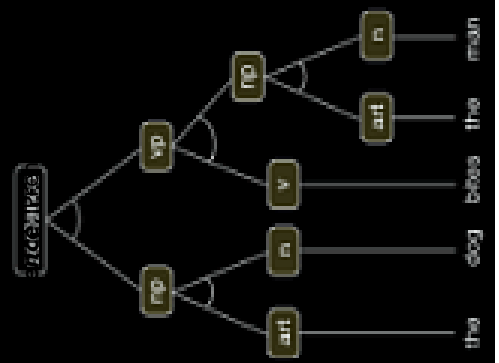
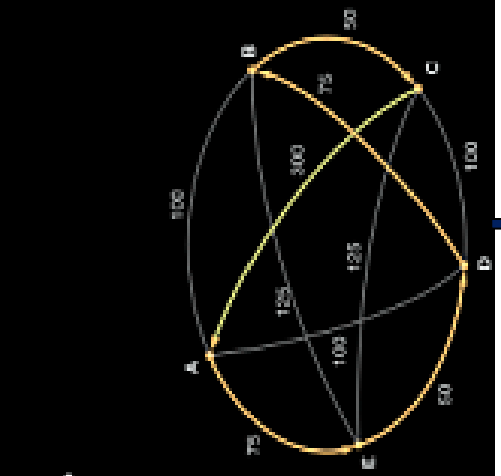
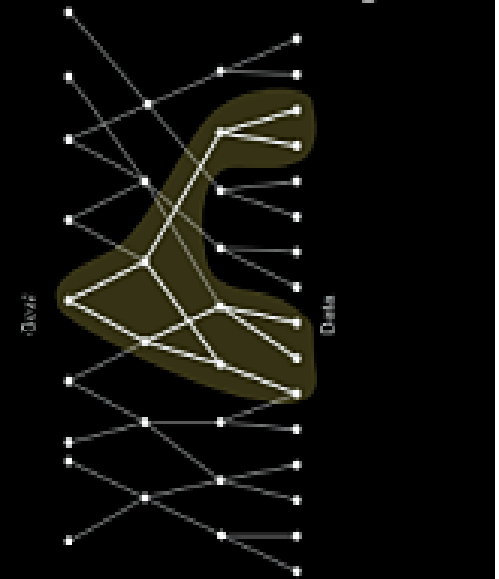






Universidade Federal de Campina Grande

Departamento de Sistemas e Computação

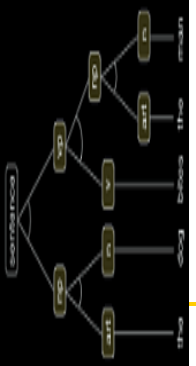
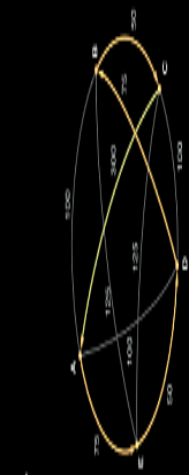
Pós-Graduação em Ciência da Computação

Inteligência Artificial

Resolução de Problemas (Parte IV)

Prof.ª Joseana Macêdo Fechine Régis de Araújo

joseana@computacao.ufcg.edu.br


Em Busca de Soluções

Tópico

- Busca Heurística
  - **Como escolher uma boa função heurística *h?***

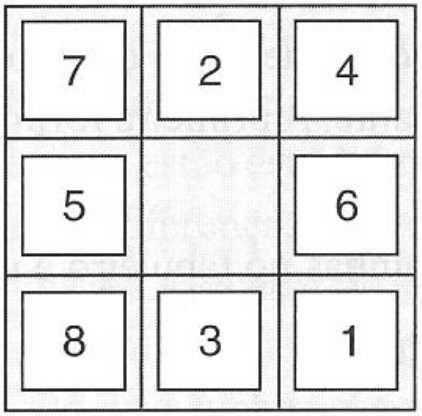
Busca Heurística

Como escolher uma boa função heurística *h*?

