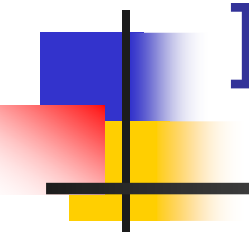


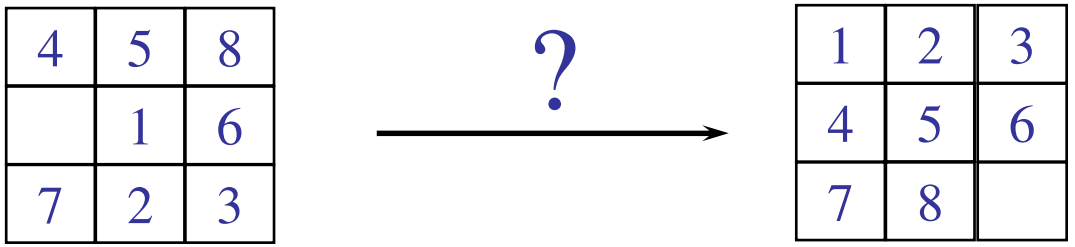
Inteligência Artificial

Prof. Tiago A. E. Ferreira

Aula 5 – Resolvendo Problemas

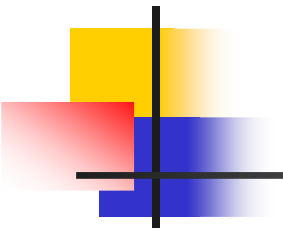
Agente solucionador de problemas (guiado por objetivo)

- O agente reativo
  - Escolhe suas ações com base apenas nas percepções atuais
    - não tem estado interno
    - portanto, não pode pensar no futuro
  - Não sabe "aonde vai"



- O Agente *solucionador de problemas*
  - busca uma *seqüência de ações* que leve a estados desejáveis (*objetivos*).

Agentes solucionadores de problemas



- O que é um problema em I.A.?
- Como formulá-lo?
- Como buscar a solução do problema?
  - Busca cega
  - Busca heurística
- Quais são os tipos de problemas?
- Quais são as aplicações?

Solução de Problemas: definições

Um problema em IA é definido em termos de...

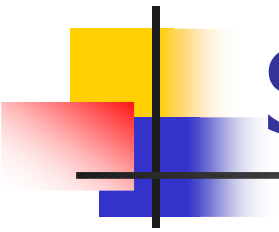
1) um espaço de estados possíveis, incluindo:

- um estado inicial
- um (ou mais) estado final = objetivo
- exemplo 1: dirigir de Recife a Cajazeiras
  - espaço de estados: *todas as cidades da região*
- exemplo 2: jogo de 8-números
  - início:

4	5	8
	1	6
7	2	3

fim:

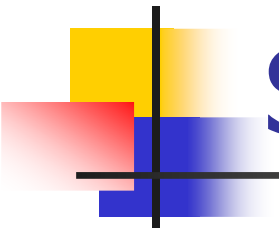
1	2	3
4	5	6
7	8	



Solução de Problemas: definições

2) um conjunto de ações (ou operadores) que permitem passar de um estado a outro

- ex1. dirigir de uma cidade a outra
- ex2. mover uma peça do jogo de n-números (*n-puzzle*)



Solução de Problemas: definições

- Definição do objetivo:
  - propriedade abstrata
    - ex., condição de xeque-mate no Xadrez
  - conjunto de estados finais do mundo
    - ex., estar em Cajazeiras
- Solução:
  - caminho (seqüência de *ações* ou *operadores*) que leva do estado inicial a um estado final (objetivo).
- Espaço de Estados:
  - conjunto de todos os estados alcançáveis a partir do estado inicial por qualquer seqüência de ações.

Solucionando o problema: formulação, busca e execução

- Formulação do problema e do objetivo:
  - quais são os ***estados*** e as ***ações*** a considerar?
  - qual é (e como representar) o **objetivo**?

