Inteligência Artificial Para Sistemas Autônomos

Andre Fonseca

a39758@alunos.isel.pt

Prof. Paulo Vieira

pjvieira@deetc.isel.pt

INTRODUÇÃO

O segundo projecto implementa a resolução autónoma do jogo *8 puzzle,* através de algoritmos de estratégias de pesquisa em árvore. O problema inicial apresenta duas configurações do puzzle - A e B - para serem resolvidas; contudo, o principal objectivo consiste em que os algoritmos permitam a resolução de qualquer tipo de configuração e, até mesmo, de problema - cuja solução possa ser resolvida por estes algoritmos.

Uma solução é um conjunto de acções para um problema formulado. Por sua vez, as estratégias de pesquisa em árvore são algoritmos que permitem considerar múltiplas sequências de acções.

O puzzle consiste numa grelha de 3 por 3. Cada quadrado da grelha é composto por um número compreendido entre 1 e 8 e um quadrado vazio. O objectivo do jogo é ordenar de forma crescente os quadrados de cada número de forma a que o quadrado vazio esteja posicionado na última posição da grelha. Para tal, o quadrado vazio pode trocar de posição com qualquer número que esteja junto ao seus lados direito, esquerdo, por cima ou baixo.

Adicionalmente existe também outro problema que consiste no planeamento de caminhos de um sistema de transportes.

Links

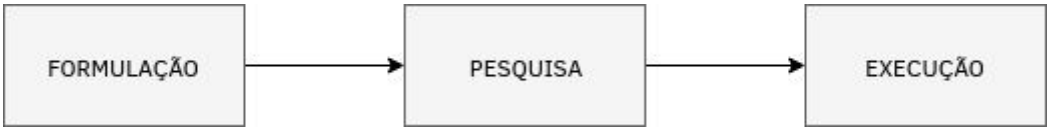
- https://github.com/andrewfonseca/ISEL-LEIM-IASA

FORMULAÇÃO DO PROBLEMA

A formulação do problema é o processo de decidir que acções e estados devem de ser considerados, tendo em conta um certo objectivo.

Os objectivos ajudam a organizar o comportamento - ao limitar o conjunto de acções que devem de ser consideradas. Um objectivo é definido como um conjunto de estados do ambiente que solucionam o problema. Para tal, torna-se necessário analisar o tipo de acções e estados que devem de ser considerados.

O processo de análise das sequências de acções que alcançam o objectivo é intitulado de pesquisa. Trata-se de um método de análise científica orientado para a procura da melhor forma de tomar decisões, a fim de conseguir os melhores resultados. Uma estratégia de pesquisa recebe um problema como entrada e devolve uma solução sob a forma de um conjunto de acções. Assim que uma solução é encontrada, as acções recomendadas podem ser executadas - trata-se da fase de execução.



Um problema pode ser definido formalmente pelos seguintes componentes:

- **Estado inicial:** o estado em que o agente se inicia;
- **Estados:** todos os estados possíveis de serem atingidos através de uma sequência de acções proveniente do estado inicial - espaço de estados;
- **Acções:** descrição das possíveis acções disponíveis ao agente. Dado um estado particular *s*, *ACTIONS(s)* retorna um conjunto de acções que podem ser executadas em *s*;
- **Modelo de transição:** descrição do resultado de cada acção definido por *RESULT(s, a)*;
- **Teste de objectivo:** determina se um dado estado *s* é o objectivo;
- **Custo de caminho:** função que atribui um custo numérico a um caminho relativo ao objectivo. Este valor é dado pelo somatório do custo de cada passo.

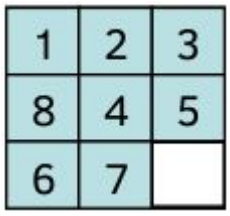
A solução de um problema é uma sequência de acções que transitam do estado inicial até ao objectivo. A qualidade da solução é medida pelo custo de caminho, por isso, a solução óptima é a que tem um menor custo de caminho.

