Inteligência Artificial

Funções Heurísticas

1

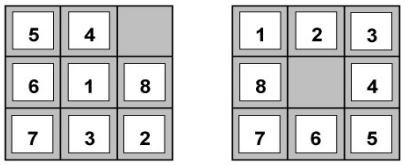
Inventando Funções Heurísticas

- **Como escolher uma boa função heurística *h*?**
- ***h* depende de cada problema particular.**
- ***h* deve ser *admissível***
  - não superestimar o custo real da solução
- **Exemplo: jogo dos 8 números**
  - um número pode mover-se de A para B se A é adjacente a B e B está vazio
  - busca exaustiva:
    - *solução média em 22 passos*
    - *fator de ramificação médio: 3*
    - *Assim, ≈ 3^{22} estados possíveis*

4	5	8
	1	6
7	2	3

2

Heurísticas para jogo 8 números



Algumas heurísticas possíveis:

h1 = no. de elementos fora do lugar (*h1*=7)

h2 = soma das distâncias de cada número à posição final (*h2*=2+3+3+2+4+2+0+2=18)

Exercício: resolver com A* e h1 e h2 e comparar soluções (10 ações).

3

Qualidade da função heurística

- **Qualidade da função heurística:** medida através do fator de expansão efetivo (*b**).
  - *b** é o fator de expansão de uma árvore uniforme com *N+1* nós e nível de profundidade *d*
  - $N+1 = 1 + b^* + (b^*)^2 + ... + (b^*)^d$, onde
    N = total de nós gerados pelo A para um problema*
    d = profundidade da solução;
- **Mede-se empiricamente a qualidade de *h* a partir do conjunto de valores experimentais de *N* e *d*.**
  - uma boa função heurística terá o *b** muito próximo de 1.
- **Se o custo de execução da função heurística for maior do que expandir nós, então ela *não* deve ser usada.**
  - uma boa função heurística deve ser *eficiente* e *econômica*.

4

Experimento (Média de 100 soluções/dado)

	Número médio de nós expandidos (100 épocas)			Effective Branching Factor b*		
d	IDS	A*(h_1)	A*(h_2)	IDS	A*(h_1)	A*(h_2)
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	364404	227	73	2.78	1.42	1.24
14	3473941	539	113	2.83	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

IDS: *Iterative-Deepening-Search* (Busca de Aprofundamento Iterativo)

Uma boa função heurística terá o *b muito próximo de 1.**

Note que h2 é melhor que h1 ➔ de fato, para qualquer nó, h2(n)≥h1(n)!

Escolhendo Funções Heurísticas

- **É sempre melhor usar uma função heurística com valores mais altos, contanto que ela seja *admissível* e que o tempo para computá-la não seja muito grande!**
  - ex. *h2* melhor que *h1*.
- **h_i domina $h_k \Rightarrow h_i(n) \geq h_k(n)$ $\forall n$ no espaço de estados**
  - h_2 domina h_1 no exemplo anterior
- **Caso existam muitas funções heurísticas para o mesmo problema, e nenhuma delas domine as outras, usa-se uma *heurística composta*:**
  - $h(n) = max(h_1(n), h_2(n),...,h_m(n))$
  - Assim definida, *h* é *admissível* e *domina* cada função h_i individualmente

6

Como inventar funções heurísticas admissíveis?

- **Existem estratégias genéricas para definir *h*:**

1) Relaxar o problema (versão simplificada);
2) Usar informação estatística;
3) Identificar os atributos relevantes do problema e usar aprendizagem.

7

(1) Relaxando o problema

- **Problema Relaxado:**
  - versão simplificada do problema original, onde os operadores são menos restritivos
- **Operadores relaxados:**
  1. Uma peça pode se mover para qualquer lugar
     - *h1 seria o custo da solução "correta" neste jogo*
  2. Uma peça pode se mover para lugares adjacentes, mesmo que ocupados
     - *h2 seria o custo da solução "correta" neste jogo*

O custo da solução ótima de um problema relaxado é uma heurística admissível para o problema original!!!

8

Relaxando o problema - exemplo

- **Descrição original da ação:**

  Peça A pode mover do lugar A para o lugar B se A é adjacente (horizontal ou vertical) a B *e* B é lugar vazio.

- **Heurísticas:**
  1. Peça A pode mover do lugar A para o lugar B. (h1)
  2. Peça A pode mover do lugar A para o lugar B se A é adjacente a B. (h2)
  3. Peça A pode mover do lugar A para B se B é lugar vazio.

9

(2) Usando informação estatística

- **Funções heurísticas podem ser "melhoradas" com informação estatística:**
  - executar a busca com um conjunto de treinamento (e.g., 100 configurações diferentes do jogo), e computar os resultados.
    - ***se**, em 90% dos casos, quando* h (n) *= 14, a distância real da solução é 18, **então**, quando o algoritmo encontrar 14 para o resultado da função, vai substituir esse valor por 18.*
- **Informação estatística expande menos nós, porém elimina *admissibilidade:***
  - em 10% dos casos do problema acima, a **função de avaliação poderá superestimar o custo da solução**, não sendo de grande auxílio para o algoritmo encontrar a solução menos custosa.

10

(3) Aprendendo heurísticas por experiência

- **Resolve o jogo diversas vezes e computa custo da solução, relacionando a algum atributo relevante do problema.**
  - Ex1: atributo x1(n) = número de peças fora do lugar no início do jogo; Para cada valor de atributo, ver custo médio da solução.
    - *Ex: para x1(n)=5, resolvo 100 vezes o problema e vejo que custo médio da solução é 14 passos*
  - Ex2: atributo x2(n) = número de pares de peças adjacentes que também são adjacentes na configuração de solução
- **➔ uso x1(n) ou x2(n) para estimar h(n) ou a combinação entre elas: h(n) = c1.x1(n) + c2.x2(n), ajustando c1 e c2 da melhor forma para os dados de custo da solução.**

11