

<p>Inteligência Artificial</p> <p>Luís A. Alexandre</p> <p>UBI</p> <p>Ano lectivo 2019-20</p>		<p>Resolução de problemas usando pesquisa</p>	<p>1 / 42</p>
<p>Luís A. Alexandre (UBI)</p>	<p>Inteligência Artificial</p>	<p>Ano lectivo 2019-20</p>	<p>1 / 42</p>

<p>Conteúdo</p>	<p>Resolução de problemas</p> <p>Resolução de problemas usando pesquisa</p> <p>Problema</p> <p>Exemplos de problemas</p> <p>Tipos de problemas</p> <p>Problemas artificiais</p> <p>Problemas reais</p> <p>Pesquisa de soluções</p> <p>Estratégias de pesquisa não informada</p> <p>Pesquisa primeiro em largura:</p> <p>PPL</p> <p>Pesquisa primeiro em profundidade: PPP</p>	<p>Pesquisa com profundidade limitada: PPP-PL</p> <p>PPP com profundidade iterativa: PPP-PI</p> <p>Pesquisa bidirecional: PB</p> <p>Pesquisas: resumo</p> <p>Estratégias de pesquisa informada</p> <p>Estratégias de pesquisa informada</p> <p>Pesquisa melhor primeiro: PMP</p> <p>Pesquisa A*</p> <p>Funções heurísticas</p> <p>Leitura recomendada</p>	<p>Inteligência Artificial</p> <p>Ano lectivo 2019-20</p> <p>2 / 42</p>
-----------------	---	---	---

<p>Resolução de problemas usando pesquisa</p>		<p>3 / 42</p>
<p>Luís A. Alexandre (UBI)</p>	<p>Inteligência Artificial</p>	<p>Ano lectivo 2019-20</p>

<p>Resolução de problemas</p> <p>Problema</p>	<p>Definição dum problema</p> <p>Um problema pode ser definido formalmente usando quatro componentes:</p> <ol style="list-style-type: none"> 1. O estado inicial. 2. Uma descrição das ações possíveis. Normalmente usa-se a função sucessor $suc()$, que recebe um estado x e devolve pares do tipo $(ação, estado)$ em que mostra o estado que se atinge partindo de x e adoptando a respectiva ação. 3. O teste de objetivo que permite avaliar se um dado estado é o estado objetivo. 4. Uma função de custo do caminho que permite atribuir um valor numérico a cada caminho. <p>O estado inicial e a função sucessor definem implicitamente o espaço de estados do problema, que é o conjunto de todos os estados que se podem alcançar partindo do estado inicial, usando qualquer sequência de ações.</p> <p>O espaço de estados forma um grafo dirigido em que os nodos são estados e as arestas são ações.</p>	<p>Inteligência Artificial</p> <p>Ano lectivo 2019-20</p> <p>4 / 42</p>
---	--	---

<p>Definição dum problema</p>		<p>5 / 42</p>
<p>Luís A. Alexandre (UBI)</p>	<p>Inteligência Artificial</p>	<p>Ano lectivo 2019-20</p>

<p>Resolução de problemas</p> <p>Problema</p>	<p>Simplificações</p> <p>Ao definirmos o problema da IA como acabámos de fazer estamos a contar com uma série de simplificações:</p> <ol style="list-style-type: none"> 1. O ambiente é estático: durante a execução da solução não existe qualquer alteração ao ambiente. 2. O ambiente é observável: permite o conhecimento dos estados através dos sensores. 3. O ambiente é discreto: permite a enumeração de vários estados. 4. O ambiente é determinístico: as soluções são sequências de ações únicas e não existem acontecimentos inesperados. 	<p>Inteligência Artificial</p> <p>Ano lectivo 2019-20</p> <p>6 / 42</p>
---	--	---

Conteúdo

Resolução de problemas
Resolução de problemas usando pesquisa
Problema

Pesquisa com profundidade limitada: PPP-PL
PPP com profundidade iterativa: PPP-PI

Exemplos de problemas

Tipos de problemas
Problemas artificiais
Problemas reais

Estratégias de pesquisa informada

Estratégias de pesquisa

Pesquisa de soluções

Estratégias de pesquisa não informada

Pesquisa melhor primeiro: PMP

Pesquisa A*

Pesquisa primeiro em largura:

PPL

Pesquisa primeiro em

profundidade: PPP

Funções heurísticas

Leitura recomendada

Tipos de problemas

- ▶ Podemos dividir os problemas em dois tipos: os reais e os artificiais (toy).
- ▶ Os artificiais têm uma resolução simplificada dado que o seu contexto é normalmente mais abstracto que o dos problemas reais.
- ▶ Vamos ver agora alguns exemplos de cada um destes tipos de problema e como se faz a sua formulação em termos de IA.