FORMULAÇÃO DO PROBLEMA DE SISTEMA DE TRANSPORTES

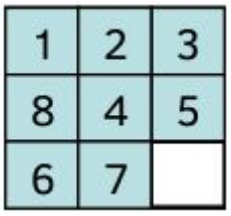
Localidade inicial	Localidade final	Custo
Loc-0	Loc-1	5
Loc-0	Loc-2	25
Loc-1	Loc-3	12
Loc-1	Loc-6	5
Loc-2	Loc-4	30
Loc-3	Loc-2	10
Loc-3	Loc-5	5
Loc-4	Loc-3	2
Loc-5	Loc-6	8
Loc-5	Loc-4	10
Loc-6	Loc-3	15

- **Estados:** uma localização pertencente ao conjunto de localizações;
- **Estado inicial:** Loc-0;
- **Acções:** deslocar-se de uma determinada localização para outra possível;
- **Modelo de transição:** quando numa determinada localização e ao aplicar uma acção, é retornada a próxima localização possível;
- **Teste de objectivo:** chegar a Loc-6;
- **Custo de caminho:** custo associado à deslocação entre localizações.

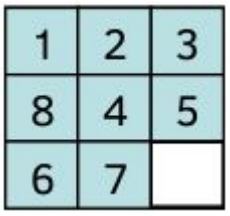
FORMULAÇÃO DO PROBLEMA 8-PUZZLE



Configuração inicial A



Configuração inicial B



Configuração final

- **Estados:** configuração do posicionamento dos quadrados da grelha;
- **Estado inicial:** qualquer configuração;

- **Acções:** movimento do quadrado em branco (esquerda, direita, cima e baixo) dependendo da sua posição;
- **Modelo de transição:** dado um determinado estado e uma acção, retorna o estado resultante;
- **Teste de objectivo:** verifica se a configuração actual corresponde à final;
- **Custo de caminho:** cada passo tem o custo de 1 unidade, então o seu total é o número de passos no caminho.

ESTRATÉGIAS DE PESQUISA

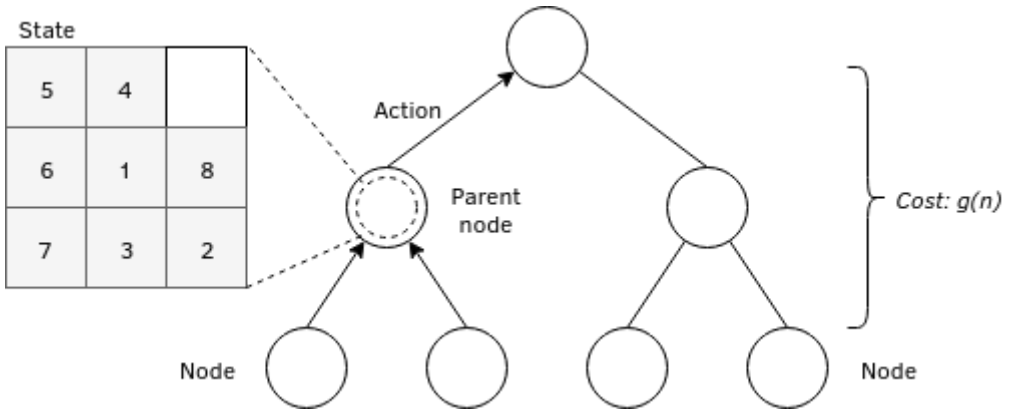
Uma solução é uma sequência de acções e os algoritmos de pesquisa analisam várias sequências de acções possíveis. As sequências de acções possíveis iniciam-se no nó-raiz que contém o estado inicial. Formam assim uma árvore de pesquisa com ramos que correspondem às acções e os nós expandidos por essas acções com os seus respectivos estados.

De maneira a analisar várias acções expande-se o nó actual aplicando as possíveis acções, e assim, obtém-se um novo conjunto de nós que são adicionados a uma estrutura de dados. A esta estrutura de dados chama-se de fronteira, que guarda em memória a ordem pela qual os nós devem de ser explorados. Este processo repete-se até que o nó-solução seja encontrado ou até que não exista mais nenhum nó para ser expandido.

