

Inteligência Artificial IA

Prof. João Luís Garcia Rosa

I. MÉTODOS DE BUSCA

2004

V1.4

Sistema de Produção

- Sistema de Produção:
 - Base de dados global
 - Regras de produção
 - Estratégia de controle
- Exemplo: tabuleiro de 8 pedras

Sistema de Produção

2	8	3
1	6	4
7		5

Inicial



1	2	3
8		4
7	6	5

Meta

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 3

Sistema de Produção

■ Procedimento *PRODUÇÃO*

- 1 *DADOS* ← base de dados inicial
- 2 até *DADOS* satisfazer a condição de terminação, faça:
- 3 *início*
- 4 selecione alguma regra, R, no conjunto de regras que possa ser aplicada a *DADOS*
- 5 *DADOS* ← resultado da aplicação de R a *DADOS*
- 6 *fim*

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 4

Sistema de Produção

■ Controle

- Passo 4: maior problema
- Estratégias não-informadas
- Estratégia informada (heurística)
 - *hill-climbing* para o tabuleiro de 8 pedras

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 5

Sistema de Produção

-4

2	8	3
1	6	4
7		5



-3

2	8	3
1		4
7	6	5



-3

2		3
1	8	4
7	6	5



-2

	2	3
1	8	4
7	6	5



-1

1	2	3
	8	4
7	6	5



0

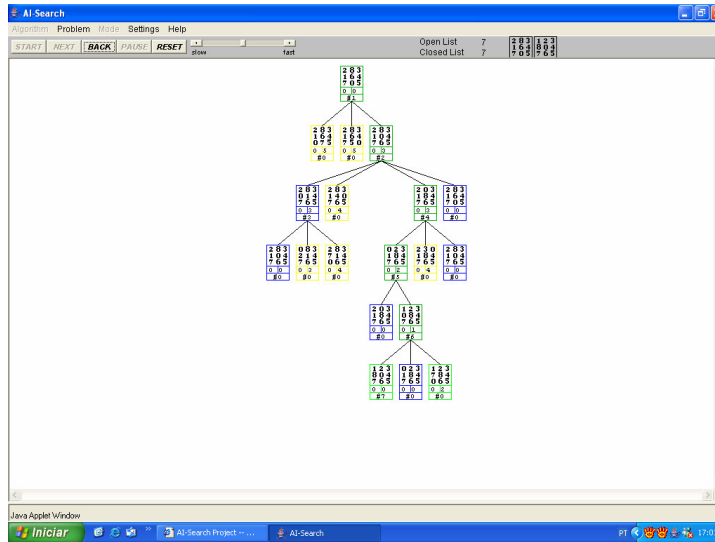
1	2	3
8		4
7	6	5

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 6

<http://www.cs.rmit.edu.au/AI-Search/Product/>



João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 7

Sistema de Produção

- Regime de controle:
 - A forma como é conduzido um processo de busca
 - Dois tipos principais:
 - Irrevogável
 - tentativo

João Luís G. Rosa

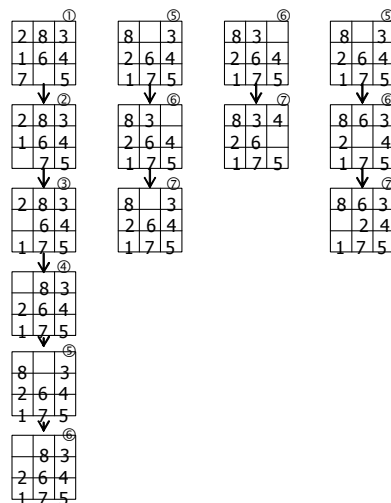
<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 8

■ Estratégias de controle

- Dentro do regime de controle existem as *estratégias de controle*.
- Dois tipos de estratégia de controle dentro do regime de controle tentativo:
 - *Backtracking*
 - Busca em grafos