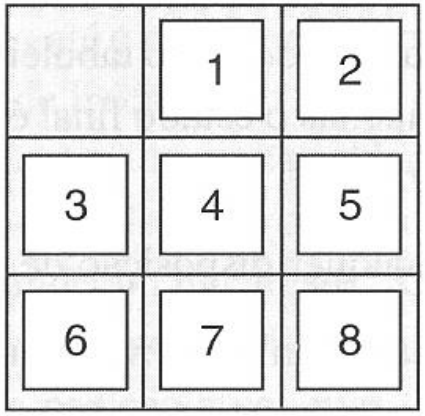
- ***h* depende de cada problema particular.**
- ***h* deve ser *admissível***
  - não superestimar o custo real da solução
- **Exemplo: jogo dos 8 números**
  - um número pode mover-se de A para B se A é adjacente a B e B está vazio
  - busca exaustiva:
    - solução média em 22 passos
    - fator de ramificação médio: 3
    - ≈ 3^{22} estados possíveis
    - ≈ 9!/2 (controlando os estados repetidos)

4	5	8
	1	6
7	2	3

Busca Heurística



Estado inicial



Estado objetivo

Distância de Manhattan: distância pombalina, distância de quarteirões ou distância de táxi. Recebeu o nome pois define a menor distância possível que um carro é capaz de percorrer numa malha urbana reticulada ortogonal, como se encontram em zonas como Manhattan ou a Baixa Pombalina.

Algumas heurísticas possíveis:

- h_1 = nº de elementos em posições erradas

$$h_1 = 8$$

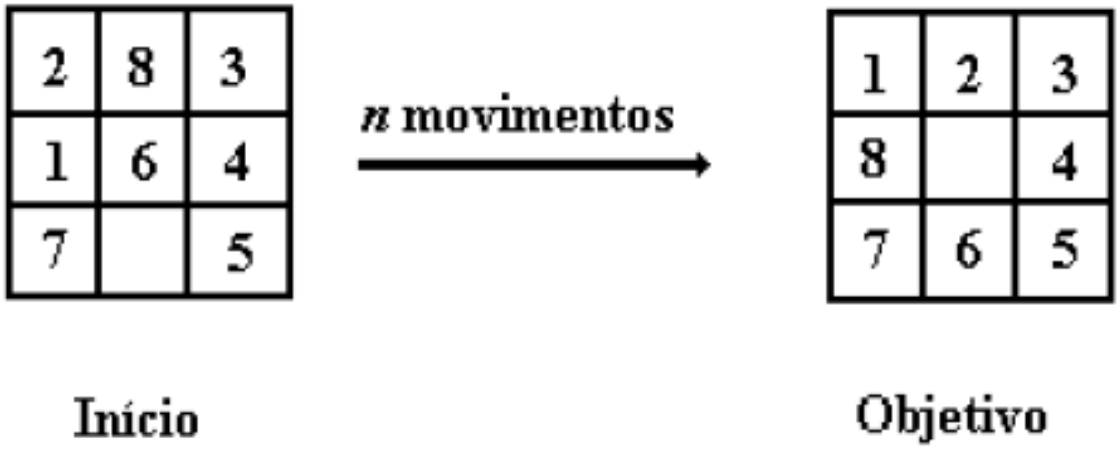
- h_2 = soma das distâncias de cada elemento à posição final - objetivo (*city block distance* - *Manhattan distance*)

$$h_2 = 3+1+2+2+2+3+3+2=18$$

Busca Heurística

- Função $h_2(n)$
  - o espaço de estados gerado é menor $\Rightarrow$
  - o algoritmo acha mais rapidamente a solução.