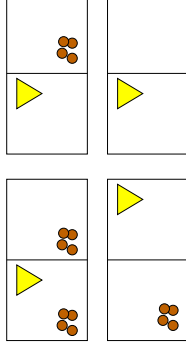
Problemas artificiais

- ▶ Vamos ver de seguida os seguintes problemas:

1. mundo do aspirador
2. puzzle de 8 peças
3. problemas das 8 rainhas

Mundo do aspirador

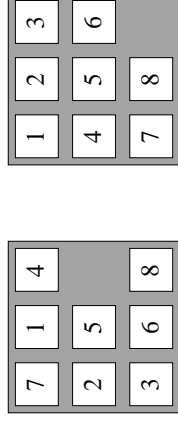
- ▶ Neste problema consideramos um mundo simples apenas com 2 posições (ou salas) que podem ou não ter pó e um agente que se pode deslocar de uma para a outra posição e aspirar o pó.
- ▶ Quatro exemplos de estados estão nas figuras abaixo onde o agente é representado pelo triângulo e o pó pelo grupo de pontos.



Mundo do aspirador

- ▶ Estados: cada uma das 2 salas pode ter ou não pó (4 possibilidades) e o agente pode estar em qualquer destas salas (2 possibilidades), logo temos um total de 8 estados possíveis.
- ▶ Estado inicial: qualquer estado pode ser escolhido como o inicial;
- ▶ Função sucessor: gera uma das seguintes ações: esquerda, direita ou aspirar;
- ▶ Teste do objetivo: verifica se todos as salas estão limpas;
- ▶ Custo do caminho: cada ação custa 1 logo o custo do caminho é o número de ações executadas.

Puzzle de 8 peças



- ▶ Neste problema temos um tabuleiro com 3x3 posições, oito peças numeradas e um espaço vazio.
- ▶ As peças adjacentes ao espaço vazio podem deslocar-se para ele (também pode ser visto da forma complementar: o espaço movimenta-se para um local ocupado por uma peça).
- ▶ O objetivo é atingir um estado final, por exemplo, ordenar as peças como no exemplo da figura da direita.

Puzzle de 8 peças

- ▶ Estados: um estado tem de especificar a localização das 8 peças (a posição vazia fica especificada implicitamente).
- ▶ Estado inicial: qualquer estado pode ser o estado inicial, por exemplo, o da figura anterior do lado esquerdo.
- ▶ Função sucessor: especifica um movimento do espaço vazio para uma das casas adjacentes
- ▶ Teste do objetivo: verificar se foi atingido o estado desejado
- ▶ Custo do caminho: cada movimento numa peça custa 1 logo o custo do caminho é simplesmente o número de movimentos efectuados.

Luís A. Alexandre (UBI)

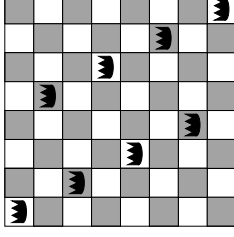
Inteligência Artificial

Ano lectivo 2019-20

13 / 42

Problemas das 8 rainhas

- ▶ O objetivo deste problema é colocar 8 rainhas num tabuleiro de xadrez sem que nenhuma seja atacada.
- ▶ Estados: qualquer arranjo de 0 a 8 rainhas no tabuleiro é um estado
- ▶ Estado inicial: nenhuma rainha no tabuleiro
- ▶ Função sucessor: adicionar uma rainha a uma posição vazia do tabuleiro
- ▶ Teste do objetivo: 8 rainhas no tabuleiro e nenhuma sob ataque



Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

14 / 42

Problemas reais

- ▶ Vamos ver de seguida os seguintes problemas:
 1. encontrar rotas
 2. problemas de visitas
 3. problema do caixeiro viajante (TSP: traveling salesperson problem)

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

15 / 42

Encontrar rotas

- ▶ O problema de **encontrar rotas** é muito frequente: encaminhamento de pacotes em redes de computadores, sistemas de planeamento de rotas de companhias aéreas, planeamento de operações militares, entre outros.
- ▶ Vejamos um exemplo simplificado do problema da especificação de rotas dum site de pesquisa de viagens aéreas:
 - ▶ Estados: cada estado representa uma localização (p.ex. um aeroporto) e a hora a que se está nessa localização;
 - ▶ Estado inicial: hora inicial e posição inicial;
 - ▶ Função sucessor: devolve os estados resultantes da aplicação de qualquer voo possível, a partir do aeroporto actual, após a hora actual;
 - ▶ Teste do objetivo: estamos no local desejado antes dum dada hora pré-definida?
 - ▶ Custo do caminho: depende do preço das viagens, do tempo de espera, do tempo de voo, tipo de avião, etc.