Backtracking



Backtracking

- **Procedimento Recursivo BACKTRACK (DADOS) (Nilsson, 1982)**
 - 1 se $TERMO(DADOS)$, retorne
 - 2 se $DEADEND(DADOS)$, retorne FALHA
 - 3 $REGRAS \leftarrow REGRASAPL(DADOS)$
 - 4 LOOP: se $NULL(REGRAS)$, retorne FALHA
 - 5 $R \leftarrow PRIMEIRA(REGRAS)$
 - 6 $REGRAS \leftarrow CAUDA(REGRAS)$
 - 7 $RDADOS \leftarrow R(DADOS)$
 - 8 $CAMINHO \leftarrow BACKTRACK(RDADOS)$
 - 9 se $CAMINHO = FALHA$, vá para LOOP
 - 10 retorne $CONC(R, CAMINHO)$

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 11

Backtracking

- **Problema das 8 rainhas**

	1	2	3	4	5	6	7	8
1	♥							
2			♥					
3					♥			
4		♥						
5				♥				
6								
7								
8								

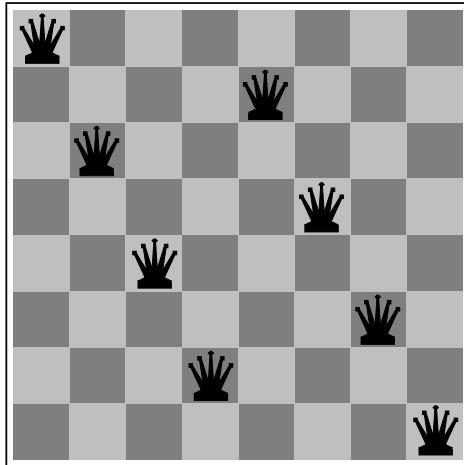
João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 12



Problema das 8 rainhas



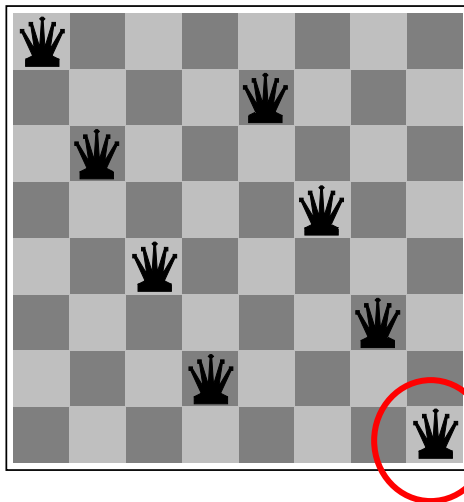
João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 13



Problema das 8 rainhas



João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 14

Problema das 8 rainhas

Uma solução depois de 91 *backtrackings*:

	1	2	3	4	5	6	7	8
1	♥							
2					♥			
3								♥
4						♥		
5			♥					
6							♥	
7		♥						
8				♥				

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 15

Busca em grafos

■ Busca em grafos

– No *backtracking*:

- o sistema de controle armazena somente o caminho correntemente sendo estendido.
- Um procedimento mais flexível envolve o armazenamento explícito de todos os caminhos de tal forma que qualquer um deles pode ser candidato a futura extensão: BUSCA EM GRAFOS.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 16

Busca em grafos

■ Notação de grafos:

- *Grafo*:
 - conjunto (não necessariamente finito) de *nós*.
- *Arcos direcionados*:
 - Certos pares de nós são conectados por *arcos*, e estes arcos são *direcionados* de um membro do par ao outro (*grafo direcionado*).
- *Sucessor/pai*:
 - Se um arco é direcionado do nó n_i para o nó n_j , então o nó n_j é o *sucessor* do nó n_i , e o nó n_i é o *pai* do nó n_j .
- *Árvore*:
 - caso especial de um grafo no qual cada nó tem, no máximo, um pai.
- *Raiz*:
 - Um nó na árvore que não tem pai é chamado de *nó raiz*.
- *Folha*:
 - Um nó na árvore que não tem sucessores é chamado de *nó folha*.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 17