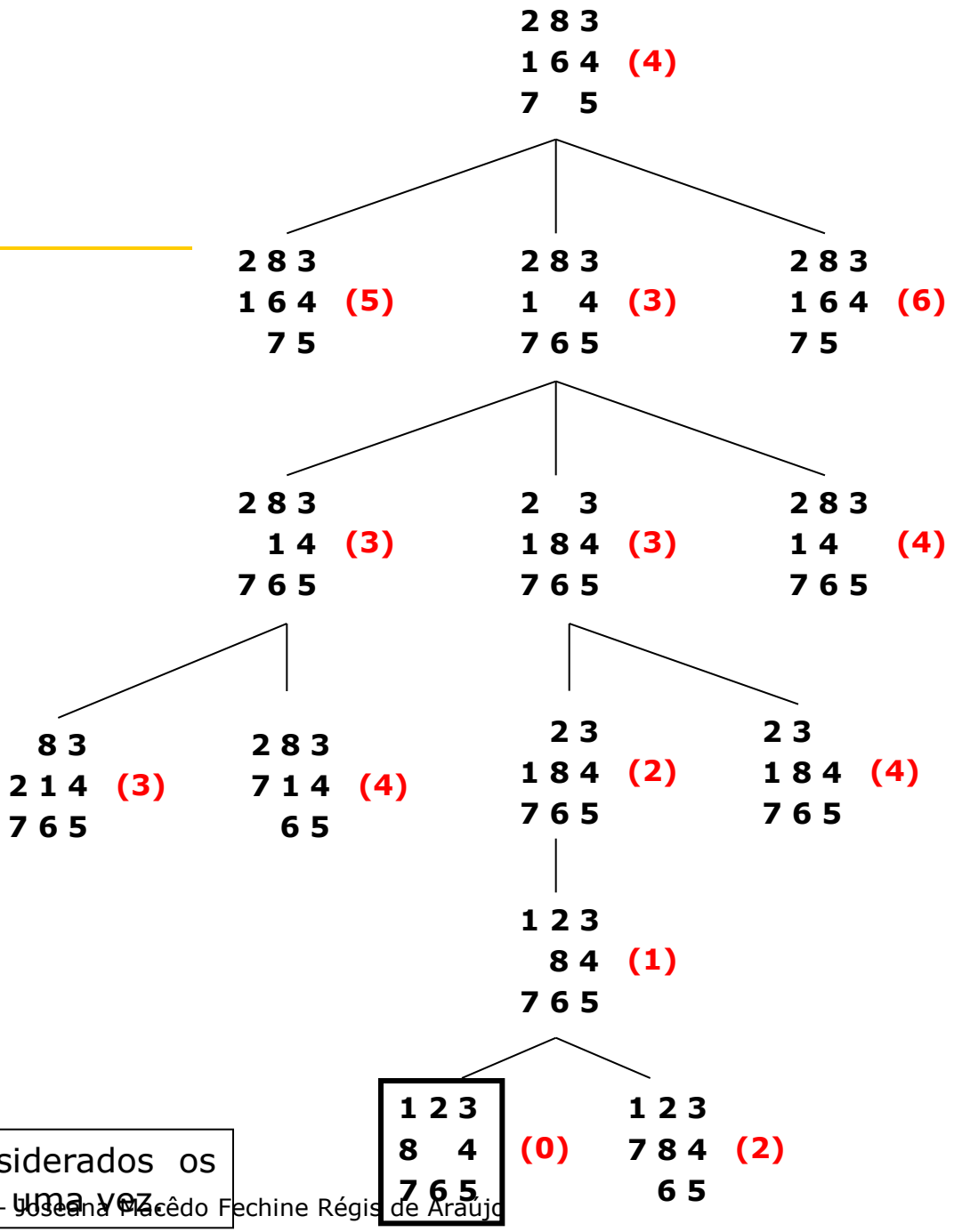
- **Exemplo:**



Exemplo:

- Espaço de estados gerado com $h_1(n)$ (para cada estado está indicado entre parênteses o valor da função heurística):