- Busca (solução do problema):
  - processo que gera/analisa seqüências de ações para alcançar um objetivo
  - *solução* = caminho entre estado inicial e estado final.
  - *custo do caminho* = qualidade da solução

- Execução:
  - executar a solução completa encontrada, ou
  - intercalar execução com busca: planejamento

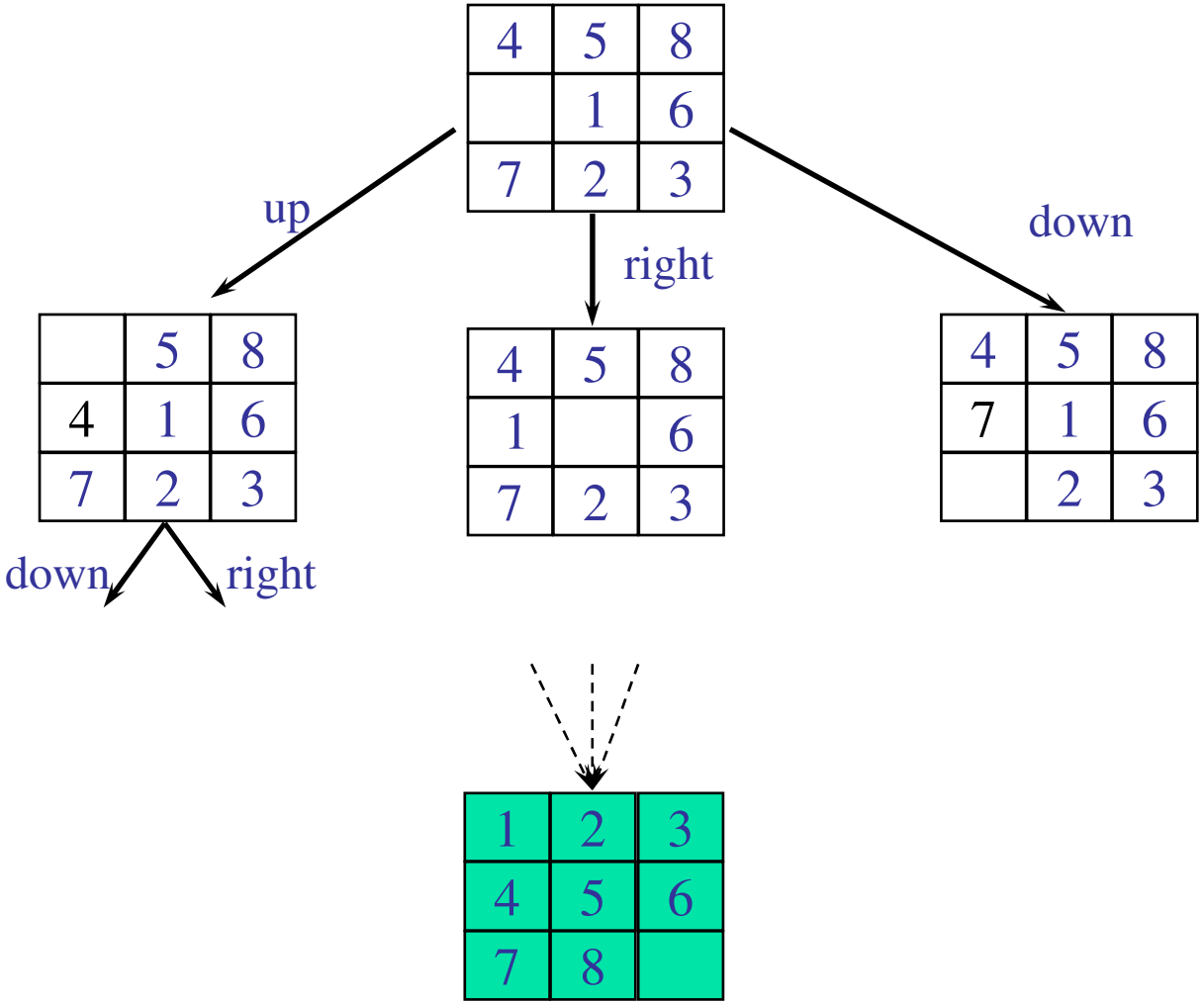
Exemplos de formulação de problemas

- Jogo de 8 números:
  - **estados** = cada possível configuração do tabuleiro
  - **estado inicial** = qualquer um dos estados possíveis
  - **teste de término** = ordenado, com branco na posição [3,3]
  - **operadores** = mover branco (esquerda, direita, para cima e para baixo)
  - **custo do caminho** = número de passos da solução