Busca em grafos

- *Profundidade*:
 - Diz-se que o nó raiz é de *profundidade zero*. A profundidade de qualquer outro nó na árvore é, por definição, a profundidade de seus pais mais 1.
- *Caminho*:
 - Uma seqüência de nós (n_1, n_2, \dots, n_k), com cada n_j um sucessor de n_{j-1} para $j = 2, \dots, k$, é chamada de um *caminho de comprimento k* do nó n_1 para o nó n_k . Se um caminho existe entre o nó n_i para o nó n_j , então o nó n_j é acessível a partir do nó n_i .
- *Descendente e ancestral*:
 - O nó n_j é então um *descendente* do nó n_i , e o nó n_i é um *ancestral* do nó n_j .
- *Custo*:
 - Frequentemente é conveniente atribuir custos positivos aos arcos, para representar o custo da aplicação da regra correspondente. Usa-se a notação $c(n_i, n_j)$ para denotar o custo de um arco direcionado do nó n_i para o nó n_j . O custo de um caminho entre dois nós é a soma dos custos de todos os arcos que conectam os nós no caminho. Em alguns problemas, deseja-se achar o caminho que tenha custo *mínimo* entre dois nós.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 18



Busca em grafos

- Para este modelo, os nós são rotulados por base de dados e os arcos são rotulados por regras.
- Note que o problema de achar uma seqüência de regras para transformar uma base de dados em outra é equivalente ao problema de achar um caminho num grafo.
- No tipo mais simples de problema, deseja-se achar um caminho (talvez tendo custo mínimo) entre um nó dado s , representando a base de dados inicial e um outro nó dado t , representando alguma outra base de dados. A situação mais usual envolve achar um caminho entre um nó s e *qualquer* membro do conjunto de nós $\{t\}$ que representa as bases de dados que satisfazem a condição terminal. Chama-se o conjunto $\{t\}$ o conjunto meta e cada nó t em $\{t\}$ é um *nó meta*.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluiz/ia.html>

IA-2004-I slide 19



Busca em grafos

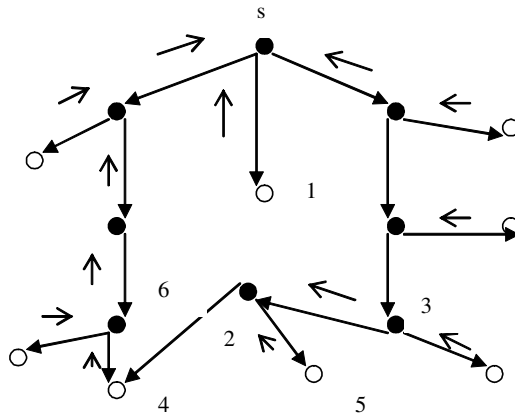
- Procedimento BUSCA-EM-GRAFOS (Nilsson, 1982)
 - 1 Crie um grafo de busca, G , consistindo somente do nó inicial, s . Ponha s numa lista chamada ABERTOS.
 - 2 Crie uma lista chamada FECHADOS que está inicialmente vazia.
 - 3 LOOP: se ABERTOS está vazia, saia com falha.
 - 4 Selecione o primeiro nó em ABERTOS, remova-o de ABERTOS, o ponha-o em FECHADOS. Chame este nó de n .
 - 5 Se n é um nó meta, saia com sucesso com a solução obtida traçando um caminho entre os ponteiros de n para s em G . (Os ponteiros são estabelecidos no passo 7.)
 - 6 Expanda o nó n , gerando o conjunto, M , de seus sucessores que não são ancestrais de n . Instale estes membros de M como sucessores de n em G .
 - 7 Estabeleça um ponteiro para n a partir daqueles membros de M que ainda não estejam nem em ABERTOS nem em FECHADOS. Adicione estes membros de M a ABERTOS. Para cada membro de M que já esteja em ABERTOS ou FECHADOS, decida se direciona ou não seu ponteiro para n . Para cada membro de M já em FECHADOS, decida para cada um de seus descendentes em G se redireciona ou não o seu ponteiro.
 - 8 Reordene a lista ABERTOS, de acordo com alguma esquema arbitrário ou de acordo com mérito heurístico.
 - 9 Vá para LOOP.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluiz/ia.html>

IA-2004-I slide 20

Busca em grafos



João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 21

Estratégias não-informadas de busca em grafos

- Procedimentos não-informados de busca em grafos
 - As estratégias não-informadas usam apenas a informação disponível na definição do problema
 - Busca em largura
 - Busca em profundidade
 - Outras.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 22