Estado Inicial	Estado Objetivo
2 8 3 1 6 4 7 5	1 2 3 8 4 7 6 5

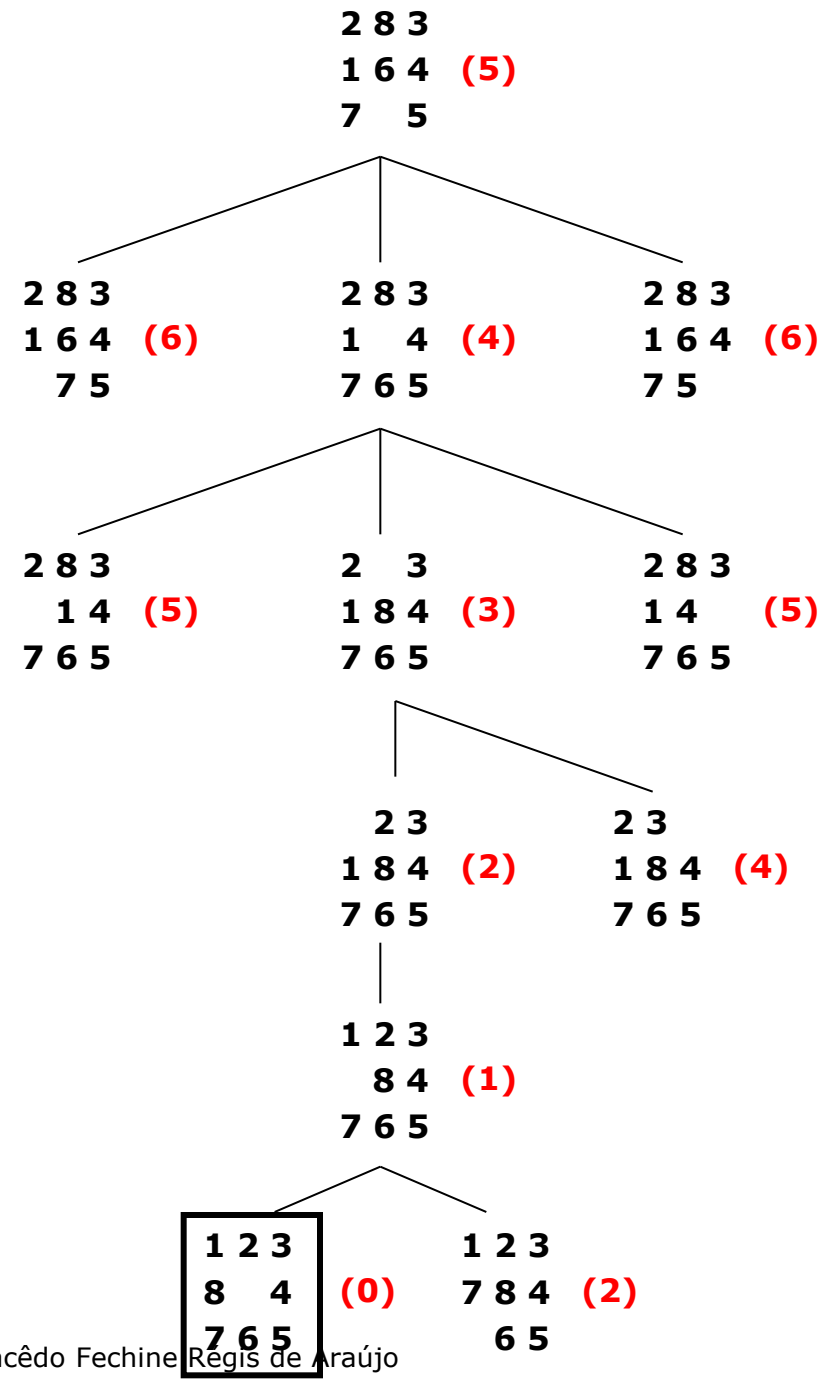


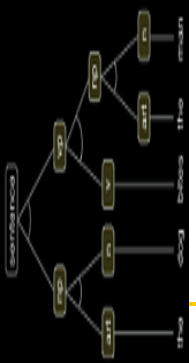
Neste exemplo não são considerados os nós que aparecem por mais de uma vez.

Exemplo:

- Espaço de estados gerado com $h_2(n)$

Estado Inicial	Estado Objetivo
2 8 3 1 6 4 7 5	1 2 3 8 4 7 6 5





Busca Heurística

Efeito da exatidão da heurística sobre o desempenho

- **Qualidade da função heurística:** medida a partir do fator de expansão efetivo (*b** ou fator de ramificação efetiva).
  - *b** - fator de expansão de uma árvore uniforme com *N+1* nós e nível de profundidade *d*

$$N+1 = 1 + b^* + (b^*)^2 + \ldots + (b^*)^d \ ,$$

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.

Busca Heurística

Comparação entre os custos da busca e entre os fatores de expansão (ramificação) efetivo para os algoritmos BUSCA-POR-APROFUNDAMENTO-ITERATIVO e A* com h_1, h_2. A média dos dados é calculada sobre 100 instâncias de quebra-cabeça de 8 peças para diversos comprimentos de solução.

