

# RESOLUÇÃO DE PROBLEMAS POR MEIO DE BUSCA

(PARTE 1)

\*Capítulo 3 (Russel & Norvig)

# Tópicos

1. Agentes para resolução de problemas
2. **Formulação** de problemas
3. Exemplos de problemas
4. **Soluções** aos problemas por busca
5. **Implementação e Avaliação**

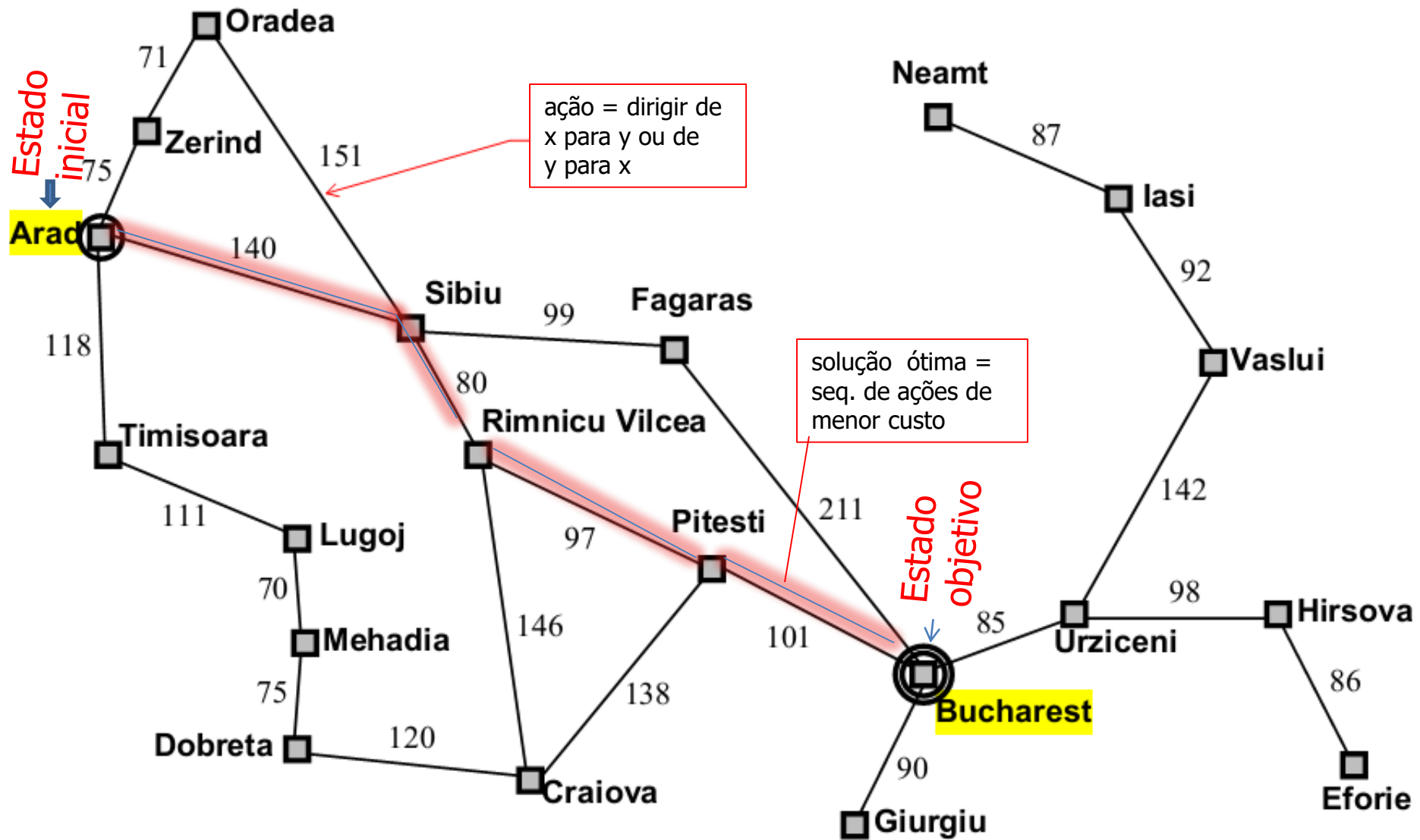
Resolução de problemas por meio de buscas

# **AGENTES PARA RESOLUÇÃO DE PROBLEMAS**

# Exemplo: Romênia

- Contexto:
  - De férias na Romênia, **estamos em Arad**
    - *(e queremos ver mais do país).*
  - Temos voo marcado para amanhã, saindo de **Bucareste**.
- Formular **objetivo**:
  - estar em Bucareste
- Formular **problema** (definir abstrações relevantes)
  - **estados**: {Arad, Timisoara, Zerind,...} // *estar em Arad, ...*
  - **ações**: Ir de Arad para Sibiu, Ir Sibiu para Fagaras, ...
- Achar **solução**:
  - **sequência de ações**, e.g., Arad, Sibiu, Fagaras, Bucareste

# Exemplo: Romênia



# FORMULAÇÃO DO PROBLEMA E DA SOLUÇÃO

- **Estado inicial:**  $Em(ARAD)$
- **Ações possíveis:** dado um estado  $s$ , quais ações são executáveis/aplicáveis em  $s$ 
  - $ações(s): S \rightarrow A$
  - $ações(ARAD) = \{IrPara(ZERIND), IrPara(TIMISOARA), \dots\}$
- **Modelo de transição ou Função sucessora**
  - $suc(s,a) : (s,a) \rightarrow s'$
  - Transição de um estado  $s$  para um estado  $s'$  pela ação  $a$
  - ex.  $suc(ARAD, IrPara(ZERIND)) = ZERIND$
- **Espaço de estados do problema**
  - É dado pelo: **estado inicial + modelo de transição**
  - É o conjunto de todos os estados alcançáveis a partir do estado inicial através de qualquer sequência de ações
  - **GRAFO**

# PROBLEMAS E SOLUÇÕES BEM FORMULADOS

- **Teste de objetivo**

- Atingiu o objetivo?
- Explícito: *Em(BUCHAREST)* ?
- Implícito: *XequeMate(x)*

- **Custo do caminho (aditivo)**

- ex. soma das distâncias, número de ações executadas, etc.
- $c(s, a, s')$  é o custo do passo/transição, supõe-se  $\geq 0$

- **Solução**

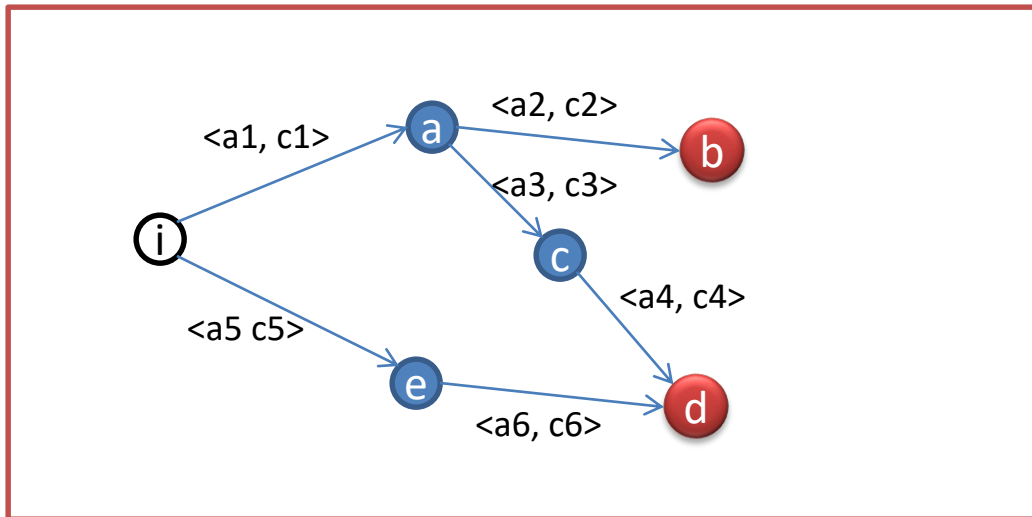
- é a sequência de ações que levam do estado inicial a um estado objetivo
- Ex. *Ir de Arad > Sibiu > Fagaras > Bucharest*

- **Solução ótima**

- é a solução de **menor custo (caminho)** entre todas as soluções.