A diferença entre os algoritmos das estratégias de pesquisa é a maneira como se escolhe o próximo nó a ser expandido.

De maneira a evitar caminhos redundantes é utilizada uma outra estrutura de dados que guarda em memória todos os estados e respectivos nós expandidos. Assim, todos os novos nós expandidos que coincidam com outros previamente explorados, ou seja, contidos na memória de explorados, podem ser descartados.

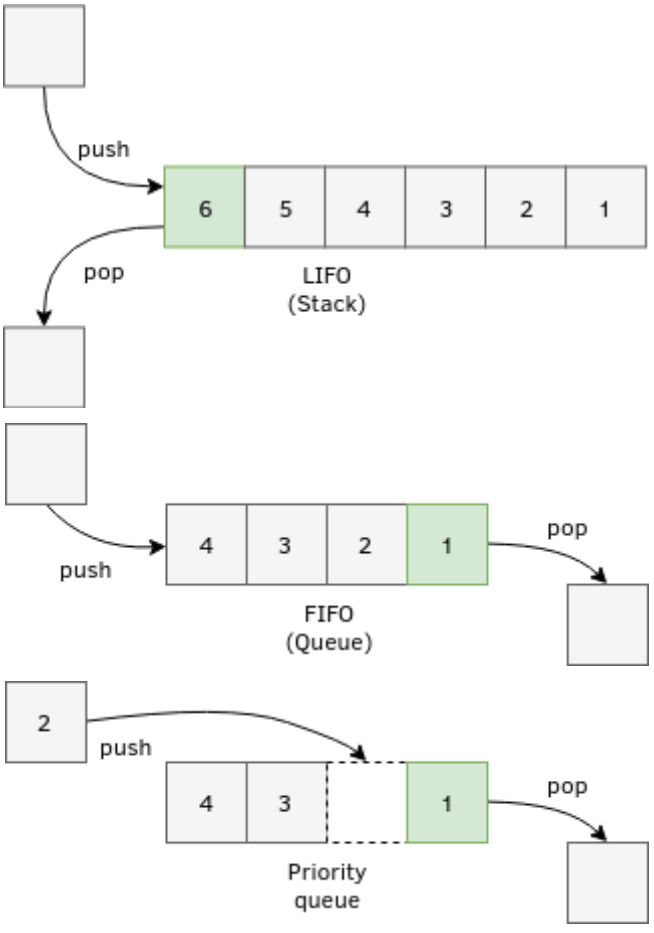
- **Estado:** estado que corresponde ao nó;
- **Pai:** o nó na árvore de pesquisa que gerou o nó actual;
- **Acção:** a acção que foi aplicada ao nó-pai que gerou o nó actual;
- **Custo de caminho:** o custo, denominado de $g(n)$, do nó inicial ao nó actual.



ESTRUTURAS DE DADOS

As estruturas de dados dividem-se em três variantes comuns:

- **FIFO:** *First In First Out*, remove o elemento mais antigo do contentor - *queue*;
- **LIFO:** Last In First Out, remove o elemento mais recente do contentor - *stack*;
- **Priority queue:** remove o elemento com maior prioridade segundo uma determinada ordem.



ANÁLISE DA ESTRATÉGIA DE PESQUISA

Ao existirem múltiplas estratégias de pesquisa é necessário atribuir critérios matemáticos para que estas possam ser analisadas e comparadas:

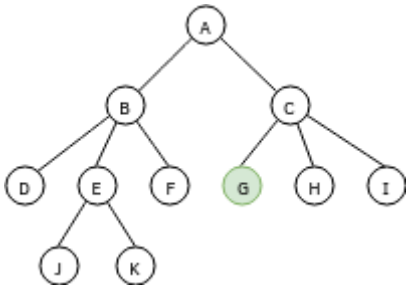
- **Completa:** garantia de ser encontrado uma solução;
- **Optima:** garantia de encontrar a solução óptima;
- **Complexidade espacial:** número máximo de nós armazenados na memória de fronteira, ou seja, quanta memória é necessária para executar o algoritmo;
- **Complexidade temporal:** número de nós explorados durante a pesquisa, ou seja, quanto tempo é necessário para realizar a pesquisa.