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

16 / 42

Problemas de visitas e TSP

- ▶ Estes problemas são parecidos com os de rotas: cada estado em vez de conter apenas uma localização contém a lista de todas as localizações visitadas até ao momento.
- ▶ A solução para estes problemas consiste na sequência de ações para que um dado grupo de locais seja visitado.
- ▶ O TSP é também um problema de visitas, mas com restrições: cada local apenas pode ser visitado uma vez e queremos a rota mais curta.
- ▶ O problema TSP é também usado para planear o movimento de perfuradores de circuitos impressos e máquinas que arrumam caixas em armazéns.

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

17 / 42

Conteúdo

Resolução de problemas	Pesquisa com profundidade limitada: PPP-PL
Resolução de problemas usando pesquisa	PPP com profundidade iterativa: PPP-PI
Problema	Pesquisa bidirecional: PB
Exemplos de problemas	Pesquisas: resumo
Tipos de problemas	Estratégias de pesquisa informada
Problemas artificiais	Estratégias de pesquisa informada
Problemas reais	Pesquisa melhor primeiro: PMP
Pesquisa de soluções	Pesquisa A*
Estratégias de pesquisa não informada	Funções heurísticas
Pesquisa primeiro em largura: PPL	Leitura recomendada
Pesquisa primeiro em profundidade: PPP	

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

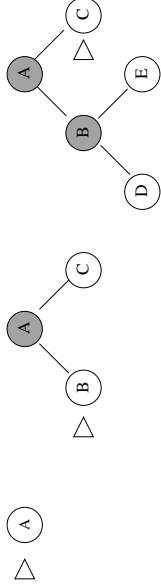
18 / 42

Estratégias de pesquisa não informada

- ▶ As estratégias que iremos discutir nesta secção usam apenas a informação que é fornecida na definição do problema.
- ▶ O que estas estratégias fazem é gerar estados e verificar se são o estado objetivo.
- ▶ Todas as estratégias de pesquisa podem distinguir-se pela ordem de expansão dos nodos no grafo de estados.

Pesquisa primeiro em largura: PPL

- ▶ Exemplo numa árvore binária: o símbolo \triangleright indica o nodo a expandir, os nodos a escuro já foram expandidos e os restantes foram gerados mas não expandidos.



Pesquisa com profundidade limitada: PPP-PL

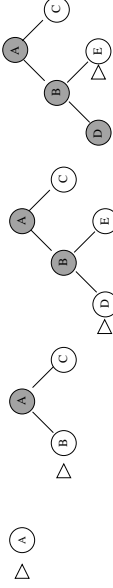
- ▶ Este caso a pesquisa é feita como na PPP com a diferença que só se atinge uma dada profundidade pré-definida na árvore de pesquisa.
- ▶ É uma boa solução quando temos árvores de pesquisa não equilibradas e podemos correr o risco de explorar profundidades muito grandes quando a solução se encontra a profundidades bastantes inferiores.

Pesquisa primeiro em largura: PPL

- ▶ Nesta pesquisa primeiro expande-se o nodo raiz, depois todos os seus filhos, de seguida todos os netos e assim sucessivamente.
- ▶ Podemos resumir o processo dizendo que se expandem todos os nodos dum dado nível antes de prosseguir a expansão aos nodos do nível seguinte.
- ▶ Esta pesquisa é **completa** se o factor de ramificação for finito.
- ▶ Esta pesquisa é **ótima** quando o custo do caminho é uma função não decrescente da profundidade do nodo: por exemplo, todas as ações têm o mesmo custo.

Pesquisa primeiro em profundidade: PPP

- ▶ Neste caso a expansão faz-se sempre ao nodo que se encontre a maior profundidade (quando estiverem todos a igual profundidade, faz-se a um qualquer).
- ▶ Quando se atinge um nodo folha, expande-se a partir do nodo gerado mais recentemente e que ainda não foi expandido.