# ESPAÇO DE ESTADOS DO PROBLEMA

- Espaço de estados: comumente representado por um **grafo** no qual
  - um nó representa um estado possível
  - um dos nós é o estado inicial
  - um ou mais nós representam estados-objetivos
  - arestas representam transições de estados resultantes de uma ação do agente
  - cada aresta tem um custo associado = custo da ação



b e d são nós objetivos  
i é o nó inicial  
<a, c> ação e custo

(não confundir com árvore/grafos de busca)



# Agente baseado em objetivos

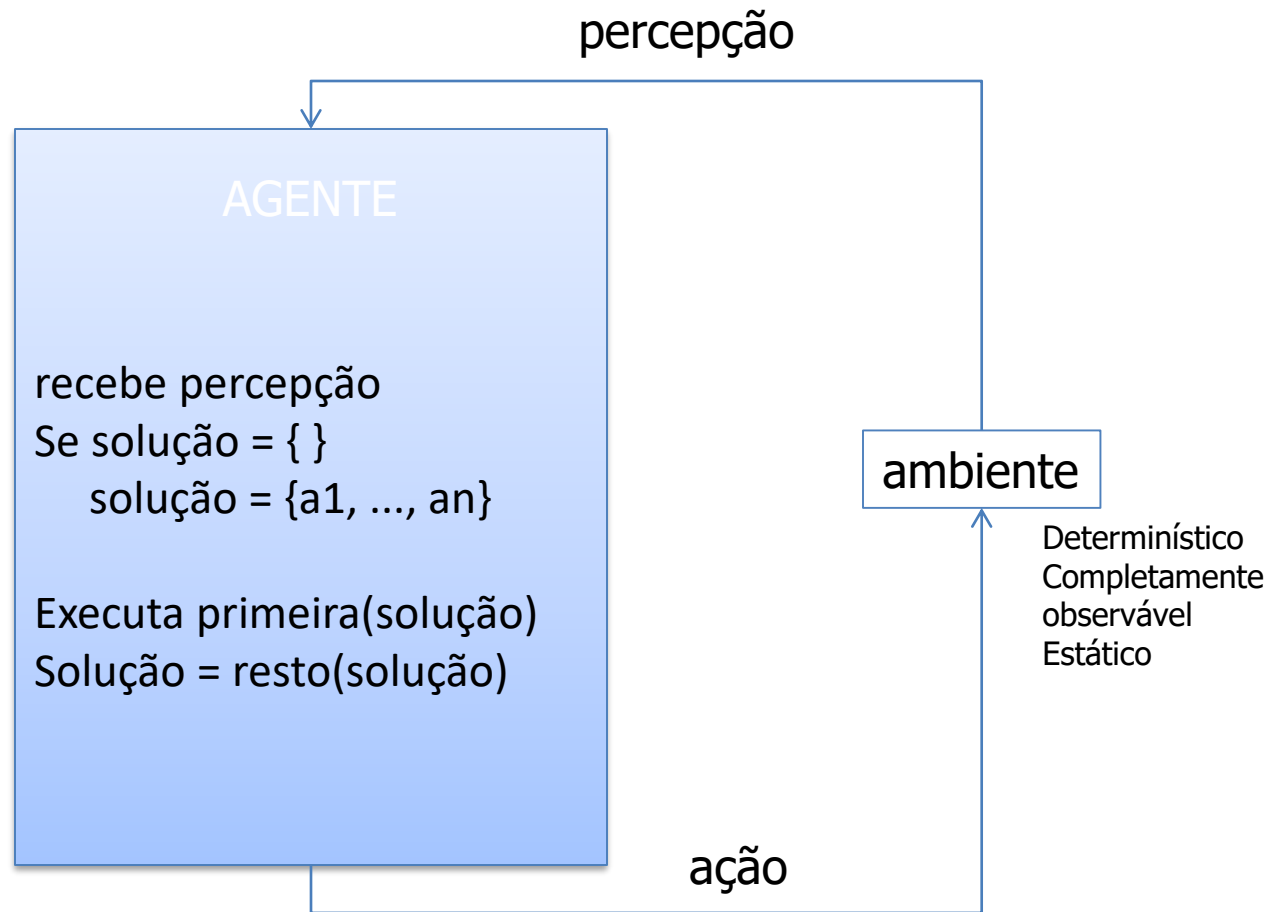
Vamos supor um agente em um ambiente **estático** e **determinístico**:

Como o ambiente é determinístico, quando o agente executa as ações não necessita tratar das percepções que dizem onde ele está a medida que ele se desloca (“tudo dá certo”)

A **solução** é calculada no início da execução e é dada por uma sequência de ações.

*A partir de uma percepção (ex.que diz onde o agente está) o agente consegue obter a solução (desde que tenha um mapa ou o ambiente seja completamente observável).*

# Agente baseado em objetivos



# Agentes de Resolução de Problemas (*goal-based agents*)

**função** AG-SIMPLES-DE-RESOLUÇÃO-DE-PROBLEMAS(*percepção*)

**retorna** uma ação

**variáveis estáticas:** *seq*, uma seqüência de ações, inicialmente vazia  
*estado*, alguma descrição do estado atual do mundo  
*objetivo*, um objetivo, inicialmente nulo  
*problema*, **uma formulação de problema (dada)**

*estado* := ATUALIZAR-ESTADO(*estado*, *percepção*)

**se** *seq* está vazia **então faça**

*objetivo* := FORMULAR-OBJETIVO(*estado*)

*problema* := FORMULAR-PROBLEMA(*estado*, *objetivo*)

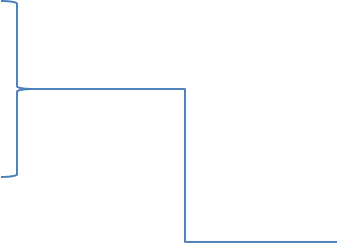
*seq* := BUSCA(*problema*)

**se** *seq* = falha **então retornar** ação nula

*ação* := PRIMEIRO(*seq*)

*seq* := RESTO(*seq*)