ESTRATÉGIAS DE PESQUISAS NÃO-INFORMADAS

As estratégias de pesquisas não-informadas, também chamadas de pesquisas cegas, não usam qualquer informação sobre os estados para além do que é fornecido pelo problema. As suas únicas funções são expandir e explorar nós sucessores e distinguir o estado-solução de outros estados.

BREADTH-FIRST SEARCH

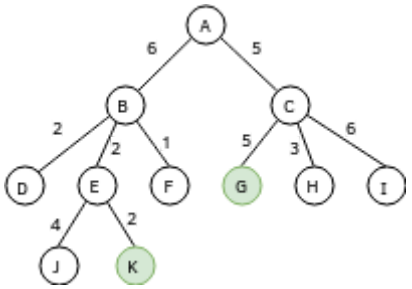
Breadth-first, ou “pesquisa em largura” em português, expande inicialmente o nó-raiz e, posteriormente, todos os seus sucessores consecutivamente após cada nível da árvore de procura. Esta pesquisa usa uma memória FIFO (queue) de maneira a explorar primeiro os nós mais superficiais.



No explorado	Fronteira	Explorados
	A	
A	B, C	
B	C, D, E, F	A
C	D, E, F, G, H, I	A, B
D	E, F, G, H, I	A, B, C
E	F, G, H, I, J, K	A, D, C, D
F	G, H, I, J, K	A, D, C, D, E
G	H, I, J, K	A, D, C, D, E, F

UNIFORM-COST SEARCH

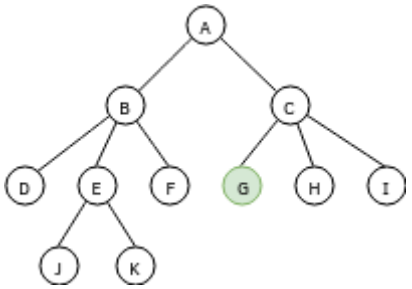
A pesquisa uniforme é uma adaptação da pesquisa em largura, sendo que a única diferença consiste na exploração do nó que tem menor custo. Isto acontece através da utilização de uma memória *priority queue* como fronteira.



No explorado	Fronteira	Explorados
	A	
A	C5, B6	
C	B6, H8, G10, I11	A
B	F7, H8, D8, E8, G10, I11	A, C
	H8, D8, E8, G10, I11	
F	D8, E8, G10, I11	A, C, B
H	E8, G10, I11	A, C, B, F
D	G10, K10, I11, J14	A, C, B, F, H
E	K10, I11, J14	A, C, B, F, H, D
G		A, C, B, F, H, D, E

DEPTH-FIRST SEARCH

Depth-first search, ou “pesquisa em profundidade” em português, expande o nó de maior profundidade da árvore de procura. Nesse sentido, esta pesquisa utiliza uma memória LIFO (stack) como fronteira.

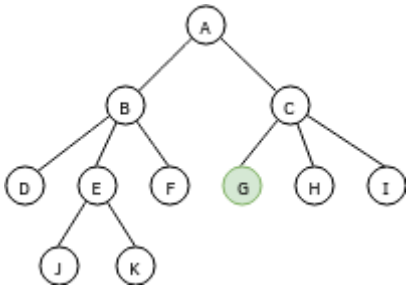


No explorado	Fronteira	Explorados
	A	
A	B, C	
B	D, E, F, C	A
D	E, F, C	A, B
E	J, K, F, C	A, B, D
J	K, F, C	A, B, D, E
K	F, C	A, B, D, E, J
F	C	A, B, D, E, J, K

C	G, H, I	A, B, D, E, J, K, F
G	H, I	A, B, D, E, J, K, F, C

ITERATIVE DEPTH-FIRST SEARCH

A pesquisa iterativa executa a pesquisa em profundidade iterando sobre o nível de profundidade. Começa pelo nível 0, depois o 1, 2 e assim sucessivamente até ser encontrado o nó-solução.



