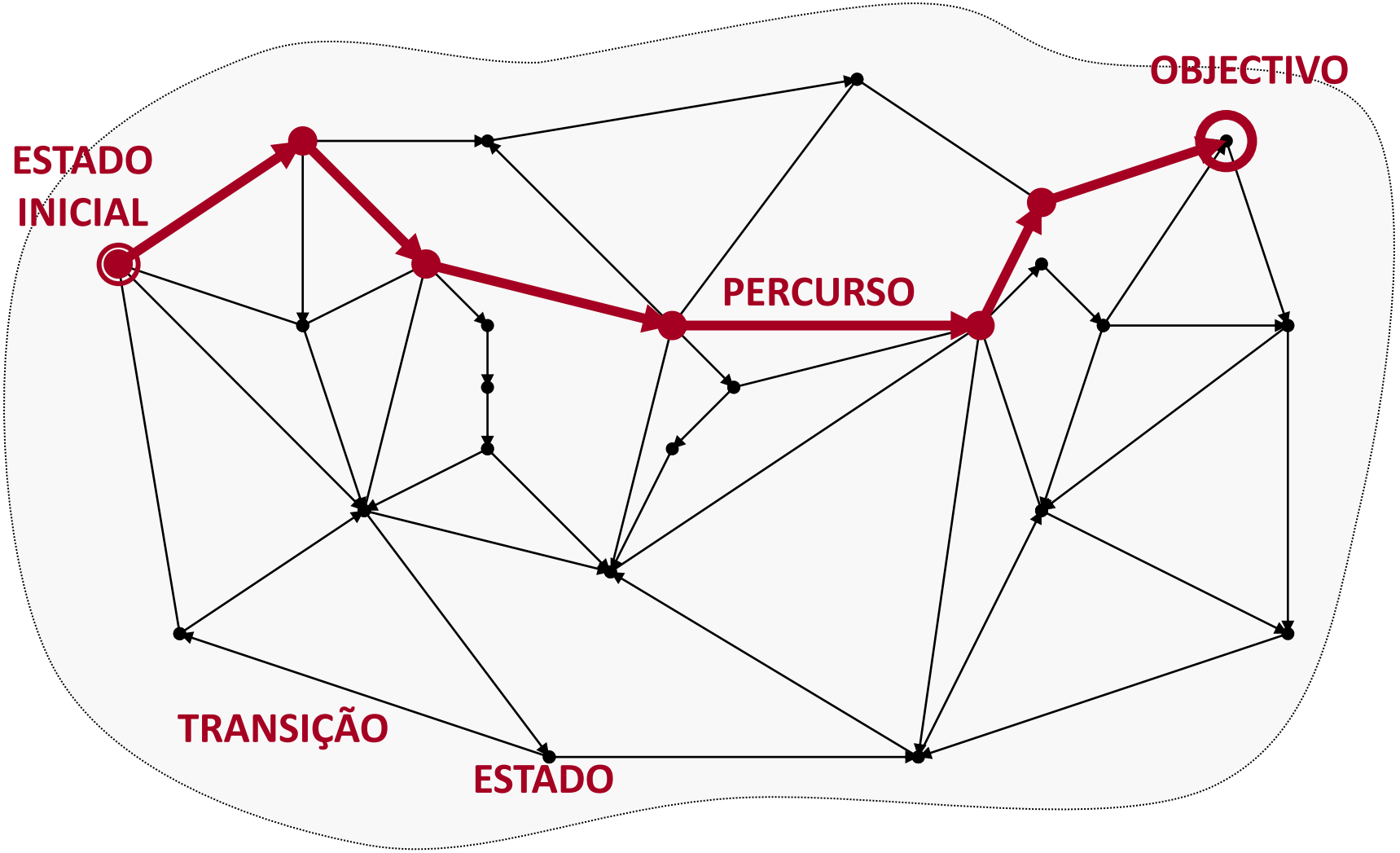
Transição de estado



PROCURA EM ESPAÇOS DE ESTADOS

RACIOCÍNO ATRAVÉS DE PROCURA

PROBLEMAS DE PLANEAMENTO

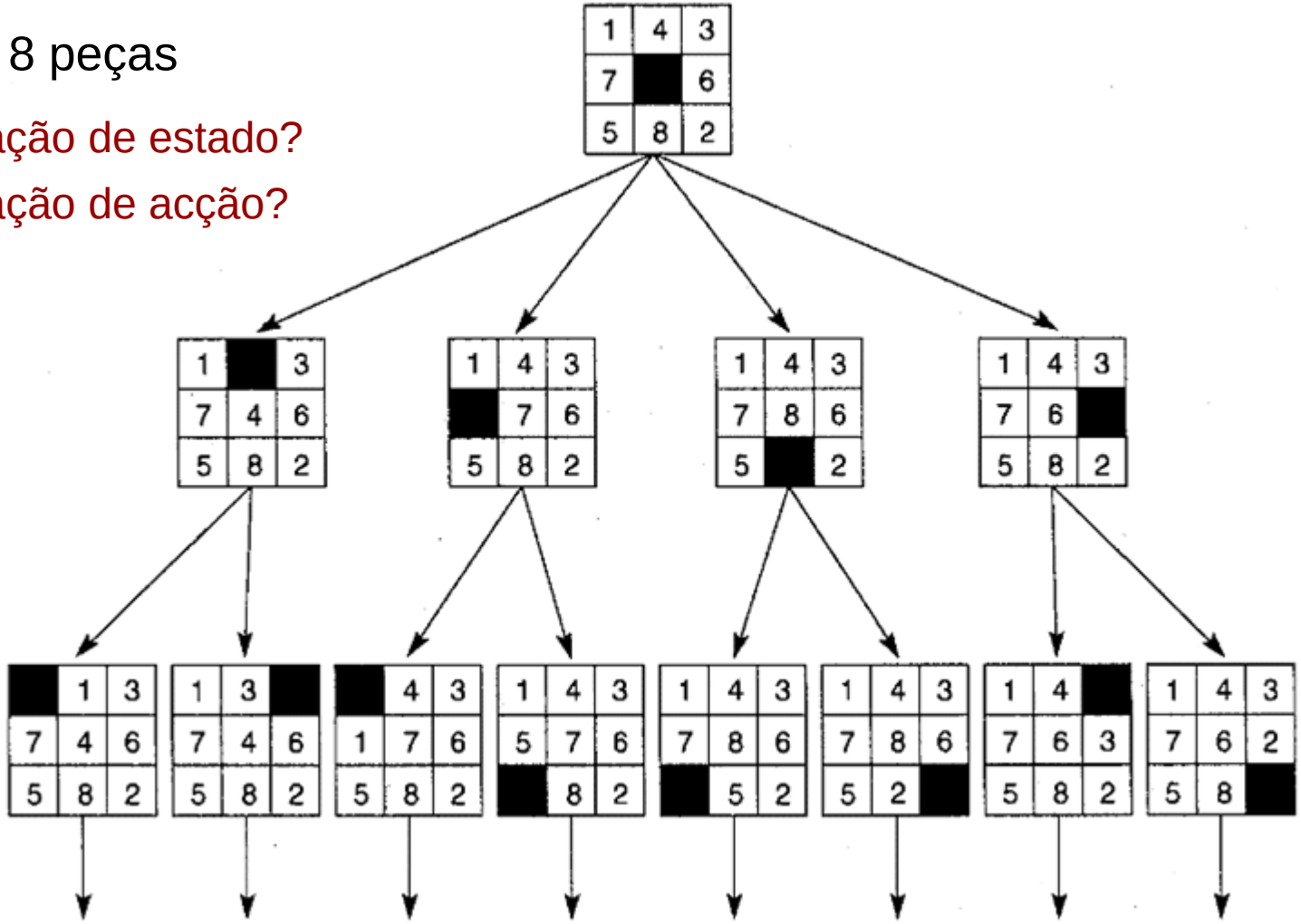


RACIOCÍNO ATRAVÉS DE PROCURA

Exemplo:
Puzzle de 8 peças

Representação de estado?

Representação de acção?



[Luger, 2009]

RACIOCÍNO ATRAVÉS DE PROCURA

- **Método *Gerar-e-Testar***
  1. Definir objectivo
  2. Definir estado inicial
  3. Repetir
  4. Comparar estado com o objectivo
  5. Se o objectivo tiver sido atingido então terminar
  6. Aplicar uma transformação possível ao estado actual gerando um novo estado