**retornar** *ação*



Sistema de malha-aberta: o agente ignora as percepções que chegam do ambiente

\***entradas:** *percepção*, uma percepção

Resolução de problemas por meio de buscas

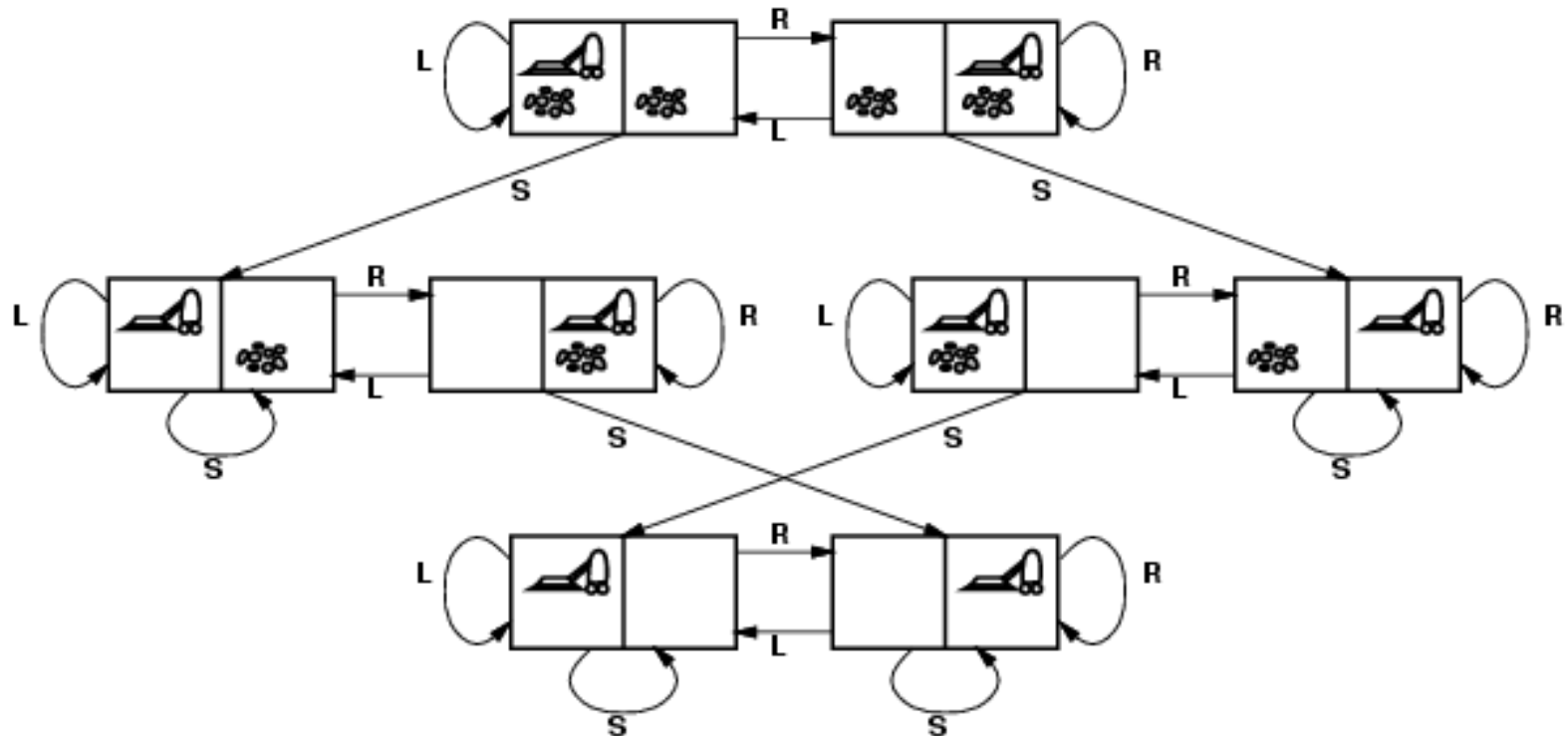
# **EXEMPLOS DE PROBLEMAS**

# Formulação de problemas

Nos slides que seguem, aborda-se a questão de como formular um problema.

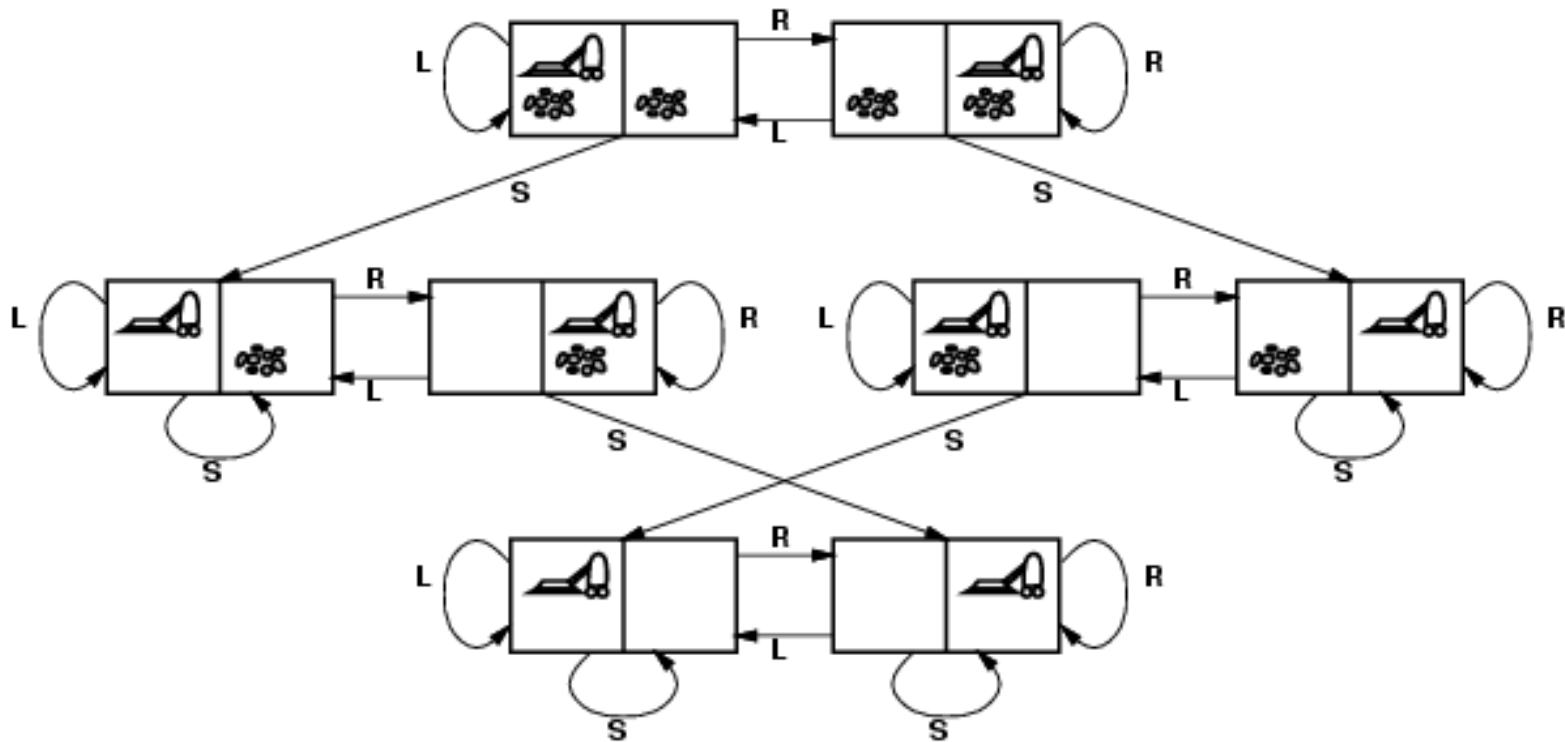
A escolha de uma boa abstração facilita a resolução do problema.

## Grafo (espaço) de estados para o mundo do aspirador



- estados?
- ações?
- estado inicial?
- testes objetivos?
- custo do caminho?

# Grafo (espaço) de estados para o mundo do aspirador



- **estados?** São 2 posições, cada posição com suja ou não:  $2 \times 2^2 = 8$
- **ações?** Left, right, suck
- **estado inicial?** Qualquer um dos 8 possíveis
- **teste objetivo?** Posição A e B limpas
- **custo do caminho?** 1 por ação executada

# Exemplo: quebra cabeças de 8 peças

|   |   |   |
|---|---|---|
| 7 | 2 | 4 |
| 5 |   | 6 |
| 8 | 3 | 1 |

Estado Inicial

|   |   |   |
|---|---|---|
|   | 1 | 2 |
| 3 | 4 | 5 |
| 6 | 7 | 8 |

Estado Objetivo

- estados?
- ações?
- testes objetivos?
- custo do caminho?