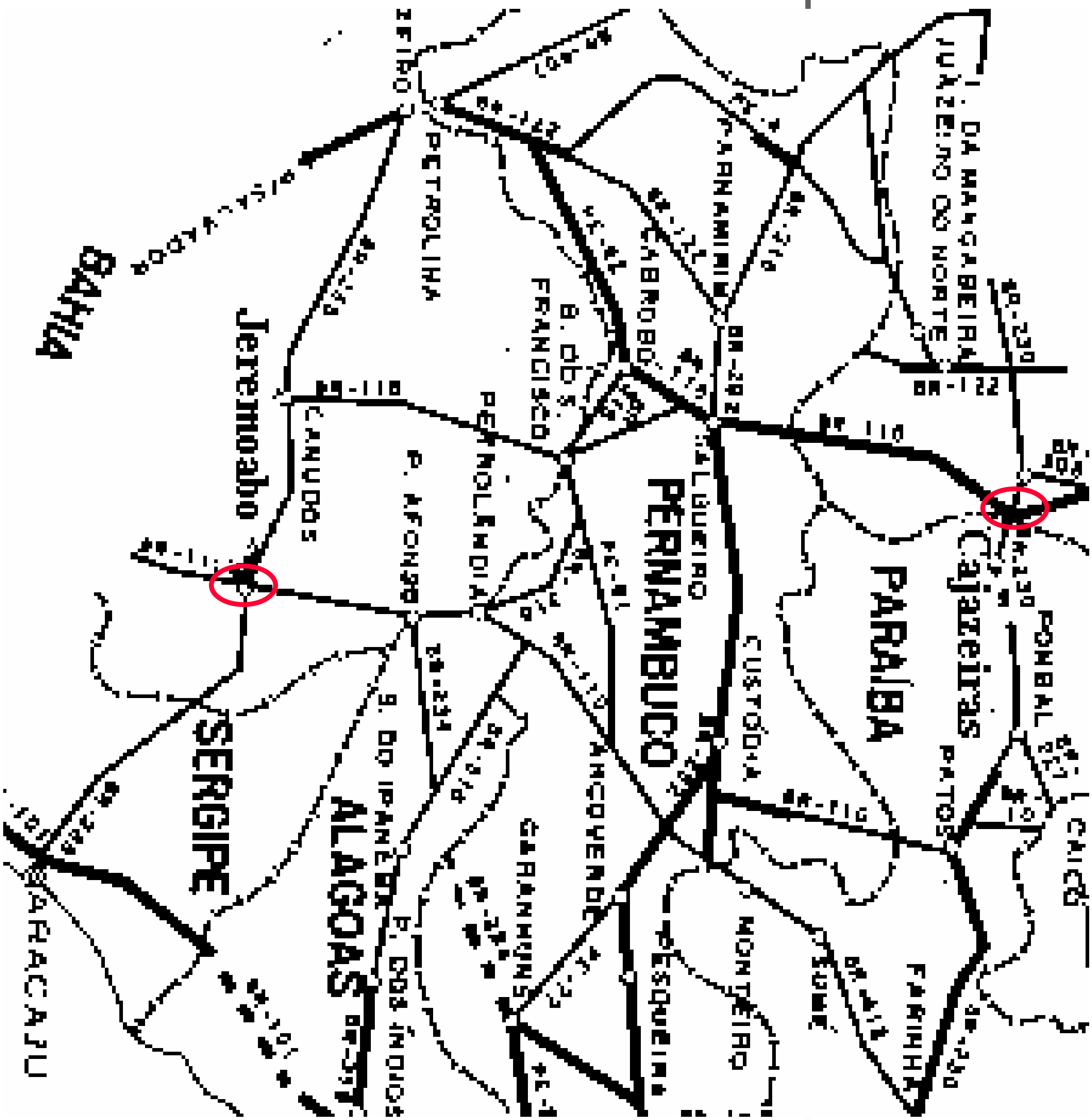
Exemplos de formulação de problemas

- Ida para Cajazeiras:
  - **estados** = cada possível cidade do mapa
  - **estado inicial** = Jeremoabo
  - **teste de término** = estar em Cajazeiras
  - **operadores** = dirigir de uma cidade para outra
  - **custo do caminho** = número de cidades visitadas, distância percorrida, tempo de viagem, grau de divertimento, etc

Árvore de busca para o "jogo dos 8 números"



Jeremoabo - Cajazeiras



Medida de Desempenho na Busca

- Desempenho de um algoritmo de busca:
  - 1. O algoritmo encontrou alguma solução?
  - 2. É uma boa solução?
    - **custo de caminho** (qualidade da solução)
  - 3. É uma solução computacionalmente barata?
    - **custo da busca** (tempo e memória)
- Custo total
  - custo do caminho + custo de busca
- Espaço de estados grande:
  - compromisso (conflito) entre a melhor solução e a solução mais barata

Custo diferente => Solução diferente

- Função de *custo de caminho*

  (1) número de cidades visitadas,

  (2) distância entre as cidades,

  (3) tempo de viagem, etc.