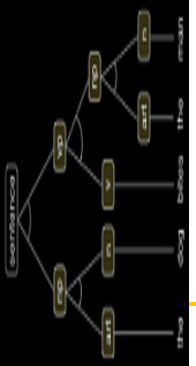
	Custo da busca			Fator de ramificação efetiva		
d	BAI	A*(h_1)	A*(h_2)	BAI	A*(h_1)	A*(h_2)
2	10	6	6	2,45	1,79	1,79
4	112	13	12	2,87	1,48	1,45
6	680	20	18	2,73	1,34	1,30
8	6384	39	25	2,80	1,33	1,24
10	47127	93	39	2,79	1,38	1,22
12	3644035	227	73	2,78	1,42	1,24
14	–	539	113	–	1,44	1,23
16	–	1301	211	–	1,45	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

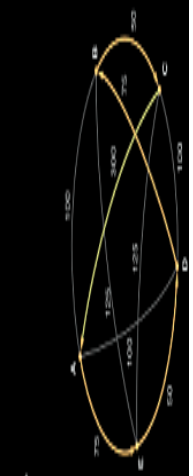
- **Uma boa função heurística → *b** muito próximo de 1.**
- **h_2 é melhor que h_1 → para qualquer nó, $h_2(n) \geq h_1(n)$!**

Busca Heurística

- **É sempre melhor utilizar uma função heurística com valores mais altos, desde que ela seja *admissível* e que o tempo para computá-la não seja muito grande!**
  - Exemplo: h_2 melhor que h_1

- h_i domina $h_k \Rightarrow h_i(n) \geq h_k(n), \forall n$ **no espaço de estados**
  - No exemplo anterior: h_2 domina h_1 ,
  - **Isso pode ser traduzido na forma**: A heurística 2 é melhor que a heurística 1, pois terá um menor fator de ramificação. Desde que h_1 e h_2 não superestimem o custo real.





Busca Heurística

- **Caso existam muitas funções heurísticas para o mesmo problema, e nenhuma delas domine as outras, usa-se uma *heurística composta*:**

$$h(n) = max(h_1(n), h_2(n), \ldots, h_m(n))$$

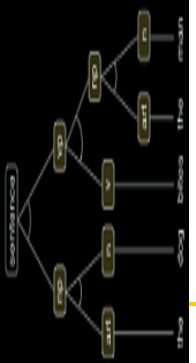
Busca Heurística

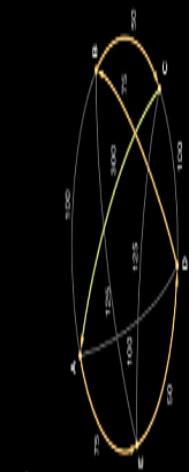
Exemplo de estratégias genéricas para definir *h*:

1) Relaxar o problema (versão simplificada);

2) Usar informação estatística;

3) Identificar os atributos relevantes do problema e usar aprendizagem.





Busca Heurística

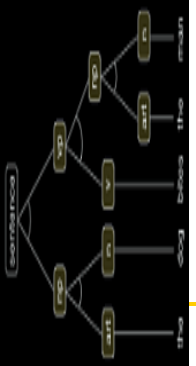
Jogos simples, como o jogo-dos-8, são veículos ideais para explorar o projeto e o comportamento de algoritmos de busca heurística por várias razões:

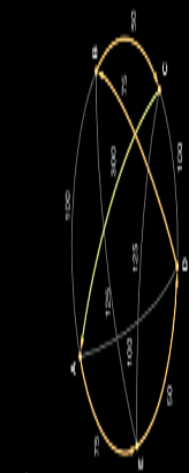
1. Os espaços de busca são grandes o suficiente para requererem poda heurística.

2. A maioria dos jogos é suficientemente complexa para sugerir uma rica variedade de avaliações heurísticas para comparação e análise.

3. Os jogos geralmente não envolvem questões representacionais complexas.

4. Como todos os nós do espaço de estados têm uma representação comum, pode-se aplicar uma heurística única para todo o espaço de busca.





Busca Heurística

- Problemas mais realistas complicam bastante a implementação e a análise da busca heurística, requerendo heurísticas múltiplas.

- O entendimento ganho de jogos simples pode ser generalizado para problemas como aqueles encontrados em sistemas especialistas, planejamento, controle inteligente e aprendizado. Porém, uma heurística simples não pode ser aplicada a todos os estados nestes domínios.