# Exemplo: quebra cabeças de 8 peças

|   |   |   |
|---|---|---|
| 7 | 2 | 4 |
| 5 |   | 6 |
| 8 | 3 | 1 |

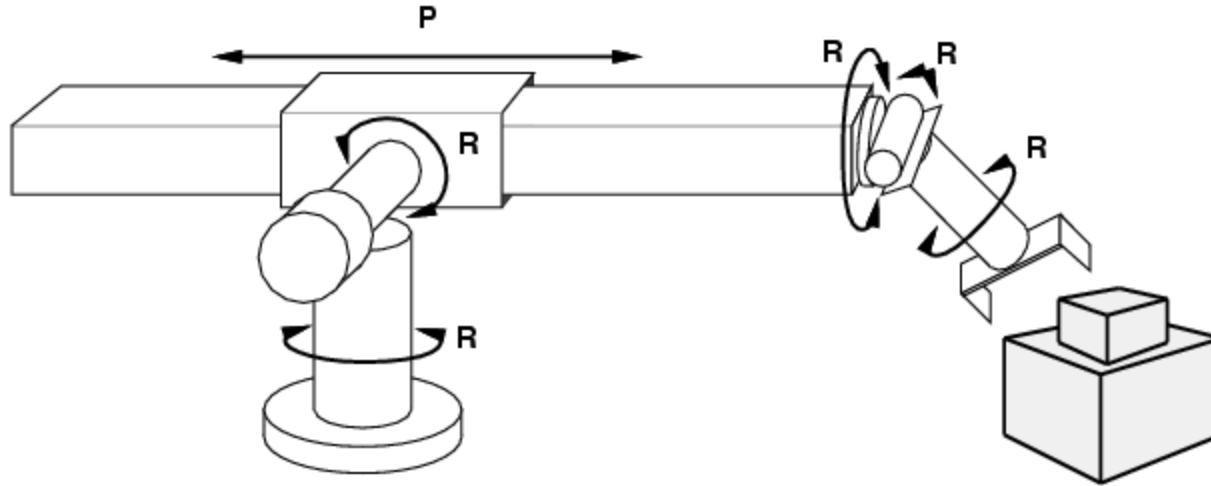
Estado Inicial

|   |   |   |
|---|---|---|
|   | 1 | 2 |
| 3 | 4 | 5 |
| 6 | 7 | 8 |

Estado Objetivo

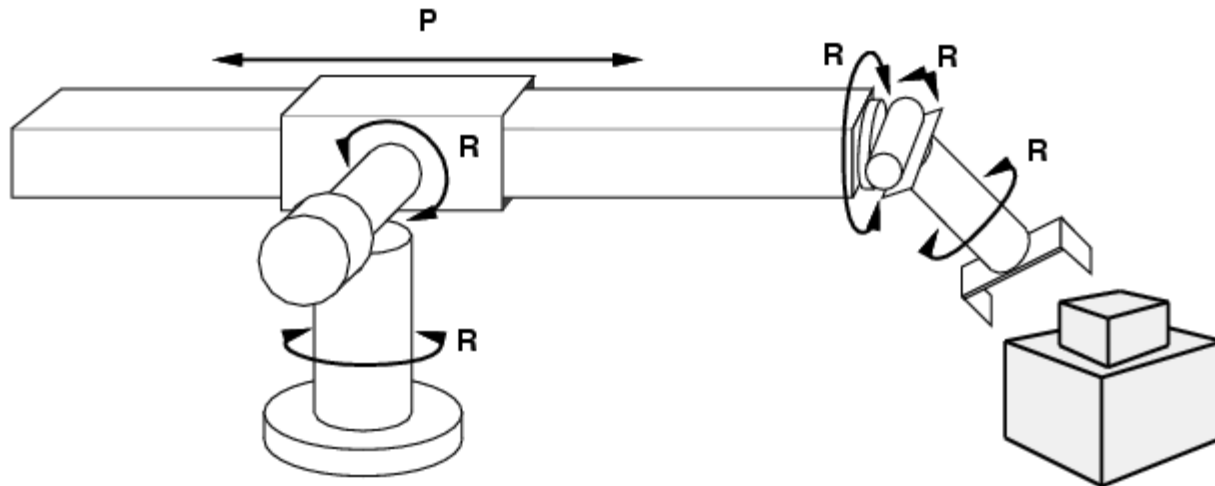
- **estados?** Localização das peças
- **ações?** Mover o vazio para esq., dir., acima, abaixo
- **Teste objetivo?** Ver figura
- **custo do caminho?** 1 por ação

# Exemplo: montagem robótica



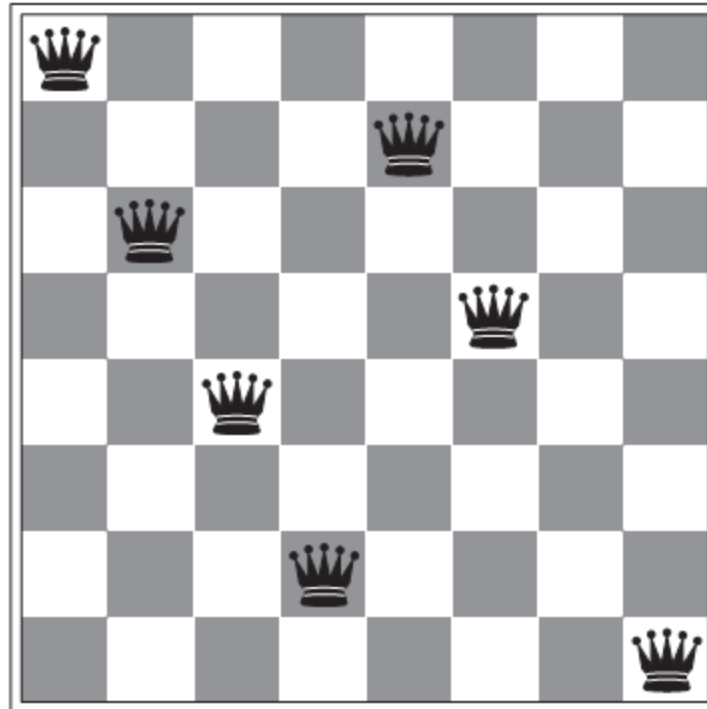
- Estados?
- Estado inicial?
- Ações?
- Teste objetivo?
- Custo do caminho?

# Exemplo: montagem robótica



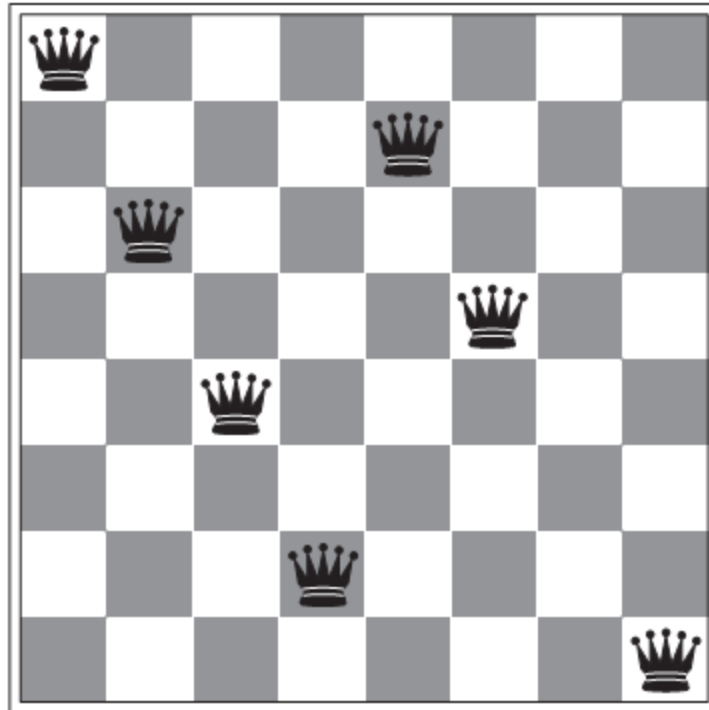
- **Estados?** Coordenadas reais dos ângulos das juntas do robô e das peças do objeto a ser montado
- **Estado inicial?** Posição afastada do suporte
- **Ações?** Movimentações contínuas das juntas do robô
- **Teste objetivo?** Montagem completa da peça
- **Custo do caminho?** Tempo de execução

# PROBLEMA DAS 8 RAINHAS



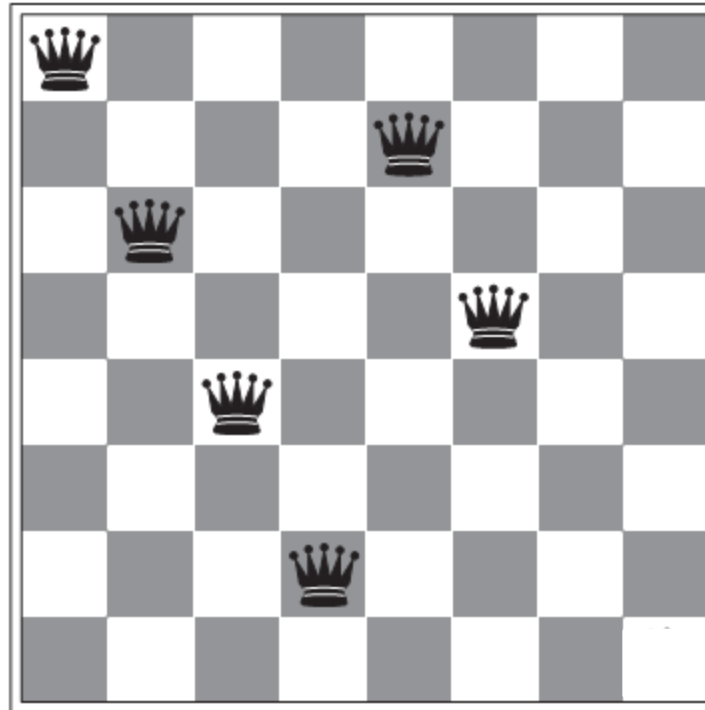
- ◉ estados?
- ◉ estado inicial?
- ◉ ações?
  