- Viável para problemas simples
- Não viável no caso geral
  - Procura exaustiva
  - **Problema da explosão combinatória**

MÉTODOS DE PROCURA NÃO INFORMADA

Estratégias de exploração do espaço de estados (*controlo da procura*) não tiram partido de *conhecimento do domínio do problema* para ordenar a fronteira de exploração - **procura não guiada** (**exaustiva**)

PROCURA EM PROFUNDIDADE

- Critério de exploração: maior profundidade
- Variantes
  - **PROCURA EM PROFUNDIDADE LIMITADA**
  - **PROCURA EM PROFUNDIDADE ITERATIVA**

PROCURA EM LARGURA

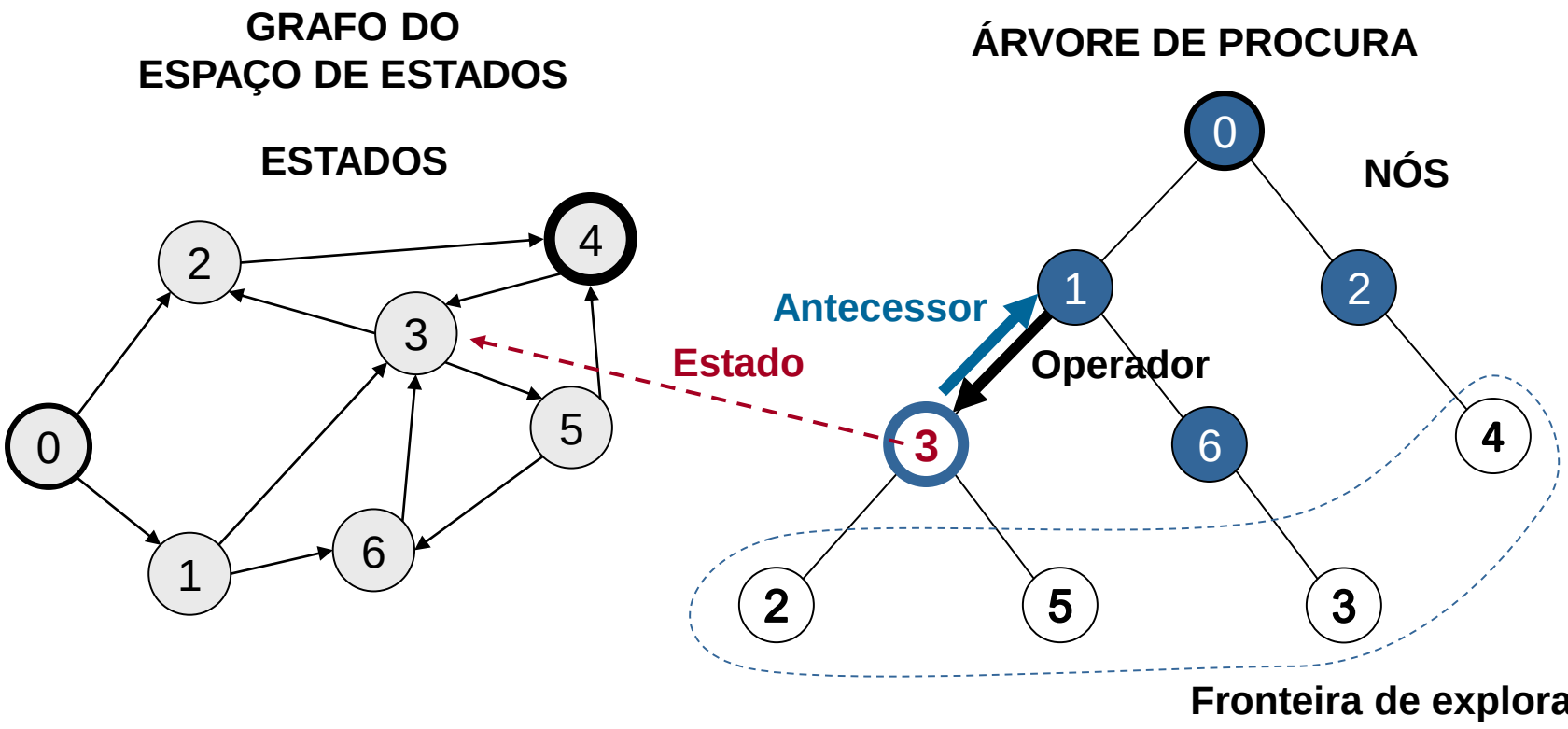
- Critério de exploração: menor profundidade