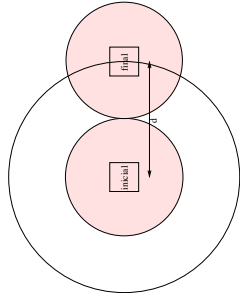
(D é nodo folha)

PPP com profundidade iterativa: PPP-PI

- ▶ Aqui temos outra variante da PPP, em que a profundidade máxima de pesquisa é variável.
- ▶ A variação da profundidade é feita gradualmente, começando por ser feita uma PPP até à profundidade 1; se não for encontrada a solução é feito nova PPP até à profundidade 2; e assim sucessivamente até se atingir o objetivo.
- ▶ Esta abordagem combina os benefícios da PPL com os da PPP:
 - ▶ tem requisitos de memória reduzidos ($O(bd)$) (como PPP)
 - ▶ é **completa** quando o factor de ramificação é finito (como PPL)
 - ▶ é **ótima** quando o custo do caminho é uma função não decrescente da profundidade do nodo: por exemplo, todas as ações têm o mesmo custo (como PPL)
- ▶ Em geral esta é a forma de pesquisa não informada mais indicada quando o espaço de pesquisa é grande e a profundidade da solução é desconhecida.

Pesquisa bidirecional: PB

- ▶ Neste caso são feitas duas pesquisas: uma com início no estado inicial e outra a começar pelo estado objetivo.
- ▶ A vantagem desta abordagem é óbvia quando olhamos para a figura abaixo: a área total de pesquisa quando consideramos os dois círculos correspondentes à pesquisa bidirecional é inferior à área quando consideramos apenas a pesquisa a partir do nodo inicial.



Pesquisa bidirecional

- ▶ Para efectuarmos a pesquisa bidirecional necessitamos de uma forma de pesquisar para trás: uma função predecessor $Pred(x)$ que nos devolve o estado anterior ao estado x .
- ▶ O problema é que nem sempre é fácil obter esta função.
- ▶ O algoritmo é **completo** se o factor de ramificação for finito e ambas as pesquisas forem em largura.
- ▶ O algoritmo **óptimo** apenas se os passos tiverem o mesmo custo e ambas as pesquisas forem em largura.

Pesquisas: resumo

Pesquisa	Óptima	Completa	Complex. espacial	Complex. temporal
PPL	S^1	S^1	$O(b^d)$	$O(b^d)$
PPP	N	N	$O(bm)$	$O(b^m)$
PPP-PL	N	N	$O(bp)$	$O(b^p)$
PPP-PI	S^1	S^1	$O(bd)$	$O(b^d)$
PB	S^1	S^1	$O(b^{d/2})$	$O(b^{d/2})$

b =factor de ramificação; m =profundidade da árvore de pesquisa;
 d =profundidade da solução menos profunda; p =limite profundidade; 1:
ver as condições no acetato respectivo;

Conteúdo

Resolução de problemas	Pesquisa com profundidade limitada: PPP-PL
Resolução de problemas usando pesquisa	PPP com profundidade iterativa: PPP-PI
Problema	Pesquisa bidirecional: PB
Exemplos de problemas	Pesquisas: resumo
Tipos de problemas	
Problemas artificiais	Estratégias de pesquisa informada
Problemas reais	Estratégias de pesquisa informada
Pesquisa de soluções	Pesquisa melhor primeiro: PMP
Estratégias de pesquisa não informada	Pesquisa A^*
Pesquisa primeiro em largura: PPL	Funções heurísticas
Pesquisa primeiro em profundidade: PPP	Leitura recomendada

Estratégias de pesquisa informada

- ▶ Já vimos que podemos usar pesquisa gerando sistematicamente novos estados e verificando se os mesmos correspondem ao objetivo.
- ▶ O problema desta abordagem é que é muitas vezes **pouco eficiente**.
- ▶ Iremos estudar agora estratégias de pesquisa que usam conhecimento específico do problema para permitir a descoberta de soluções de forma mais eficiente.
- ▶ As estratégias de **pesquisa informada** conseguem saber se um estado é melhor que outro.