- ◉ testes objetivos?
- ◉ custo do caminho?

# PROBLEMA DAS 8 RAINHAS: formulação 1



- **estados?** Qualquer arranjo contendo de 0 a 8 rainhas no tabuleiro
- **estado inicial?** Nenhuma rainha no tabuleiro
- **ações?** Adicionar uma rainha ao tabuleiro em qualquer posição vazia
  
- **teste objetivo?** 8 rainhas no tabuleiro e nenhuma atacada
- **custo do caminho?** Não interessa – interessa apenas o goal

# PROBLEMA DAS 8 RAINHAS: formulação 2



**estado possível:**

6 rainhas nas colunas  
mais a esquerda sem  
nenhum ataque

- **estados?** Qualquer arranjo de  $n$  rainhas ( $0 \leq n \leq 8$ ), uma por coluna, nas  $n$  colunas mais a esquerda, com nenhuma rainha atacando outra
- **estado inicial?** Nenhuma rainha no tabuleiro
- **ações?** Adicionar uma rainha ao tabuleiro na  $n$ -ésima coluna mais a esquerda sem que nenhuma rainha ataque outra
- **teste objetivo?** 8 rainhas no tabuleiro e nenhuma atacada
- **custo do caminho?** Não interessa – interessa apenas o goal

# PROBLEMAS DAS 8 RAINHAS

Este exemplo ilustra a importância de bem formular um problema:

## **Formulação 1:**

Colocar as 8 rainhas incrementalmente em qualquer posição vazia.

Desvantagem: esta formulação explode em número de estados:

$$64 \times 63 \times 62 \dots \times 57 = 1.8 \times 10^{14}$$

Nesta formulação um dos estados pode ser todas as rainhas na mesma coluna (o que claramente não é um estado objetivo).

## **Formulação 2:**

Colocar as rainhas nas colunas mais a esquerda evitando ataques

$$\text{Número de estados} = 2.057$$

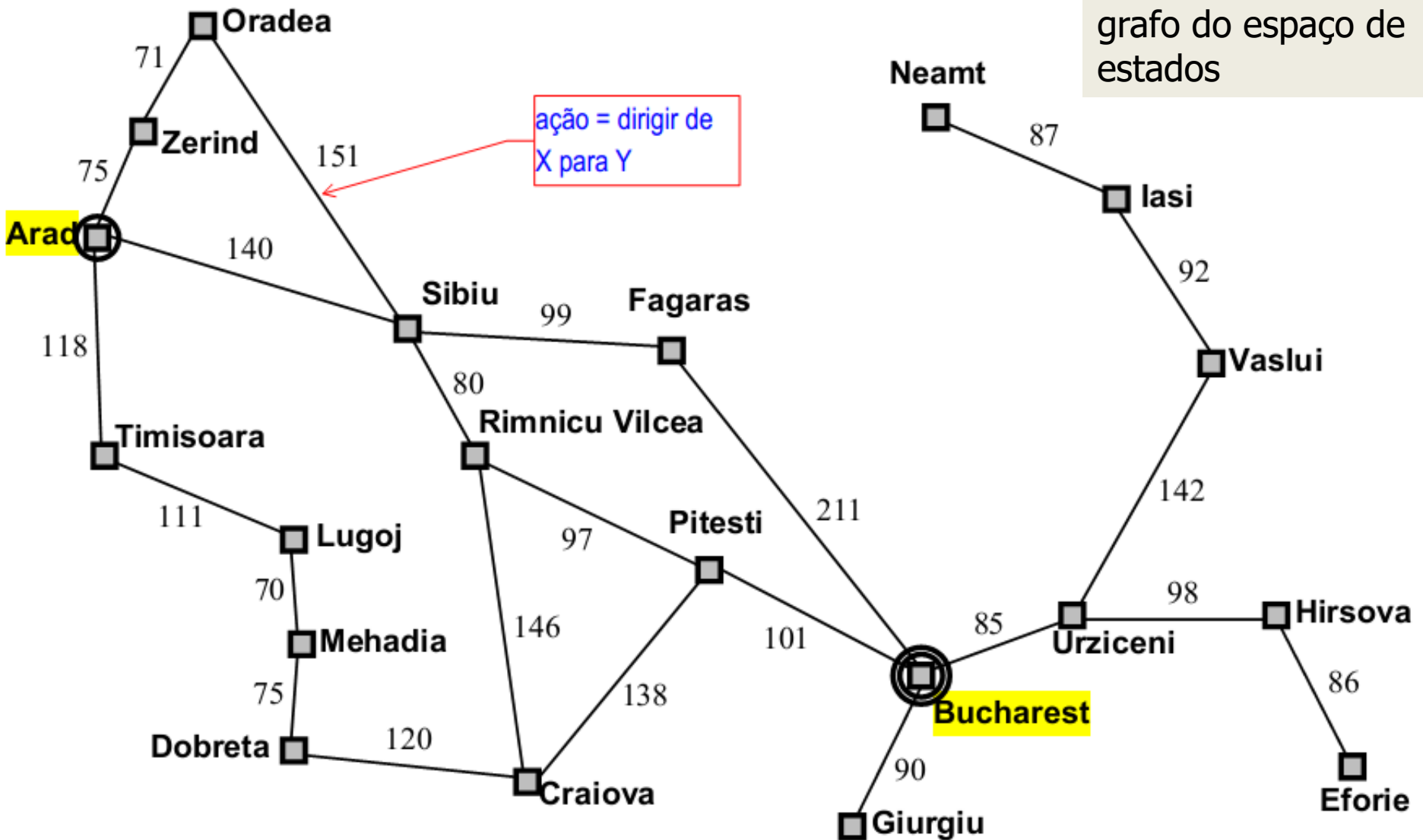
Resolução de problemas por meio de buscas

# **SOLUÇÕES AOS PROBLEMAS POR BUSCA**



# Espaço de estados: ex. Romênia

grafo do espaço de estados

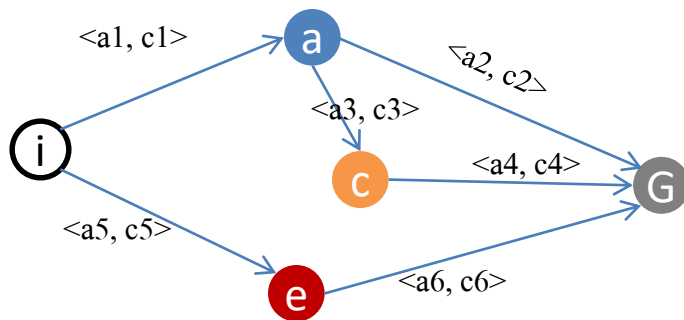


# Grafo de estados x Árvore de busca (1)

O conjunto de todos os caminhos do nó inicial até os nós objetivos de um grafo de estados podem ser representados numa árvore (árvore é um grafo direcionado acíclico)