PROCURA DE CUSTO UNIFORME

- Critério de exploração: custo da solução

PROCESSO DE PROCURA

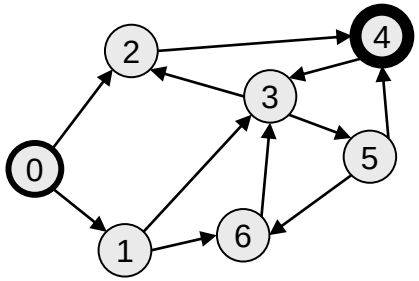
- Exploração sucessiva do espaço de estados
- Etapa de procura: **Nó**
- **Árvore de procura**
  - Raiz: Nó correspondente ao e*stado inicial*
- **Fronteira de exploração** (estrutura de dados com relação de ordem)
  - Critério de ordenação determina estratégia de controlo da procura



PROCURA EM PROFUNDIDADE

(*Depth-First Search*)

- Estratégia de controlo
  - Explorar primeiro os nós mais **recentes**



Grafo do
Espaço de Estados

Fronteira de exploração [ ]

[0]

0 [ ]

[**1, 2**]

1 [2]

[**3, 6**, 2]

3 [6,2]

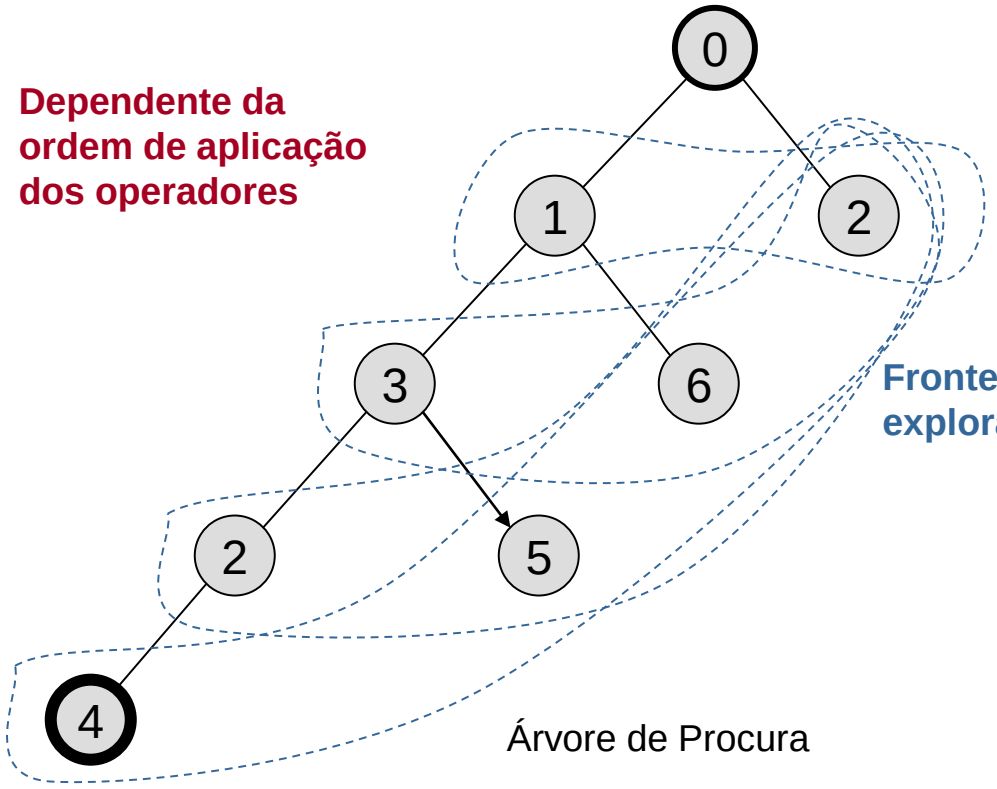
[**2, 5**, 6, 2]

2 [5,6,2]

[**4**, 5, 6, 2]

4 [5,6,2]