- Solução mais barata:

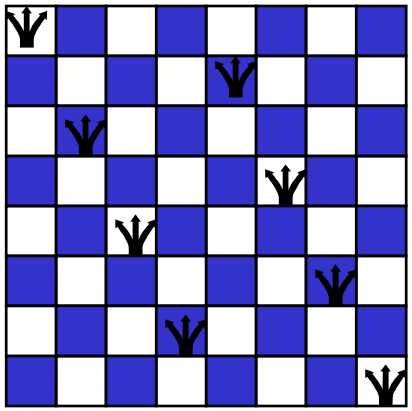
  (1) Canudos, Belém do S. Francisco, Salgueiro, ...

  (2) Canudos, Belém do S. Francisco, Cabrobó, ...

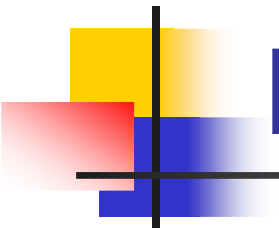
  (3) Canudos, Juazeiro, Pretrolina, Cabrobó, Salgueiro

Importância da formulação

- Jogo das 8 Rainhas
  - dispor 8 rainhas no tabuleiro de forma que não possam se "atacar"
    - não pode haver mais de uma rainha em uma mesma linha, coluna ou diagonal
  - somente o custo da busca conta
    - não existe custo de caminho

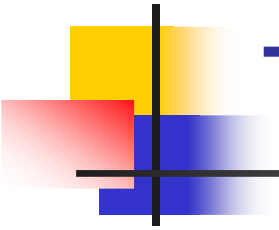


- Existem diferentes estados e operadores possíveis
  - essa escolha pode ter conseqüências boas ou nefastas na complexidade da busca ou no tamanho do espaço de estados



Formulações para 8 Rainhas

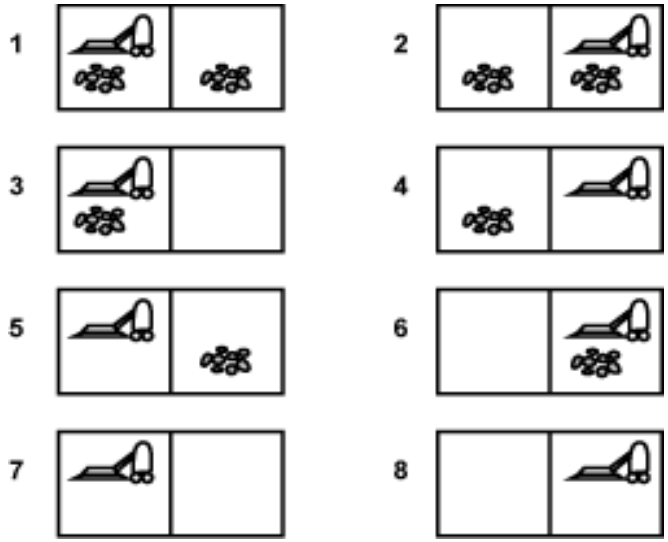
- Formulação A
  - estados: qualquer disposição com n (n ≤ 8) rainhas
  - operadores: adicionar uma rainha a qualquer quadrado
  - 64^8 possibilidades: vai até o fim para testar se dá certo
- Formulação B
  - estados: disposição com n (n ≤ 8) rainhas sem ataque mútuo (teste gradual)
  - operadores: adicionar uma rainha na coluna vazia mais à esquerda em que não possa ser atacada
  - melhor (2057 possibilidades), mas pode não haver ação possível
- Formulação C
  - estados: disposição com 8 rainhas, uma em cada coluna
  - operadores: mover uma rainha atacada para outra casa na mesma coluna



Tipos de Problemas

- Existem 4 tipos
  - Problemas de estado único (o mais tratado por busca!)
  - Problemas de múltiplos estados
  - Problemas contingenciais
  - Problemas exploratórios
- Tudo depende do conhecimento do agente...