Pesquisa melhor primeiro: PMP

- ▶ Nesta pesquisa o nodo a expandir é escolhido em função do valor dado por uma **função de avaliação** $f(n)$ que dá uma **estimativa do custo** associado ao nodo.
- ▶ Deste modo, é expandido primeiro o nodo com menor valor dado pela função de avaliação.
- ▶ Existem vários algoritmos de PMP dependendo da **função heurística** $h(n)$ que usam.
- ▶ A **função heurística**, $h(n)$, é uma componente da função de avaliação que **estima** o custo do caminho mais barato do nodo n até ao objetivo.
- ▶ Estas funções têm a seguinte restrição: $h(n) = 0$ se n for um nodo objetivo.

Pesquisa melhor primeiro gulosa (greedy): PMPG

- ▶ Nesta pesquisa o nodo a expandir é o que estiver mais próximo do nodo objetivo.
- ▶ Isto quer dizer que a função de avaliação consiste apenas na função heurística: $f(n) = h(n)$.
- ▶ Esta pesquisa sofre dos mesmos problemas da PPP: não é ótima nem é completa (pode seguir um caminho infinito e nunca voltar para trás para experimentar outras hipóteses).
- ▶ A complexidade espacial e temporal é $O(b^m)$, onde m é a profundidade máxima do espaço de pesquisa.

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

37 / 42

Funções heurísticas

Conteúdo

- Resolução de problemas
 - Resolução de problemas usando pesquisa
 - Problema
- Exemplos de problemas
 - Tipos de problemas
 - Problemas artificiais
 - Problemas reais
- Pesquisa de soluções
 - Estratégias de pesquisa não informada
 - Pesquisa primeiro em largura: PPL
 - Pesquisa primeiro em profundidade: ppp
- Pesquisa com profundidade limitada: PPP-PL
- PPP com profundidade iterativa: PPP-PI
- Pesquisa bidirecional: PB
- Pesquisas: resumo
- Estratégias de pesquisa informada
 - Estratégias de pesquisa informada
 - Pesquisa melhor primeiro: PMP
 - Pesquisa A*
- Funções heurísticas**
- Leitura recomendada

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

39 / 42

Funções heurísticas

Funções heurísticas

- ▶ Duas possíveis heurísticas:
 - ▶ h_1 : número de peças fora do sítio em relação à solução. No caso da figura, $h_1 = \underline{\hspace{1cm}}$.
 - ▶ h_2 : soma das distâncias das peças em relação às suas posições finais. Usa-se a distância city block ou Manhattan: não há movimento diagonal. Para o caso da figura temos $h_2 = \underline{\hspace{1cm}}$.

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

41 / 42

Pesquisa A*

- ▶ Esta pesquisa (chamada A estrela) é a mais conhecida das PMP.
- ▶ A função de avaliação, $f(n)$, além da função heurística, $h(n)$, usa também o **custo do caminho** para chegar até ao nodo vindo do estado inicial, $g(n)$: $f(n) = g(n) + h(n)$.
- ▶ Se $h(n)$ satisfizer algumas condições, esta pesquisa é completa e ótima.
- ▶ Se $h(n)$ for uma **heurística admissível** - o que significa que $h(n)$ nunca sobrestima o custo de atingir o objetivo - A* é ótima (para pesquisa em árvores).
- ▶ Para um problema de achar um caminho num mapa, um exemplo dum heurística admissível seria o comprimento da distância em linha recta entre a posição actual e a objetivo: qualquer que seja o verdadeiro caminho da posição actual para a objetivo, será sempre maior que a distância em linha recta.

Luís A. Alexandre (UBI)

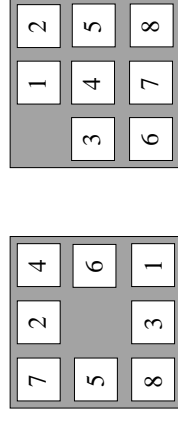
Inteligência Artificial

Ano lectivo 2019-20

38 / 42

Funções heurísticas

Funções heurísticas



Estado inicial

objetivo

- ▶ Vamos usar para exemplo o puzzle de 8 peças.
- ▶ A solução atinge-se em média com custo 22.
- ▶ O factor de ramificação é aproximadamente 3. Porquê?
- ▶ Assim, uma pesquisa exaustiva teria de procurar em $3^{22} \approx 3.1 \times 10^{10}$ estados.
- ▶ Se evitarmos estados repetidos, reduzimos o espaço de pesquisa para aproximadamente 2×10^5 estados.

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

40 / 42

Leitura recomendada

Leitura recomendada

- ▶ Russell e Norvig, cap. 3.

Luís A. Alexandre (UBI)

Inteligência Artificial

Ano lectivo 2019-20

42 / 42