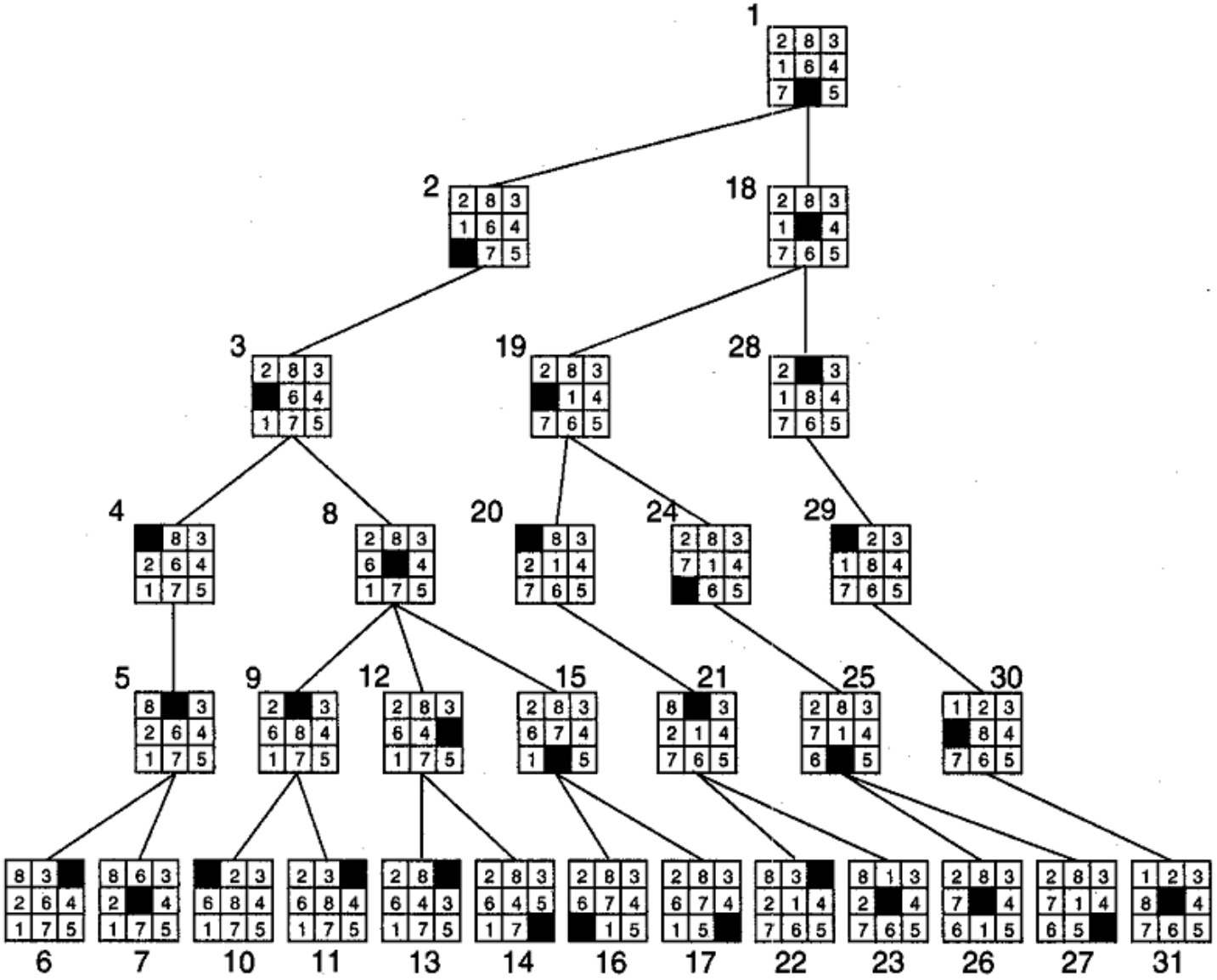
Dependente da ordem de aplicação dos operadores



Fronteira de exploração

Árvore de Procura

PROCURA EM PROFUNDIDADE

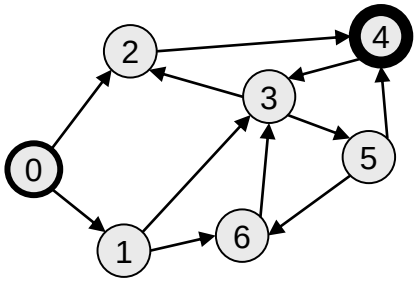


[Luger, 2009]

PROCURA EM LARGURA

(*Breadth-First Search*)

- Estratégia de controlo
  - Explorar primeiro os nós mais **antigos**



Grafo do
Espaço de Estados

Fronteira de exploração [ ]

[0]

0 [ ]

[**1, 2**]

1 [2]

[2, **3, 6**]

2 [3,6]

[3, 6, **4**]

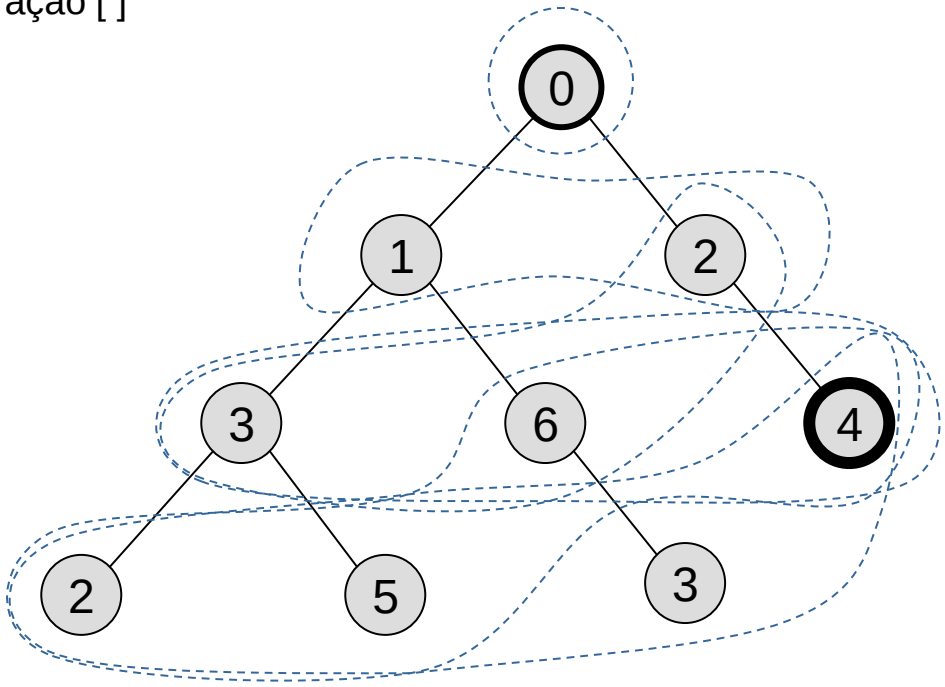
3 [6,4]