Exemplo: Agente Aspirador de Pó



- Formulação do problema:
  - estados = mostrados na figura
  - estado inicial = qualquer um dos estados possíveis
  - teste de término = os dois quartos limpos
  - operadores = mover direita, mover esquerda, aspirar
  - custo do caminho = quantidade de ações realizadas

Tipo 1: Problemas de estado único

- Conhecimento do agente
  - sabe em que estado está (mundo totalmente acessível)
  - sabe o efeito de cada uma de suas ações

    => sabe onde está depois de uma seqüência qualquer de ações

- Cada ação leva a um único estado
  - ex. aspirador de pó:

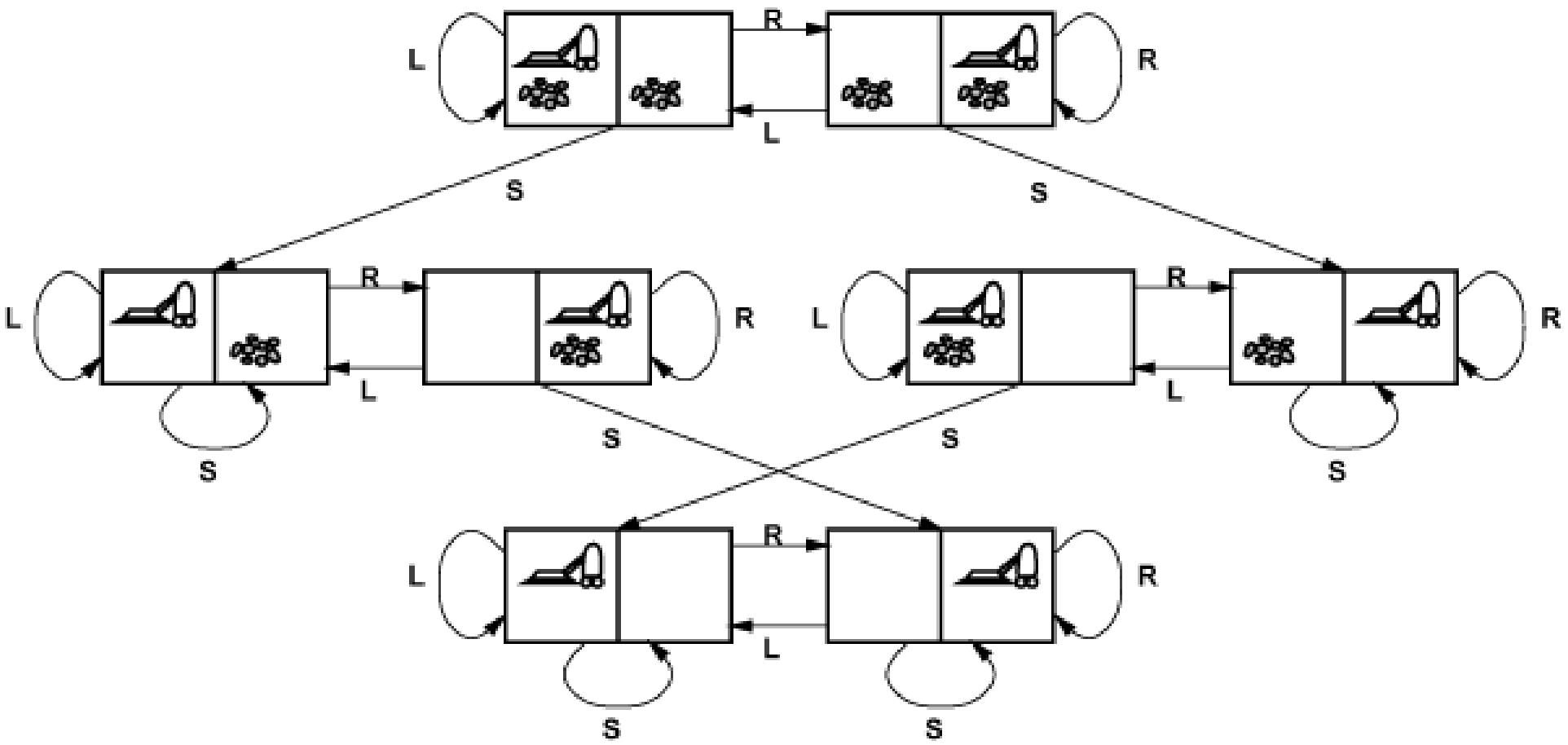
    estado inicial = 5

    seqüência de ações = [direita, aspirar]

    leva ao estado 8 (final)