No explorado	Fronteira	Explorados
	A	
A		
		A
	A	
A	B, C	
B	C	A
C		A, B
		A, B, C
	A	
A	B, C	
B	D, E, F, C	A
D	E, F, C	A, B
E	F, C	A, B, D
F	C	A, B, D, E
C	G, H, I	A, B, D, E, F
G	H, I	A, B, D, E, F, C

COMPARAÇÃO DAS ESTRATÉGIAS DE PESQUISAS NÃO-INFORMADAS

Estratégia de pesquisa	Completa	Optima	Complexidade espacial	Complexidade temporal
BFS	Sim[1]	Sim[3]	$O(b^d)$	$O(b^d)$
UCS	Sim[1,2]	Sim	$O(b^{1+C*/\epsilon})$	$O(b^{1+C*/\epsilon})$
DFS	Não	Não	$O(b*m)$	$O(b^m)$
IDS	Sim[1]	Sim[3]	$O(b*d)$	$O(b^d)$

[1] Completa se *b* for finito

[2] Completa se custo de passo ≥ ϵ

[3] Óptimo se os custos de passos forem idênticos

b Número máximo de nós sucessores de qualquer nó - *branching factor*

d	Profundidade do nó que contém a solução - *depth*
C*	Custo da solução óptima
ϵ	Custo mínimo de uma transição de estado ($\epsilon > 0$)
m	Profundidade da árvore de procura

Ao comparar as quatro estratégias de pesquisa não-informadas conseguimos concluir:

- BFS: tem complexidade espacial exponencial - o que em termos computacionais é uma grande desvantagem, mas, por outro lado, garante que se encontra uma solução óptima (dada as condições);
- DFS: apesar de ter uma complexidade espacial menor do que a BFS, tem a desvantagem de não ser completa nem óptima, correndo assim o risco de não encontrar nenhuma solução;
- UCS: tem a capacidade de encontrar a solução óptima (dadas as condições), no entanto a complexidade espacial e temporal pode ser bastante mais elevada do que as restantes pesquisas;
- IDS: combina os benefícios da DFS e BFS em que a sua complexidade espacial é modesta tendo a capacidade de encontrar a solução óptima (dadas as condições).