[6,4,**2,5**]

6 [4,2,5]

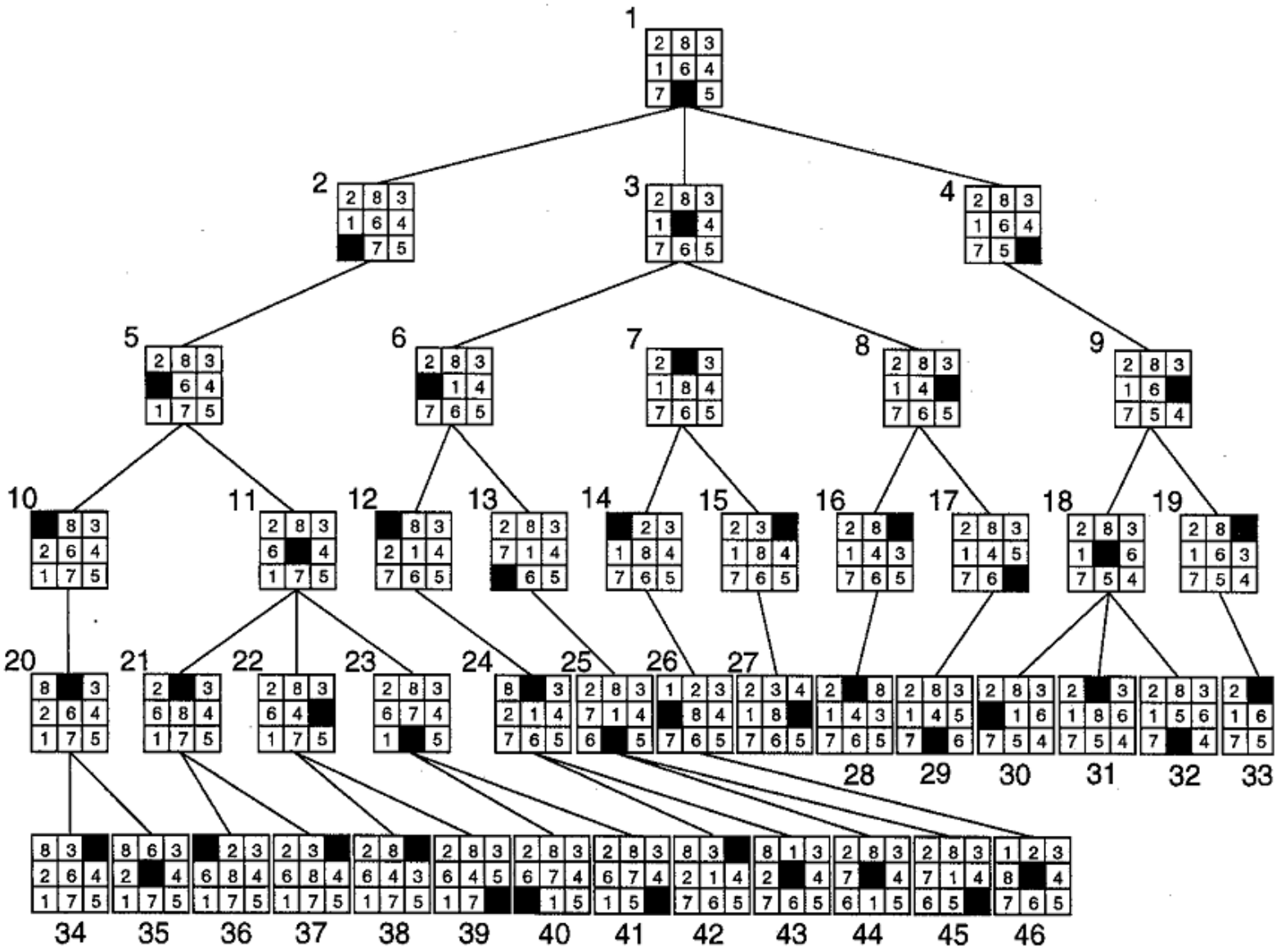
[4,2,5,**3**]

4 [2,5,3]