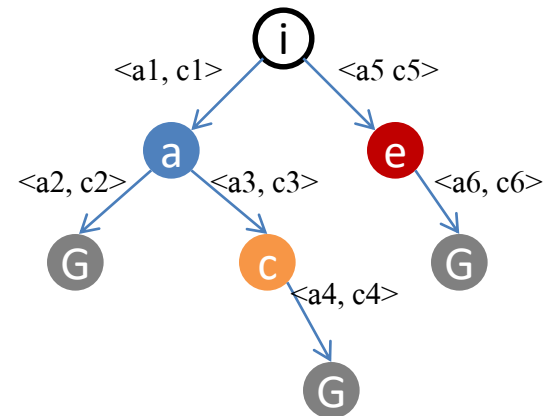
Grafos podem ser transformados em árvores por meio da **duplicação dos nós** e pela **eliminação de caminhos cíclicos** (se existirem e se forem tratados)

formulação



Espaço de estados do problema

busca de solução



Árvore gerada por um algoritmo de busca exaustiva

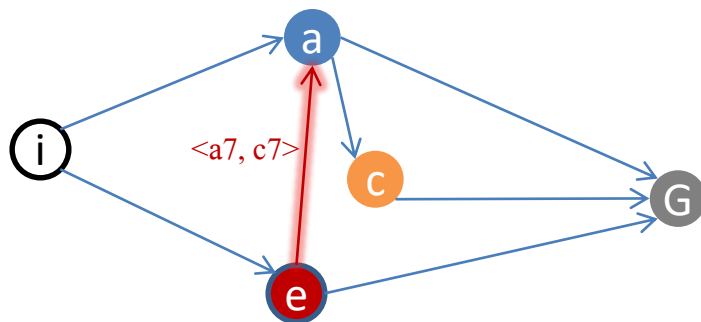
# Grafo de estados x Árvore de busca (2)

## CAMINHOS REDUNDANTES NO ESPAÇO DE ESTADOS

Um caminho redundante no espaço de estados pode aumentar exponencialmente o **espaço de busca**.

**As estratégias de busca tratam estas redundâncias de modos diferentes.**

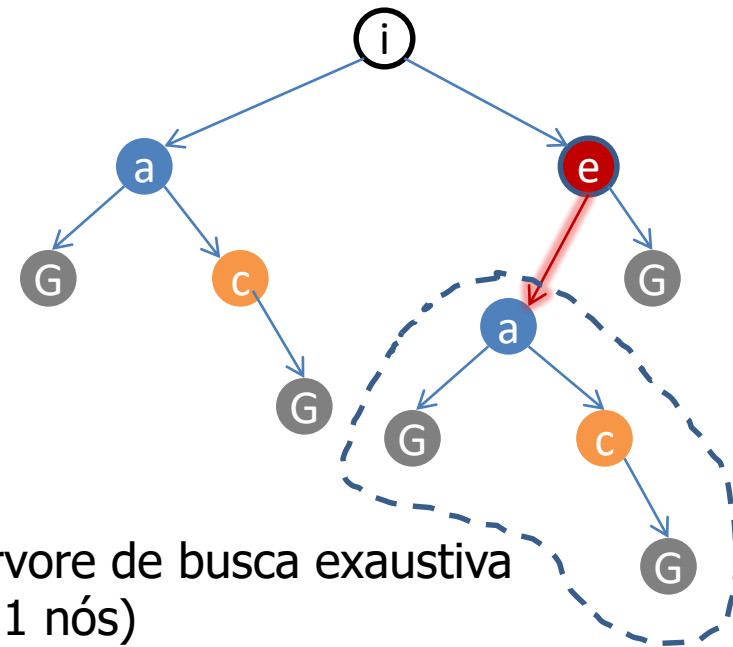
formulação



Espaço de estados do problema  
(6 estados)

Incluída a aresta (e, a)

busca de solução



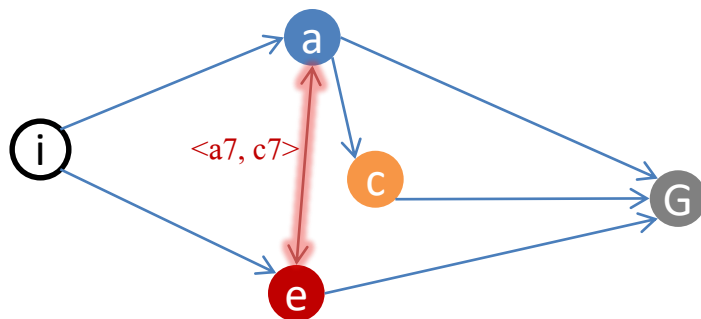
Árvore de busca exaustiva  
(11 nós)

# Grafo de estados x Árvore de busca (3)

## CAMINHOS CÍCLICOS NO ESPAÇO DE ESTADOS

Um caminho cíclico pode levar, se não controlado, a criação de um **espaço de busca infinito**.

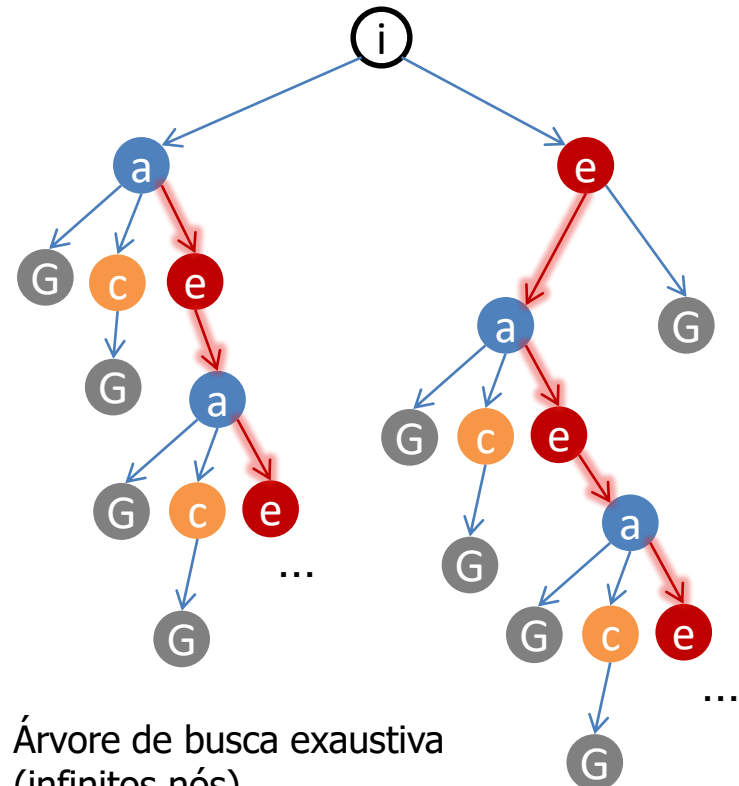
formulação



Espaço de estados do problema  
(6 estados)

