ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Síntese do trabalho prático 2

Inteligência Artificial para Sistemas Autónomos

Introdução ao projeto

Este projeto, visa a construção de uma biblioteca em Java que produz a resolução de puzzles com peças deslizantes, tendo em conta certas configurações.

Iremos verificar alguns dos métodos de pesquisa em árvore de modo à biblioteca suportar diversos tipos de problemas.

Raciocínio Automático

Domínio do problema	**Sistema Inteligente**
Problema	Representação interna do problema (Modelo "do mundo")
Objetivo	Mecanismo de raciocínio
Solução	Representação da solução

De modo a encontrar uma solução ao problema, a biblioteca utiliza um racíocinio prospectivo, através de uma simulação interna do mundo, criada em forma de árvore.

Esra árvore possuí estados que representam as possíveis situações/configurações (ex. configuração do puzzle). Com o intuito de percorrer a árvore, aplicam-se operadores aos estados. Operadores são transformações que se podem aplicar a estados.

O espaço de estados de um sistema representa todos os estados com todas as opções possíveis.

O espaço de estados tem o seguinte aspecto:

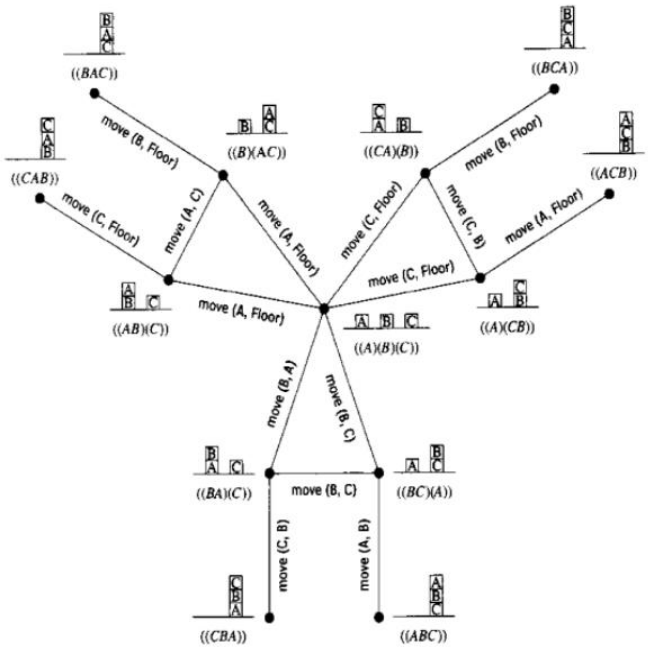

Figura 1 Exemplo de espaço de estados

Raciocínio através de procura

Este tipo de raciocínio explora sucessivamento o espaço de estados até encontrar uma resposta. Através deste mesmo raciocínio temos os seguintes conceitos:

- **Situação**

  Espaço de estados

- **Ação**

  Operador

- **Raciocínio**

  Simulação interna do mundo

Métodos de procura

Para compreender o método de procura que necessitamos, temos de ter em noção os seguintes aspetos:

- **Completo**

  Garante que, caso a existência de uma solução, a mesma será encontrada

- **Ótimo**

  Garante que, caso existam várias soluções, a melhor solução será encontrada.

- **Complexidade**

  1. Tempo

     Tempo necessário para encontrar uma solução.

  2. Espaço

     Memória necessária para encontrar uma solução

Após a explicação dos tipos de características presentes em estratégias de procura em árvora, podemos verificar algumas das suas variantes:

- **Estratégia de controlo**
    1. Explorar primeiro os nós mais recentes
    2. Explorar primeiro os nós mais antigos
- **Procura em profundidade**

    Este tipo de procura, começa por expandir sempre o primeiro nó, até encontrar uma resposta. Se nenhuma resposta for encontrada começasse a expandir um nó paralelo.

    Este método nem sempre encontra uma solução o que o faz ser **não completo** (caso o ramo a ser extendido seja infinito, nenhuma resposta será localizada).

    - **Procura em profundidade limitada**

        Limita a procura em profundidade até uma certa profundidade.

    - **Procura em profundidade iterativa**

        Como a procura em profundidade limitada, esta procura tem uma profundidade limitada. No entanto, caso a solução não seja encontrada, será feita outra procura com profundidade limitada. Isto continua até encontrar eventualmente uma solução.