- Técnica a aplicar: **Busca**

Espaço de estado do agente aspirador

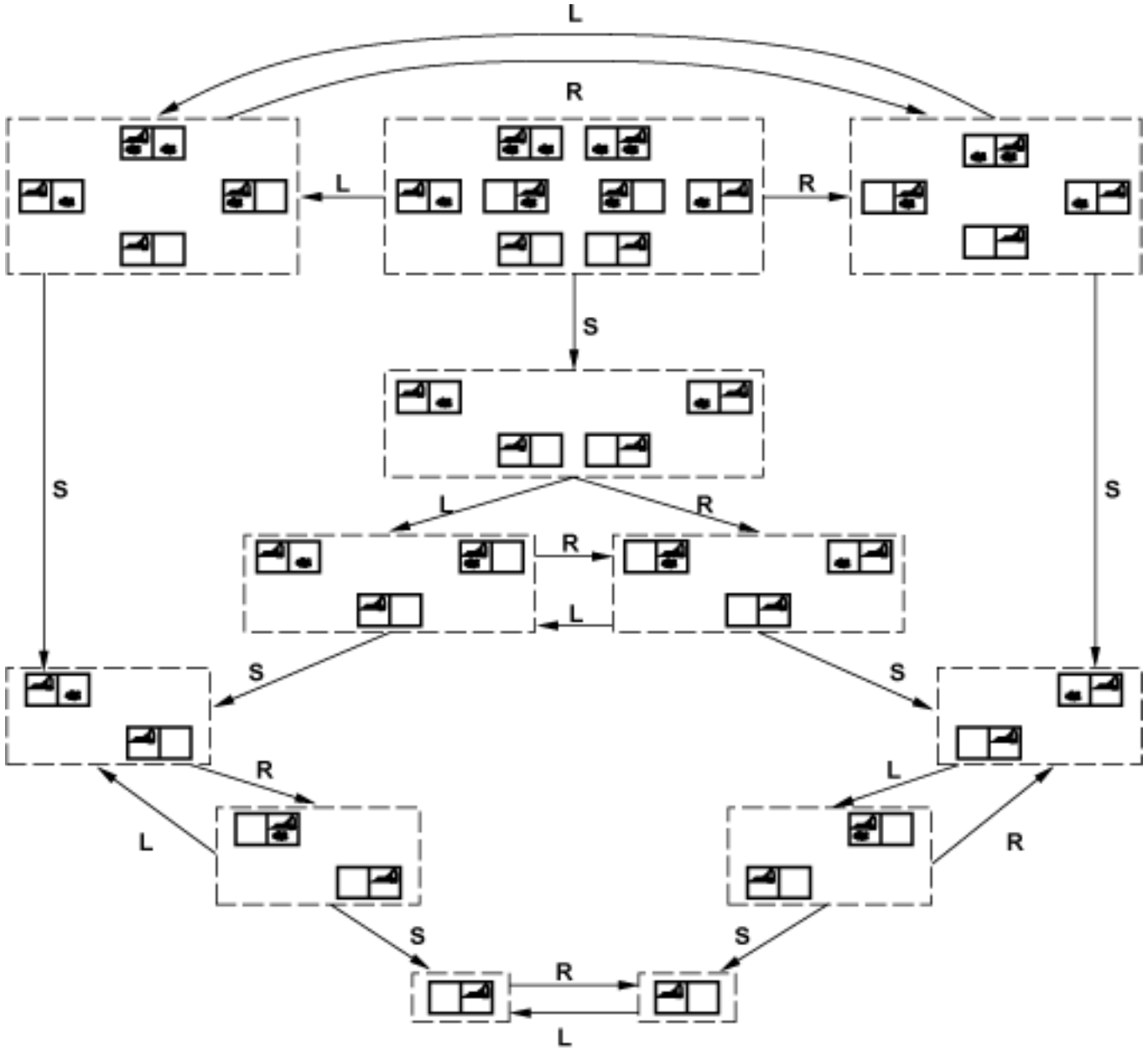


Tipo 2: Problemas de múltiplos estados

- Conhecimento do agente
  - não sabe seu estado inicial (percepção deficiente), mas sabe o efeito de suas ações, OU
  - não sabe o efeito das ações (execução deficiente), mas sabe seu estado inicial

    → lei de Murphy: aspira poeira existente e também pode jogar poeira quando o quarto já estava limpo
- **O agente deve raciocinar sobre os** conjuntos de estados **aos quais ele pode chegar pelas ações.**
- **Nestes casos,** *sempre* existe **uma seqüência de ações que leva a um estado final.**
- Técnica a aplicar: **Busca**