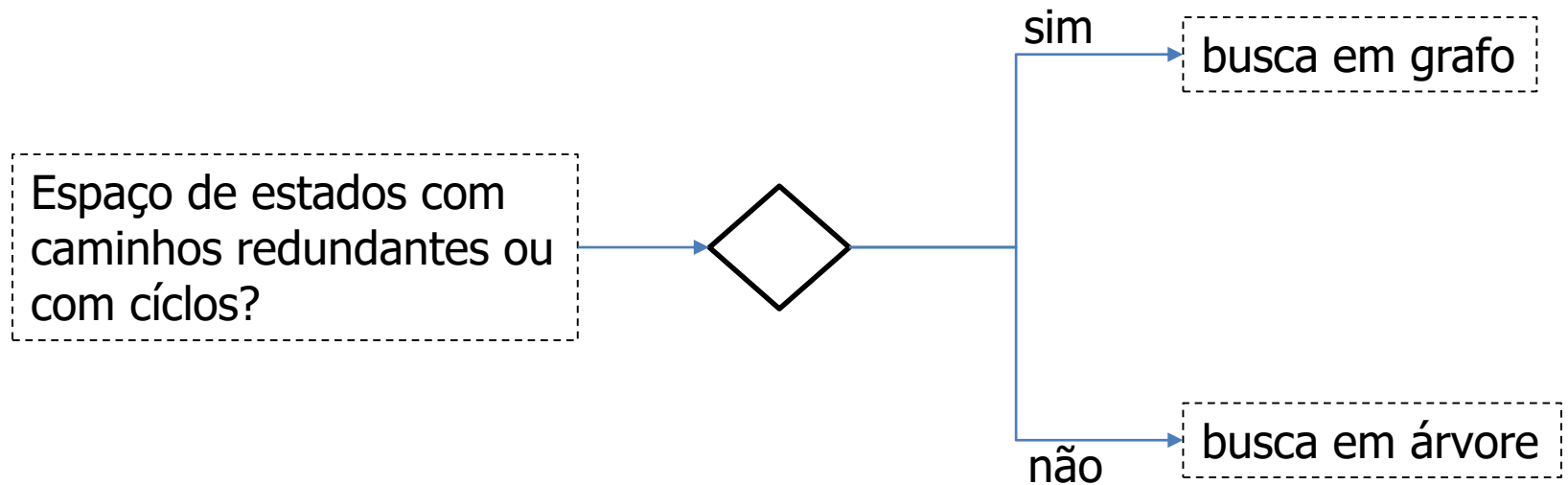
A aresta (e, a) agora é bidirecional

busca de solução

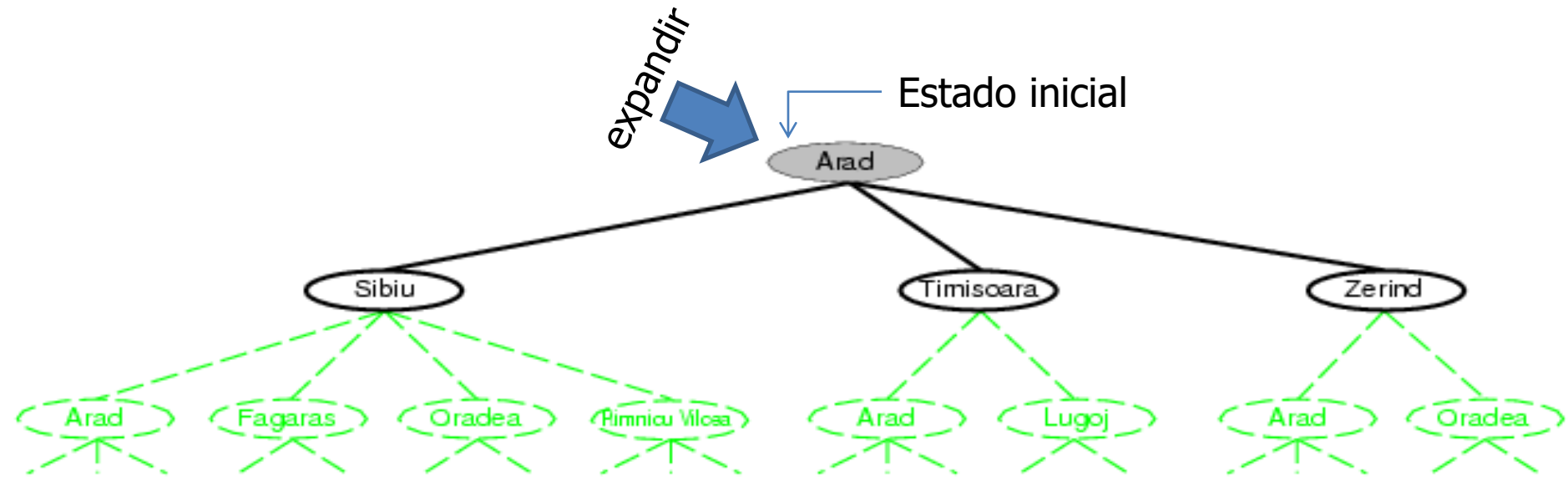


Árvore de busca exaustiva  
(infinitos nós)

# Algoritmos de busca



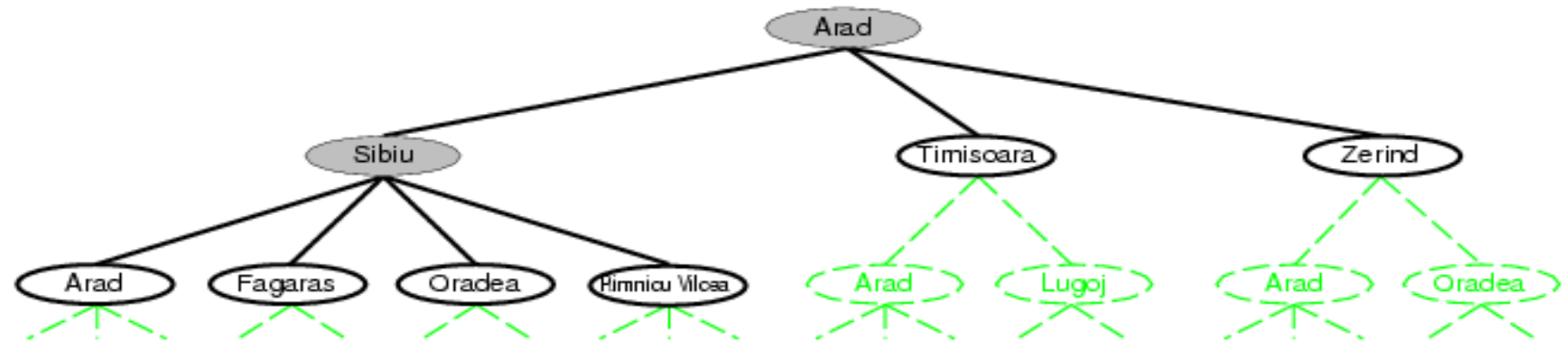
# Exemplo de busca em árvore



Expandir o nó corrente resulta na árvore em negrito

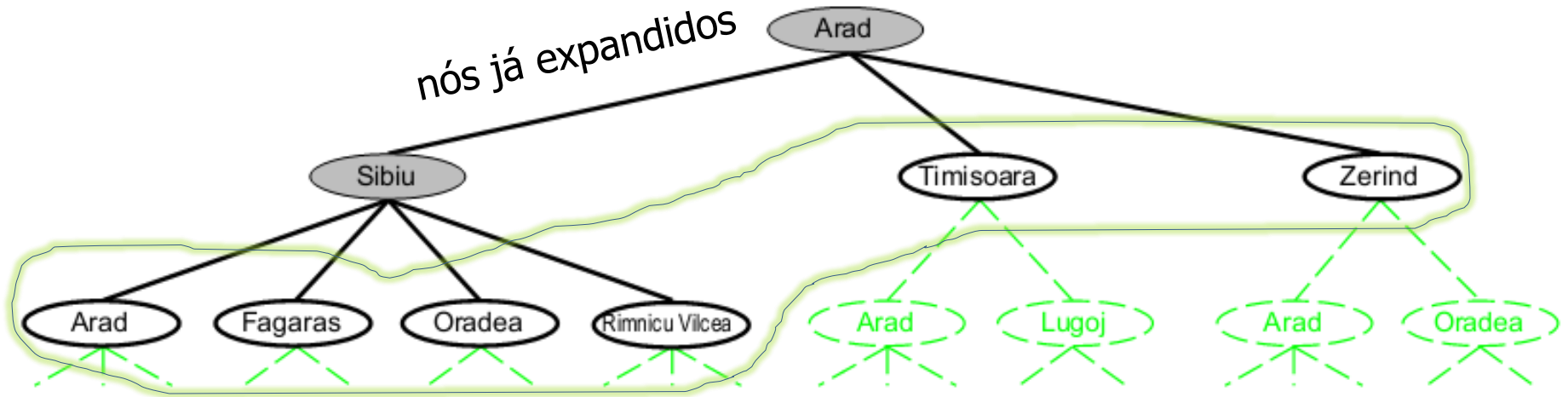
Árvore de busca parcial

# Exemplo de busca em árvore



Árvore de busca parcial

# Algoritmos de busca: nomenclatura



**Fronteira** da árvore de busca: são todos os nós folhas num dado momento.

**Estratégia de busca:** define qual nó será o próximo a ser expandido



# Algoritmos de busca em árvore

**Idéia básica:** explorar o espaço de estados de forma simulada (off-line), gerando-se estados sucessores (expandindo) até atingir o estado objetivo.