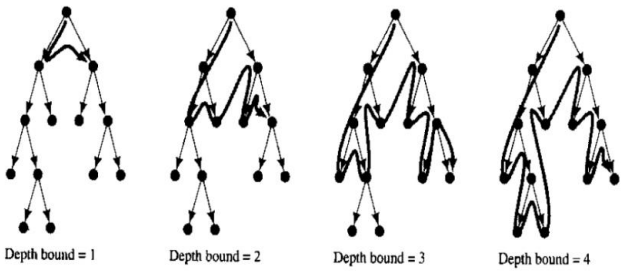

Figura 2 Exemplo de procura em profundidade iterativa na árvore

- **Procura em largura**

  Este género de procura, não explora outro nível de profundidade enquanto não expandir todos os nós com um certo nível de profundidade.

  Este método utiliza uma grande **complexidade de tempo**, pois todos os nós de um sistema vão sendo expandidos (de nível em nível). No entanto, a procura em largura é **ótima**, pois caso haja uma solução, a mesma será encontrada.

  - **Procura bidirecional**

    Esta procura consiste em "correr" duas procuras em largura, uma no início do percurso e outra no final do mesmo até chegar à solução.

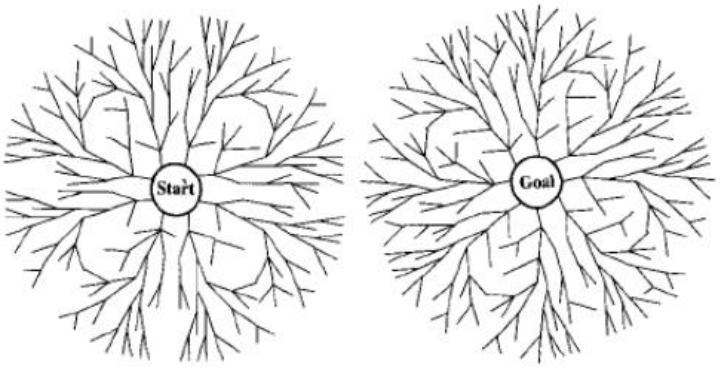

Figura 3 Representação esquemática de uma procura bidirecional

- **Procura de custo uniforme**

  Este tipo de procura, explora primeiro caminhos com o menor custo garantindo assim uma procura ótima.

Todos os exemplos acima demonstrados pertencem à categoria de **procura não informada**. Este género de procura não tira partido de conhecimento do domínio do problema, nem é guiada.

De seguida observarei métodos de **procura informada**, ou seja, que tiram partido do conhecimento do domínio do problema, sendo uma procura guiada.

Procura em grafos com ciclos

Como podemos rapidamente verificar, que podemos estar a observar nós repetidos. Isto acontece quando as ações correspondentes às transições de estado são reversíveis.

Desta forma, ao expanditmos nós anteriormente analisados, estamos a desperdiçar recursos da máquina, que fazem o programa correr mais lento.

Função heurística h(n)

Representa uma esimativa do custo do percurso desde o nó n até ao nó objetivo.

Alguns dos métodos de procura informada apresentam-se a seguir:

- **Procura melhor-primeiro**

  Utiliza uma função ***f*** para avaliação de cada nó ***n*** gerado. Esta função tipicamente representa uma estimativa do custo da solução através do nó ***n***

  - **Procura de Sôfrega** (Greedy Search) [$f(n) = h(n)$]

    Não tem em conta o custo do percurso

    Minimização do custo local

    Não é **ótima**

  - **Procura A*** [$f(n) = g(n) + h(n)$]

    Minimização do custo global

Tabela de análise de desempenho

Tabela 1 Tabela de análise de desempenho da biblioteca

	Puzzle	Número Iterações	Complexidade Temporal	Complexidade Espacial	Heurística
Profundidade	A	40	160	35	
	B	50	535228	43	
Profundidade Iterada	A	18	1924	18	
	B	30	1277344	26	
Largura	A	14	14576	2156	
	B	26	657596	24053	
Custo Uniforme	A	14	18388	2487	
	B	26	638004	24204	
Sôfrega	A	18	108	25	Peças fora do Lugar
	A	24	164	36	Distância de Manhattan
	B	132	5088	796	Peças fora do Lugar
	B	70	588	99	Distância de Manhattan
A*	A	14	332	54	Peças fora do Lugar
	A	14	808	146	Distância de Manhattan
	B	26	8444	1148	Peças fora do Lugar
	B	26	146968	15603	Distância de Manhattan

Docente: Engº Luís Morgado

Arman Khan de Freitas

Turma 41D

Nº 45414