

O que é IA? Sistemas que pensam como humanos. Sistemas que pensam racionalmente. Sistemas que agem como humanos. Sistemas que agem racionalmente.

Agentes e Ambientes

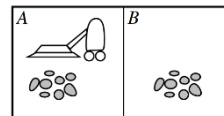
Agentes incluem humanos, robots, softbots, termóstatos, etc.

A função agente mapeia histórias de percepções em ações. O programa agente corre num computador para produzir a função.

Por exemplo: Mundo do aspirador.

Percepções: local e conteúdo, exemplo [A,Sujo]

Ações: Esquerda, Direita, Aspira, nada



Um **agente racional** escolhe qual a ação que maximiza o valor esperado da medida do desempenho dadas sequências de percepções.

PEAS (Medida de desempenho, Ambiente, Atuadores, Sensores) - Para desenhar um agente racional deve-se especificar o ambiente da tarefa a desempenhar.

Tipos de Agentes - Quatro tipos básicos de agentes:

Agente de reflexos simples (simple reflex agents);

Agente de reflexos com estados (reflex agents with state);

Agente orientado por objetivos (goal-based agents);

Agente orientado por utilidade (utility-based agents).

Problem-solving agentes (Tipos de problemas)

Determinísticos, observáveis → *Single-state problema* - O agente sabe exatamente em que cidade está, a solução é uma sequência.

Não-observável → *Conformant problem* - O agente pode não ter ideia de onde está; a solução, se existir, é uma sequência.

Não determinísticos e/ou parcialmente observável → *problema de contingência* - As percepções dão nova informação sobre o estado corrente. A solução é um plano de contingência ou uma política. Muitas vezes intercala procura, execução.

Espaços de estado desconhecido → *problema de exploração ("online")*

Formulação de problema de estado único - Um problema é definido com:

Estado inicial, Função sucessor (conjunto de ações-pares de estados), Estado final, Custo do caminho.

Uma solução é uma sequência de ações que levam o agente do estado inicial ao estado final.

Implementação: Estados vs. Nós

Um **estado** é a representação de uma configuração física.

Um **nó** é uma estrutura de dados que faz parte da árvore de pesquisa.

Estratégias de pesquisa

As estratégias de pesquisa são avaliadas nas seguintes dimensões.

Completude - Encontra sempre uma solução se existir?

Complexidade temporal - Número de nós gerados/expandidos.

Complexidade espacial - Número máximo de nós em memória.

Optimalidade - Encontra sempre a solução de menor custo?

Estratégias não informadas só usam a informação disponível na definição do problema:

Pesquisa em largura (Breadth-first search) - Expandir o nó menos profundo que ainda não foi expandido.

Pesquisa de custo uniforme (Uniform-cost search) - Expande o nó de menor custo que ainda não foi expandido.

Pesquisa em profundidade (Depth-first search) - Expande o nó mais profundo que ainda não foi expandido.

Pesquisa em profundidade limitada (Depth-limited search) - Com profundidade limitada.

Pesquisa em profundidade iterativa (Iterative deepening search) - Usado um loop para percorrer a árvore a cada profundidade.

| Criterion | Breadth-First | Uniform-Cost | Depth-First | Depth-Limited | Iterative Deepening |
|-----------|---------------|----------------------------------|-------------|--------------------|---------------------|
| Complete? | Yes* | Yes* | No | Yes, if $l \geq d$ | Yes |
| Time | b^{d+1} | $b^{\lceil C^*/\epsilon \rceil}$ | b^m | b^l | b^d |
| Space | b^{d+1} | $b^{\lceil C^*/\epsilon \rceil}$ | bm | bl | bd |
| Optimal? | Yes* | Yes | No | No | Yes* |

Pesquisa o melhor primeiro (Best-first search)

Ideia: Usar uma função de avaliação para cada nó - estimando a "adequação". Expandir o nó mais adequado.

Casos especiais:

Pesquisa ansiosa ou gananciosa (Greedy search)

Função de avaliação $h(n)$ (heurística) estimativa do custo de n ao objetivo mais próximo. A pesquisa ansiosa expande o nó que aparenta estar mais próximo do objetivo.

Pesquisa A (A* search)*

Ideia: Evitar expandir os caminhos caros.

Função de avaliação $f(n) = g(n) + h(n)$

$g(n)$ - Custo até atingir n

$h(n)$ - Custo estimado para atingir o objetivo a partir de n

$f(n)$ - Custo total estimado do caminho que passa por n até ao objetivo

A pesquisa A* usa uma heurística admissível, ou seja, o custo de $h(n)$ tem de ser inferior ou igual ao verdadeiro custo de n .

Teorema: A pesquisa A* é ótima.

Constraint satisfaction problems (CSPs)

Em problemas de pesquisa standard o estado é uma "caixa preta" - qualquer estrutura de dados.

Em problemas de CSP o estado é definido por variáveis X_i com valores de domínio D_i . O objetivo é um conjunto de restrições que especificam as combinações permitidas dos valores dos subconjuntos das variáveis.

Exemplo: Coloração de mapas.

Variáveis: WA, NT, Q, NSW, V, SA, T

Domínios: $D_i = \{\text{red, green, blue}\}$

Restrições: regiões adjacentes devem ter cores diferentes.

A pesquisa em profundidade para CSPs com afetação de uma única variável chama-se

pesquisa backtracking. Pesquisa Backtracking é um algoritmo de pesquisa não informada para CSPs.

Melhorar a eficiência da pesquisa:

- Minimum remaining values (Menos valores no domínio): Escolhe as variáveis com menos valores possíveis.

- Variáveis com mais restrições: Escolher a variável com mais restrições com as variáveis por instanciar.

- Least constraining value (O valor menos restringido): Dada uma variável é escolhido o valor menos restringido, ou seja, o que retira menos valores aos domínios das restantes variáveis.

Forward checking: Retira valores que já não sejam possíveis das outras variáveis.

Termina a pesquisa quando uma variável já não tem valores possíveis. Não prevê erros.