Árvore de Procura

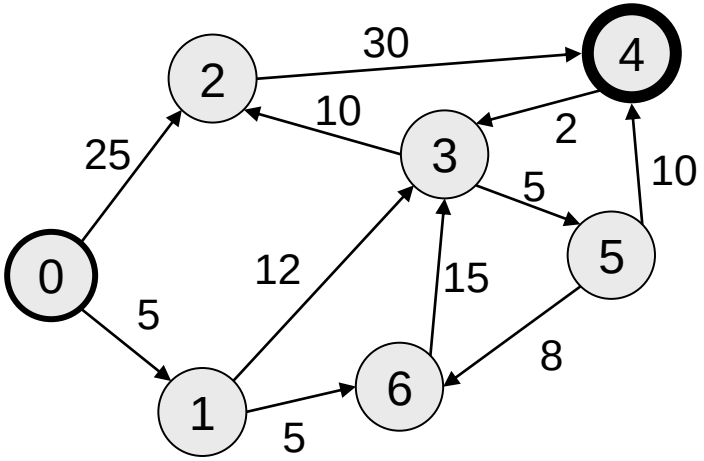
PROCURA EM LARGURA



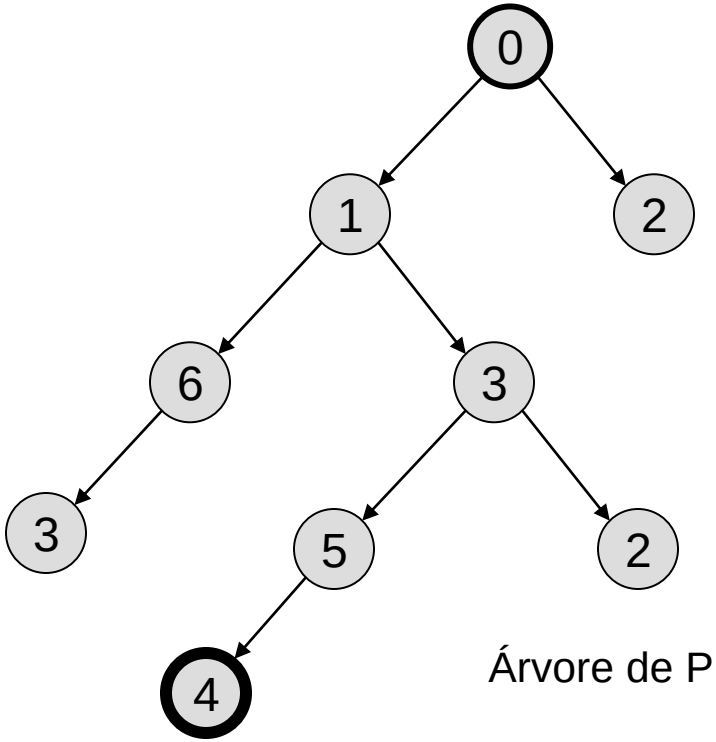
[Luger, 2009]

PROCURA DE CUSTO UNIFORME

- Estratégia de controlo
  - Explorar primeiro caminhos com menor custo
  - Custo de transição $\geq \varepsilon > 0$



Grafo do
Espaço de Estados



Árvore de Procura

PROCURA EM GRAFOS COM CICLOS

ESTADOS REPETIDOS NA ÁRVORE DE PROCURA

- Grafo do espaço de estados apresenta ciclos
- Múltiplas transições para o mesmo estado
- Acções correspondentes às transições de estado são reversíveis

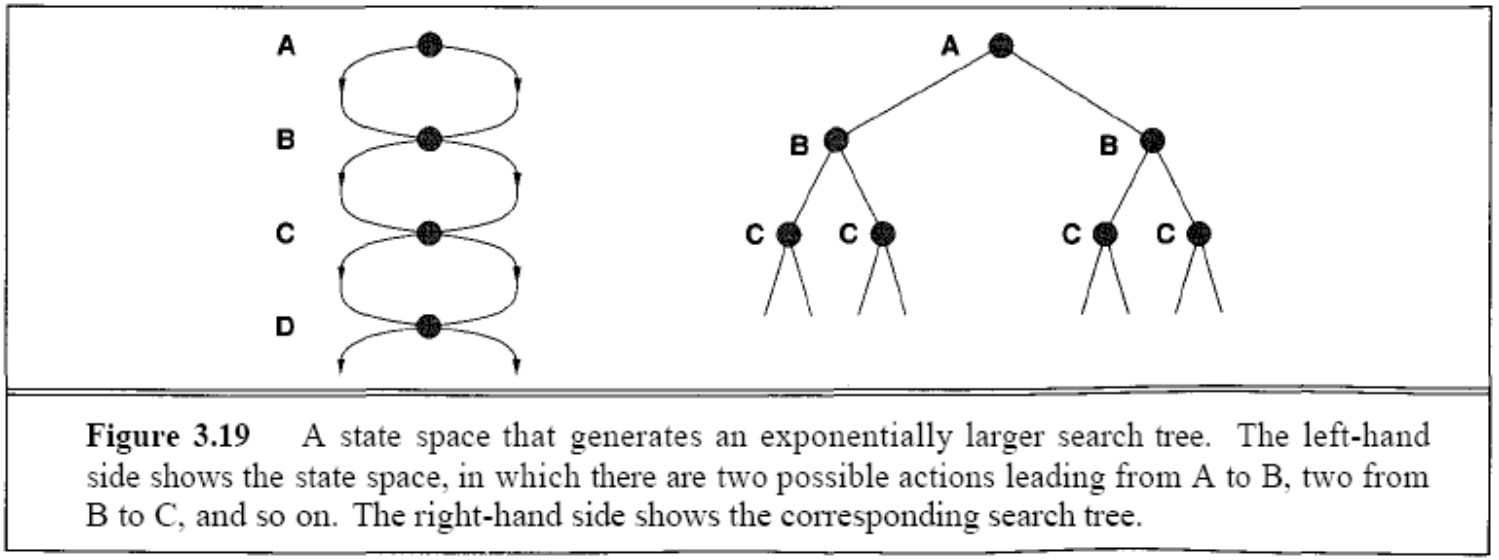


Figure 3.19 A state space that generates an exponentially larger search tree. The left-hand side shows the state space, in which there are two possible actions leading from A to B, two from B to C, and so on. The right-hand side shows the corresponding search tree.