Exemplo - problema tipo 2



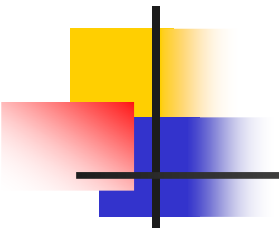
Tipo 3: Problemas contingências

- Conhecimento do agente:
  - o agente não enxerga o ambiente inteiro OU
  - o agente não sabe precisar o efeito das ações
    - exs. agente enxerga apenas o quarto onde está, dirigir com mapa
- **Não há** seqüência prévia de ações **que garanta a solução do problema**
- O **agente precisa** intercalar busca e execução
  - ex. o agente só pode decidir aspirar quando chegar ao quarto
  - {1,5} -> [aspirar, direita, aspirar se existe poeira]
- Técnica a aplicar: **Planejamento**
  - O agente constrói uma **árvore de ações,** onde cada ramo lida com uma possível **contingência.**

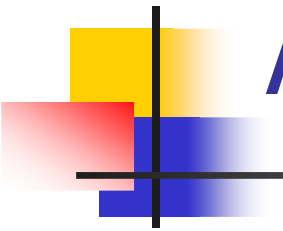
Tipo 4: Problemas exploratórios

- Conhecimento do agente:
  - o agente não conhece seus possíveis estados E
  - o agente não sabe o efeito de suas ações
    - ex. estar perdido em uma cidade desconhecida sem mapa.
- **O agente deve** explorar seu ambiente**, descobrindo gradualmente o resultado de suas ações e os estados existentes.**
  - Se o agente "sobreviver", terá aprendido um mapa do ambiente, que poderá ser reutilizado em problemas subseqüentes.
- Técnica a aplicar: **Aprendizagem**

Aplicações de Busca: "Toy Problems"



- Jogo das n rainhas
- Jogo dos n números (*n-puzzle*)
- Palavras cruzadas



Aplicações: Problemas Reais

- Cálculo de rotas
  - rotas em redes de computadores
  - sistemas de planejamento de viagens
  - planejamento de rotas de aviões
  - Caixeiro viajante
- Alocação (Scheduling)
  - Salas de aula
  - Máquinas industriais (job shop)
- Projeto de VLSI
  - Cell layout
  - Channel routing

Aplicações: Problemas Reais

- Navegação de robôs:
  - generalização do problema da navegação
  - robôs movem-se em espaços contínuos, com um conjunto (infinito) de possíveis ações e estados
    - controlar os movimentos do robô no chão, e de seus braços e pernas requer espaço multi-dimensional
- Montagem de objetos complexos por robôs:
  - ordenar a montagem das diversas partes do objeto
- etc...

Busca em Espaço de Estados

- Uma vez o problema bem formulado... o estado final deve ser "buscado"
- **Em outras palavras, deve-se usar um** método de busca **para saber a** ordem correta de aplicação dos operadores **que lavará do estado inicial ao final**
- **Uma vez a busca terminada com sucesso, é só** executar a solução **(= conjunto ordenado de operadores a aplicar)**

Busca em Espaço de Estados: Geração e Teste

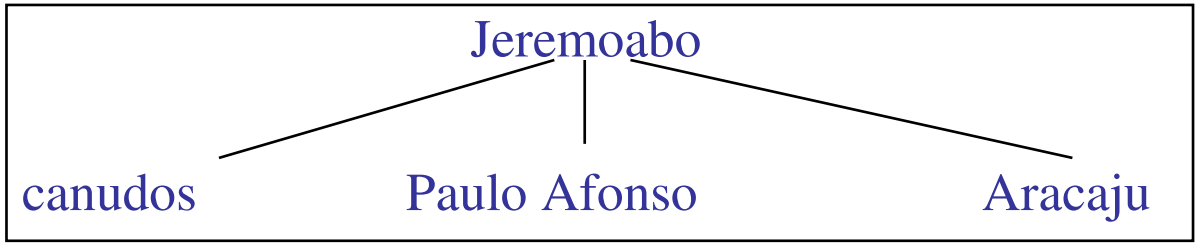
- *Fronteira* do espaço de estados
  - nós (estados) a serem expandidos no momento.
- Algoritmo:

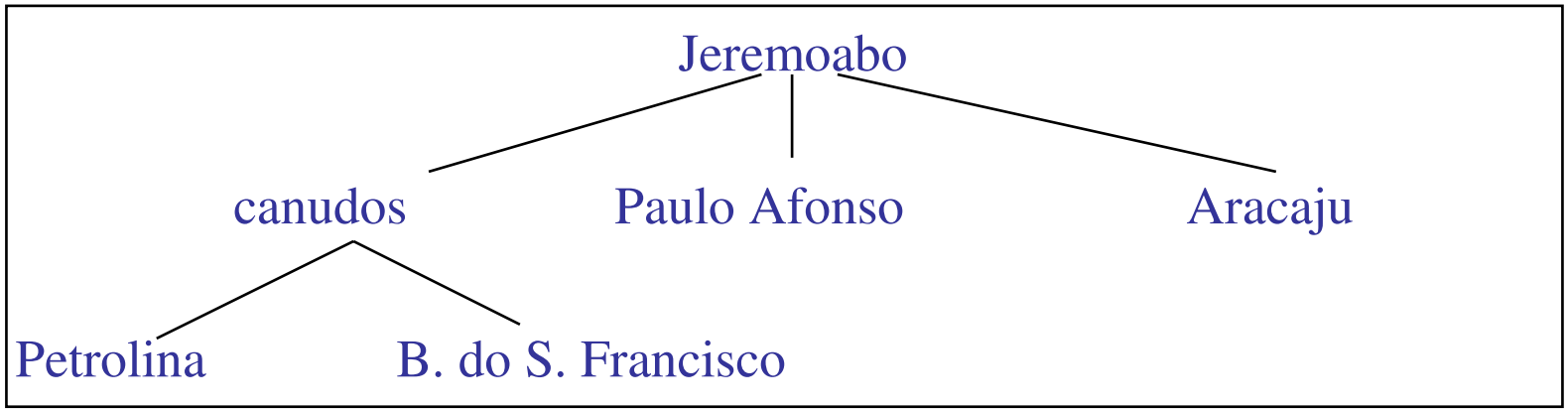
Obs: o algoritmo começa com a fronteira contendo o estado inicial do problema.

1. Selecionar o primeiro nó (estado) da *fronteira* do espaço de estados;
   - se a fronteira está vazia, o algoritmo termina com falha.
2. Testar se o nó é um estado final (solução):
   - se "sim, então retornar nó - a busca termina com sucesso.
3. Gerar um novo conjunto de estados pela aplicação dos operadores ao estado selecionado;
4. Inserir os nós gerados na *fronteira*, de acordo com a estratégia de busca usada, e voltar para o passo (1).

Exemplo: viajar de Jeremoabo a Cajazeiras

estado inicial => Jeremoabo





Busca em Espaço de Estados: Implementação

- Espaços de Estados
  - podem ser representados como uma árvore onde os estados são nós e as operações são arcos.
- Os nós da árvore podem guardar mais informação do que apenas o estado:

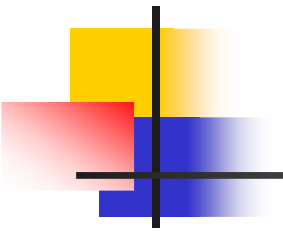
  → são uma estrutura de dados com 5 componentes:

  1. o estado correspondente
  2. o seu nó pai
  3. o operador aplicado para gerar o nó (a partir do pai)
  4. a profundidade do nó
  5. o custo do nó (desde a raiz)

Métodos de Busca

- Busca exaustiva - cega
  - Não sabe qual o **melhor** nó da fronteira a ser expandido = menor custo de caminho desse nó até um ***nó final*** *(objetivo)*.
  - Estratégias de Busca (ordem de expansão dos nós):
    - caminhamento em largura
    - caminhamento em profundidade
  - Direção de Busca:
    - Do estado inicial para o objetivo
    - Do objetivo para o estado inicial
- Busca heurística - informada
  - Estima qual o melhor nó da fronteira a ser expandido com base em ***funções heurísticas*** => conhecimento
  - Estratégia de busca: best-first search (melhor escolha)
  - Direção de Busca: idem à busca cega

Critérios de Avaliação das Estratégias de Busca



- Completude:
  - a estratégia sempre encontra uma solução quando existe alguma?
- Custo do tempo:
  - quanto tempo gasta para encontrar uma solução?
- Custo de memória:
  - quanta memória é necessária para realizar a busca?
- Otimização/qualidade (*optimality*):
  - a estratégia encontra a melhor solução quando existem diferentes soluções?