Estratégias não-informadas de busca em grafos

■ Estratégias de busca:

- Uma estratégia é definida através da *ordem da expansão dos nós*
- Avaliação:
 - Completeza: sempre acha a solução se ela existir?
 - Complexidade temporal: número de nós gerados / expandidos
 - Complexidade espacial: número máximo de nós na memória
 - Otimalidade: sempre acha a solução de custo mínimo?
- b = fator de ramificação máximo
- d = profundidade da solução de menor custo
- m = profundidade máxima do espaço de estados (pode ser ∞)

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 23

Estratégias não-informadas de busca em grafos

■ Propriedades da Busca em Largura:

- Completa? Sim (se b é finito)
- Tempo? $1 + b + b^2 + \dots + b^d + b(b^d - 1) = O(b^{d+1})$
- Espaço? $O(b^{d+1})$ (mantém todo nó na memória)
- Ótima? Sim (se custo = 1 por passo); não ótima em geral
- *Espaço* é o maior problema: pode facilmente gerar nós a 10MB/s, tal que 24h = 860GB.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 24



Estratégias não-informadas de busca em grafos

■ Propriedades da Busca em Profundidade:

- Completa? Não: falha em espaços de profundidade infinita, espaços com loops. Modificar para evitar estados repetidos no caminho → completa em espaços finitos
- Tempo? $O(b^m)$ terrível se m é muito maior que d , mas se soluções são densas, pode ser mais rápido que a busca em largura.
- Espaço? $O(bm)$ i. e. espaço linear!
- Ótima? Não.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 25



Estratégias heurísticas de busca em grafos

■ Procedimentos heurísticos de busca em grafos

- A palavra *heurística* vem da palavra grega *heuriskein*, que significa “descobrir”, que é também a origem de *eureka*, a famosa exclamação de Arquimedes (“Eu achei”)

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 26

Estratégias heurísticas de busca em grafos

■ *Problema do caixeiro viajante:*

- Um vendedor tem uma lista de cidades, que ele deve visitar apenas uma vez. Existem estradas diretas ligando cada par de cidades da lista. Ache a rota que o vendedor deve seguir para a viagem mais curta possível que começa e termina em uma das cidades.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 27

Estratégias heurísticas de busca em grafos

■ Heurística do vizinho mais próximo:

- 1. Arbitrariamente selecione uma cidade inicial.
- 2. Para selecionar a próxima cidade, olhe todas as cidades ainda não visitadas e selecione a mais próxima a cidade corrente. Vá a ela.
- 3. Repita o passo 2 até que todas as cidades tenham sido visitadas.
- Este procedimento é executado em tempo proporcional a N^2 , uma melhora significativa sobre $N!$, que seria o tempo gasto caso fosse usada uma estratégia não informada (que geraria uma *explosão combinatorial*).

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 28

Heurísticas

- Argumentos a favor do uso de heurística:
 - Raramente precisamos da solução ótima; uma boa aproximação normalmente serve muito bem. De fato, existe alguma evidência de que as pessoas, quando resolvem problemas, ao invés de serem otimizadoras, elas procuram a satisfação. Em outras palavras, elas procuram qualquer solução que satisfaça algum conjunto de requisitos, e assim que elas encontram um elas terminam.
 - Ainda que as aproximações produzidas por heurísticas possam não ser muito boas no pior caso, os piores casos raramente acontecem na vida real.
 - A tentativa de entender por que uma heurística funciona, ou por que ela não funciona, freqüentemente leva a um entendimento mais aprofundado do problema.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 29

Métodos de Busca

- Exercício:
 - Especifique uma base de dados global, regras e uma condição de terminação para um sistema de produção para resolver o seguinte problema dos jarros de água:
 - Dado um jarro de 5 litros cheio de água e um jarro de 2 litros vazio, como se pode obter precisamente 1 litro no jarro de 2 litros? A água pode ser desperdiçada ou transferida de um jarro para outro; entretanto, só os 5 litros iniciais estão disponíveis.

João Luís G. Rosa

<http://docentes.puc-campinas.edu.br/ceatec/joaoluis/ia.html>

IA-2004-I slide 30