**função** BUSCA-EM-ÁRVORE(*problema*) **retorna** uma solução, ou falha  
inicialize a *fronteira* da árvore usando o estado inicial do *problema*  
**repita**  
    **se** a *fronteira* está vazia **então retornar** falha  
    **escolha** um nó folha para expansão e remova-o da *fronteira* (*estratégia*)  
    **se** o nó contém o estado objetivo **então retornar** a solução correspondente  
    expandir o nó e adicionar os nós resultantes à *fronteira* da árvore de busca

# Algoritmos de busca em GRAFO

**função** BUSCA-EM-GRAFO(*problema*) **retorna** uma solução, ou falha  
inicialize a *fronteira* da árvore usando o estado inicial do *problema*  
inicialize *explorados* com vazio  
**repita**  
    **se** a *fronteira* está vazia **então retornar** falha  
    **escolha** um nó folha para expansão e remova-o da *fronteira* (*estratégia*)  
    **se** o nó contém o estado objetivo **então retornar** a solução correspondente  
    adicione o nó aos *explorados*  
    expandir o nó e adicionar os nós resultantes à *fronteira* da árvore de busca  
    somente se não estiverem em *explorados*

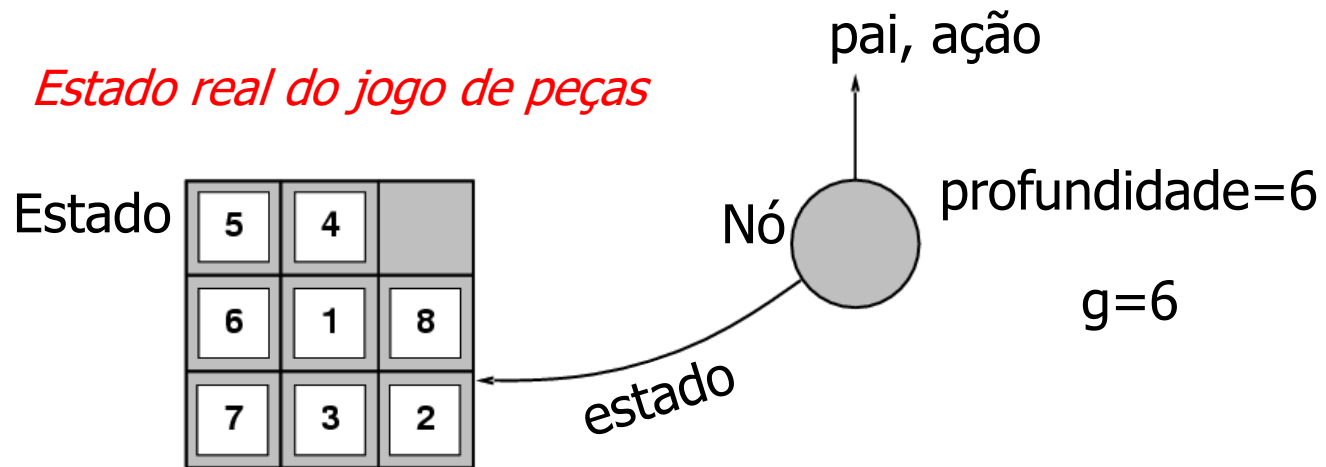
Soluciona o problema de redundância da busca em árvore por meio de uma lista de nodos já explorados

# Implementação: estados vs. nós

Um **estado** é a abstração de uma configuração física do mundo

Um **nó** é uma **estrutura de dados** e faz parte de uma árvore de busca.

Logo, um estado é representado por um ou mais nós em uma árvore de busca.

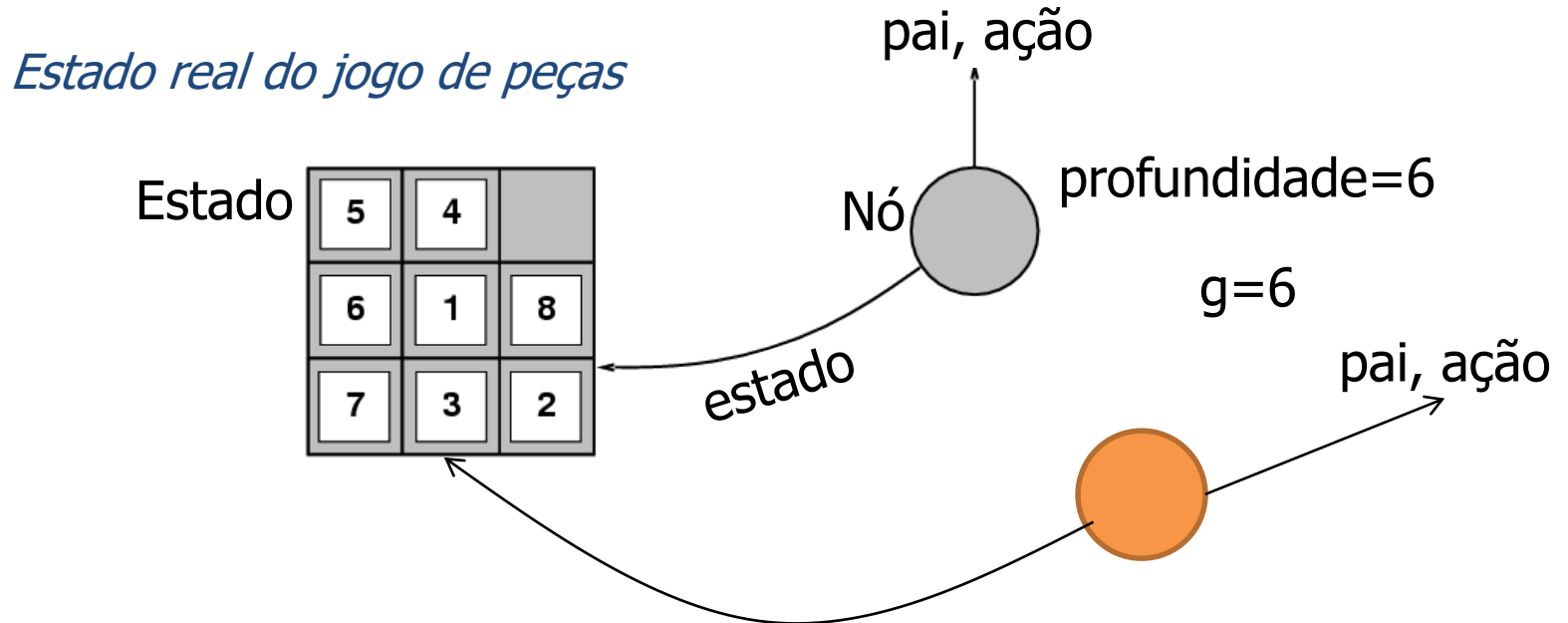


A função **EXPANDIR** cria novos nós, preenchendo os diversos campos do nó e utiliza a função **SUCCESSOR** do problema para criar os estados correspondentes aos nós.

# Implementação: estados vs. nós

Dois nós diferentes podem apontar para o mesmo estado  
*ex. quando há redundância de caminhos.*

**Número de nós** da árvore de busca **pode ser maior** do que o número de estados do espaço de estados



# Estratégia de busca

- Uma **estratégia de busca** é definida pela escolha do próximo nó a ser expandido
- Logo, os algoritmos
  - busca em árvore
  - busca em grafo
  - representam uma família de algoritmos

# AVALIAÇÃO DAS ESTRATÉGIAS DE BUSCA

| Tipos de avaliação           | Significado  |
|------------------------------|--|
| Completa                     | garante encontrar <u>uma solução</u> se ela existe?  |
| Ótima                        | garante encontrar a <b>solução de menor custo</b> (se existe solução)?                             |
| Complexidade de <b>tempo</b> | medido pelo <b>número de nós gerados</b> para executar a busca                                     |
| Complexidade <b>espacial</b> | medido pelo <b>número máximo de nós</b> mantidos em memória em algum instante da execução da busca |

# AVALIAÇÃO DAS ESTRATÉGIAS DE BUSCA

Variáveis que afetam as complexidades temporal e espacial

| Variável                                     | Significado   |
|--|---|
| <b>b</b> = <i>branching</i> =<br>ramificação | número máximo de sucessores (dentre todos os nós da árvore)   |
| <b>d</b> = <i>depth</i> =<br>profundidade    | profundidade do nó objetivo mais raso<br>(não é necessariamente o nó objetivo ótimo)  |
| <b>m</b><br>(maximum len.)                   | tamanho máximo de caminho entre todos os caminhos do <b>espaço de estados</b> (pode ser $\infty$ ; não leva em conta o custo do caminho) →<br>corresponde à prof. da árvore de busca* |

\*eliminando-se caminhos cíclicos

# Exemplo das medidas