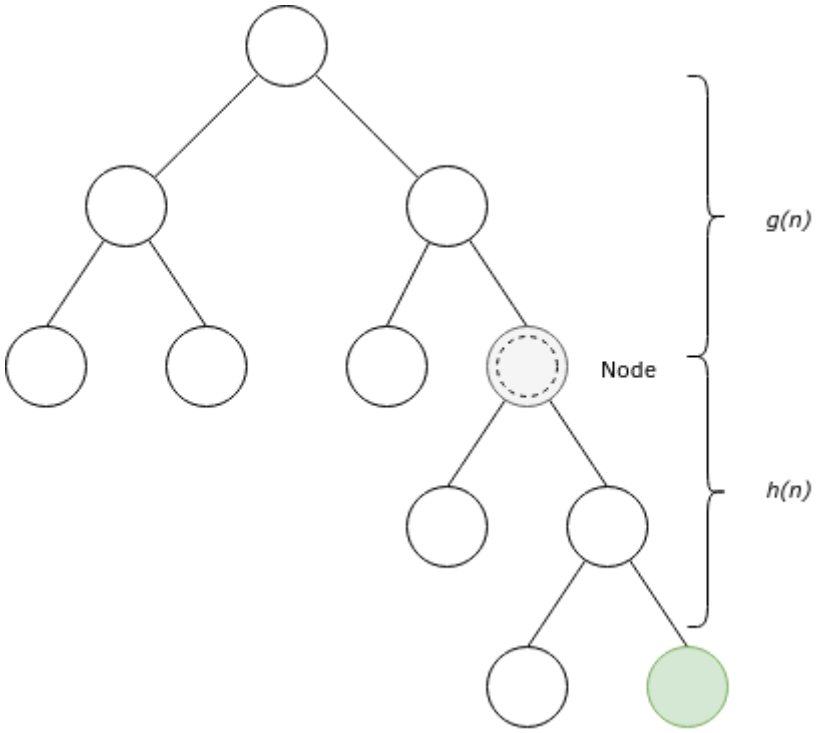
ESTRATÉGIAS DE PESQUISAS INFORMADAS

As estratégias de pesquisas informadas utilizam informação específica do problema que pode conter factores internos ou externos do problema. Assim, melhoram a sua eficiência em comparação às estratégias de pesquisa não-informadas.

A selecção do nó para expansão é realizada através de uma função de avaliação *f(n)*. Esta avaliação é construída através de uma estimativa, assim, o nó com menor valor de *f(n)* será expandido primeiro. A sua implementação é idêntica à estratégia UCS com excepção de ser usar *f(n)* em vez de *g(n)* para ordenar a memória *priority queue* que serve de fronteira.

A maioria dos algoritmos de estratégias de pesquisas informadas incluem como uma componente de *f(n)* uma função heurística, denominada de *h(n)* - sendo *h(n)* o menor valor estimado do caminho entre o estado do nó actual até ao estado do nó-solução.

Uma heurística deve de ser admissível, nunca deve de superestimar o custo de atingir o nó-solução, isto é, ser optimista. Pode-se considerar admissível se valor proveniente de *h(n)* for menor ou igual do que o valor real necessário para atingir o objectivo. No caso de um problema de distâncias entre vários pontos o valor de *h(n)* admissível seria a distância euclidiana entre dois pontos.



GREEDY BEST-FIRST SEARCH

Este algoritmo expande o nó que esteja mais próximo do objectivo através da avaliação da função heurística *f(n) = h(n)*. A cada passo tenta aproximar-se o mais possível do nó-solução.

A* SEARCH

Este é o algoritmo mais conhecido deste tipo de pesquisas. Os nós são avaliados através do mínimo valor da combinação de g(n) e h(n):

$$f(n)=g(n)+h(n)$$

COMPARAÇÃO DAS ESTRATÉGIAS DE PESQUISAS INFORMADAS

Estratégia de pesquisa	Completa	Optima	Complexidade espacial	Complexidade temporal
Greedy BFS	Não	Não	$O(b^m)$	$O(b^m)$
A*	Sim	Sim	$O(b^d)$	$O(b^d)$

O desempenho de estratégias de pesquisa informadas depende da qualidade da heurística utilizada. Uma boa heurística pode ser definida removendo certas definições ou obstáculos do problema, guardando custos pré-calculados de sub-problemas, ou aprendendo com a experiência do problema em questão.

ANÁLISE DO PROBLEMA 8-PUZZLE

Puzzle	Estratégia de pesquisa	Custo	Complexidade espacial	Complexidade temporal
A	BFS	14	2061	9147
	UCS	14	2497	12243
	DFS	56150	38270	181108
	IDS	14	13	21143
	Greedy BFS	18	35	116
	A*	14	55	226
B	BFS	26	24049	442332
	UCS	26	24182	459320
	DFS	48498	38270	331015
	IDS	26	23	3461312
	Greedy BFS	40	90	456
	A*	26	1066	5330

As estratégias de pesquisa informadas são mais convenientes e oferecem um desempenho melhor no caso do problema do 8-puzzle, sendo preferível o algoritmo A* devido a ter a capacidade de encontrar a solução óptima.

No caso das estratégias de pesquisa não-informada, pode-se considerar a possibilidade de usar o algoritmo IDS caso o custo entre passos seja idêntico, isto porque, tem a capacidade de encontrar a solução óptima com uma complexidade espacial linear.