[Russel & Norvig, 2003]

EXPANSÃO DE ESTADOS JÁ ANTERIORMENTE ANALISADOS

- **Desperdício** de recursos (**tempo**, **memória**)

PROCURA EM GRAFOS COM CICLOS

MEMÓRIA DE NÓS PROCESSADOS

- Nós gerados mas **não expandidos** (**fronteira** de exploração - ***fringe***)
  - **ABERTOS**
- Nós **expandidos**
  - **FECHADOS**

function GRAPH-SEARCH(*problem*) **returns** a solution, or failure
  initialize the frontier using the initial state of *problem*
  initialize the explored set to be empty
  loop do
    if the frontier is empty **then return** failure
    choose a leaf node and remove it from the frontier
    if the node contains a goal state **then return** the corresponding solution
    add the node to the explored set
    expand the chosen node, adding the resulting nodes to the frontier
      only if not in the frontier or explored set

[Russel & Norvig, 2010]

PROCURA EM GRAFOS COM CICLOS

- Ao gerar novo nó sucessor *noSuc* é necessário considerar:
  - $noSuc \notin Abertos \wedge noSuc \notin Fechados$
    - Inserir *noSuc* em *Abertos*
  - $noSuc \in Abertos$
    - Se *noSuc* foi atingido através de um caminho mais curto (com menor custo)
      - Remover nó anterior de *Abertos*
      - inserir *noSuc* em *Abertos*
  - $noSuc \in Fechados$
    - Se *noSuc* foi atingido através de um caminho mais curto (com menor custo)
      - Remover nó anterior de *Fechados*
      - inserir *noSuc* em *Abertos*

PROCURA EM GRAFOS COM CICLOS

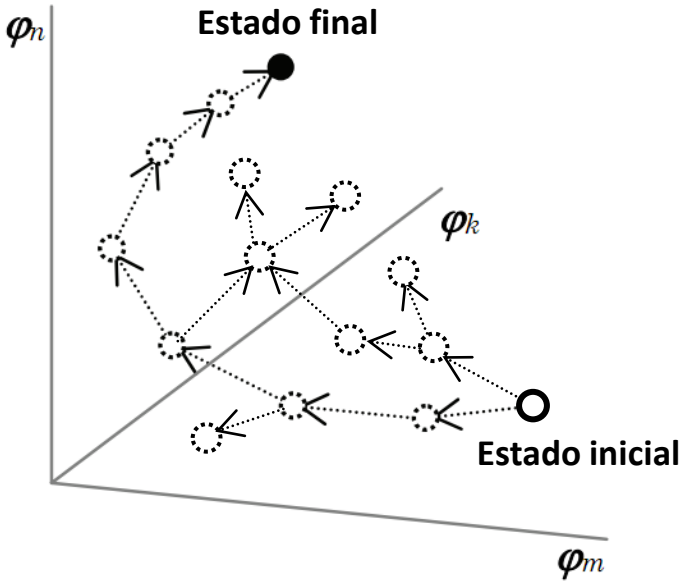
MEMÓRIA DE NÓS PROCESSADOS

- Nós gerados mas **não expandidos** (**fronteira** de exploração)
  - **ABERTOS**
- Nós **expandidos**
  - **FECHADOS**

EXPLORADOS

RACIOCÍNIO ATRAVÉS DE PROCURA

- **ESTADO**
  - Define situação
- **TRANSIÇÃO**
  - Define transformação de estado
  - **Operador**
    - Representação de **acção**
- **PROBLEMA**
  - Estado inicial
  - Estado final ou função objectivo
  - Operadores
  - Função de avaliação (custo)
- **SOLUÇÃO**
  - Percurso no espaço de estados



RACIOCÍNO ATRAVÉS DE PROCURA
COMPLEXIDADE COMPUTACIONAL

FACTOR DE RAMIFICAÇÃO
***b* –** *branching factor*

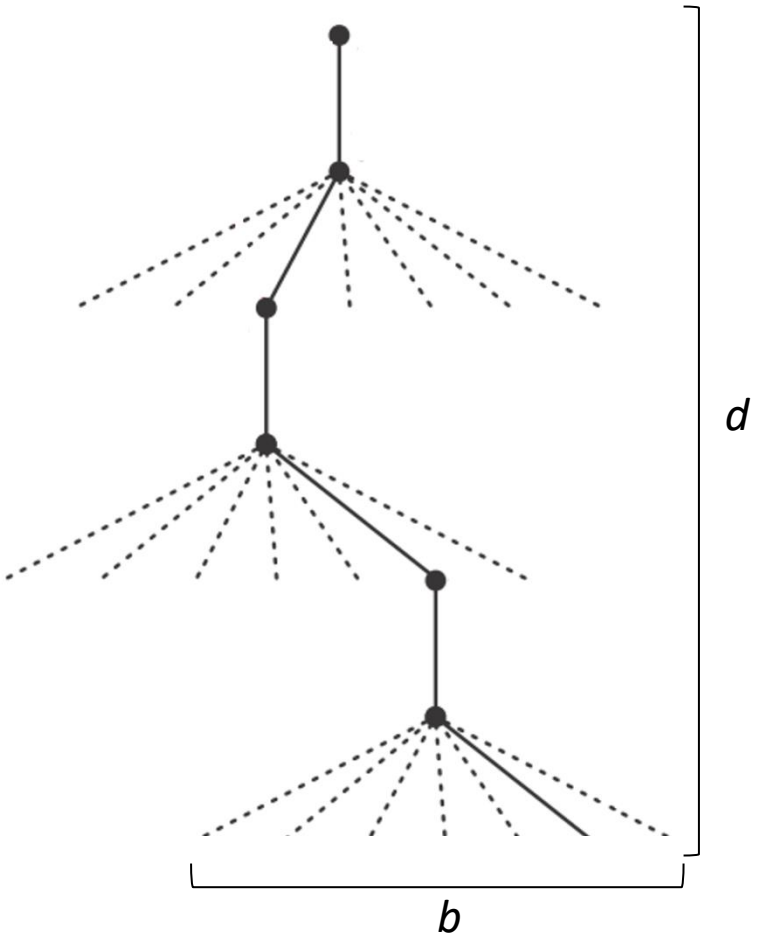
- Número máximo de sucessores para um qualquer estado

PROFUNDIDADE DA PROCURA
***d* –** *depth*

- Dimensão do percurso entre o estado inicial e o estado objectivo

Evolução exponencial do número de alternativas

$$N = b^d$$



RACIOCÍNO ATRAVÉS DE PROCURA

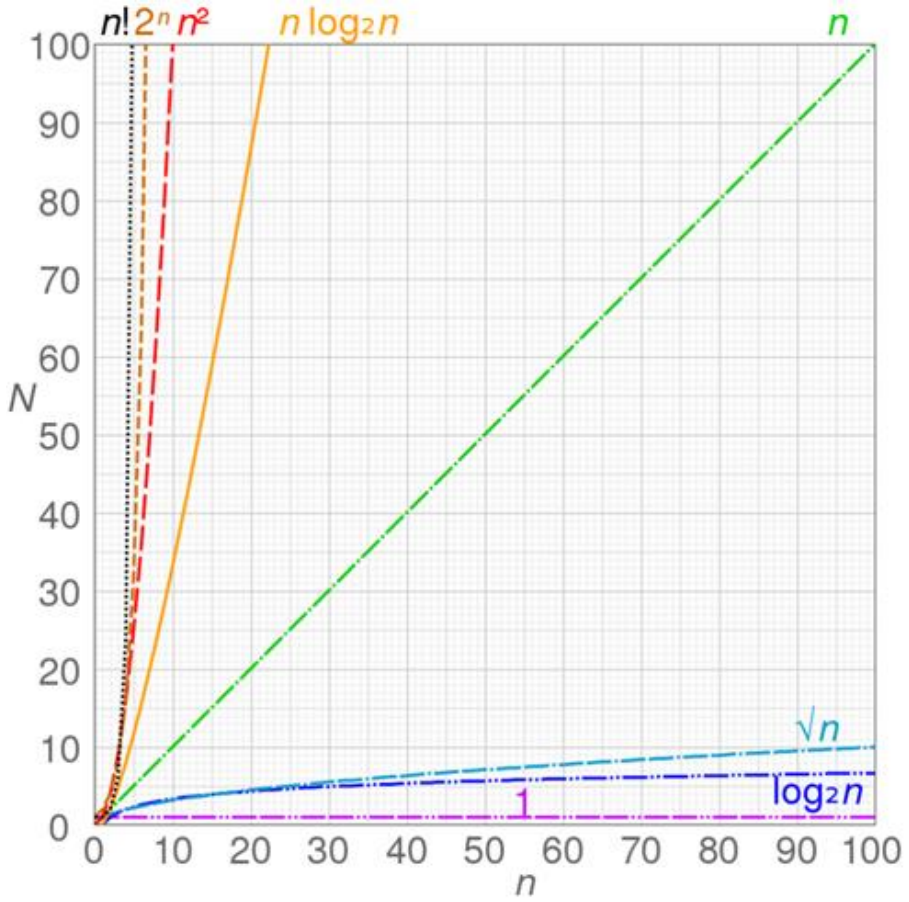
- **Aspectos a considerar num método de procura**
  - **Completo**
    - O método de procura garante que, caso exista solução, esta será encontrada
  - **Óptimo**
    - O método de procura garante que, existindo várias soluções, a solução encontrada é a melhor
  - **Complexidade**
    - **TEMPO** (complexidade temporal)
      - **Tempo necessário** para encontrar uma solução
    - **ESPAÇO** (complexidade espacial)
      - **Memória necessária** para encontrar uma solução

COMPLEXIDADE COMPUTACIONAL

Notação $f = O(g)$

$f(x)$ é de ordem $O(g(x))$ se existirem duas constantes positivas x_0 e c tal que:
$(x > x_0) : f(x) \leq cg(x)$

Exemplo:
Funções de referência de complexidade computacional



COMPLEXIDADE COMPUTACIONAL

Método de Procura	Tempo	Espaço	Óptimo	Completo
Profundidade	$O(b^m)$	$O(bm)$	Não	Não
Largura	$O(b^d)$	$O(b^d)$	Sim	Sim
Custo Uniforme	$O(b^{[C^*/\varepsilon]})$	$O(b^{[C^*/\varepsilon]})$	Sim	Sim
Profundidade Limitada	$O(b^l)$	$O(bl)$	Não	Não
Profundidade Iterativa	$O(b^d)$	$O(bd)$	Sim	Sim

b – factor de ramificação
d – dimensão da solução
m – profundidade da árvore de procura
l – limite de profundidade

C^* – Custo da solução óptima
ε – Custo mínimo de uma transição de estado ($\varepsilon > 0$)

Notação *O*

$g(n)$ é de ordem $O(f(n))$ se existirem duas constantes positivas k e N tal que:

$\forall (n > N) : g(n) \leq kf(n)$

BIBLIOGRAFIA

[Russel & Norvig, 2010]

S. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd Edition, Prentice Hall, 2010

[Nilsson, 1998]

N. Nilsson , *Artificial Intelligence: A New Synthesis*, Morgan Kaufmann 1998

[Luger, 2009]

G. Luger , *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* , Addison-Wesley, 2009

[Jaeger & Hamprecht, 2010]

M. Jaeger, F. Hamprecht, *Automatic Process Control for Laser Welding*, Heidelberg Collaboratory for Image Processing (HCI) , 2000

[Pearl, 1984]

J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, 1984

[Montemerlo, 2008]

M. Montemerlo *et al.*, *Junior: The Stanford Entry in the Urban Challenge*, Stanford Artificial Intelligence Lab, 2008