QXD0037 - Inteligência Artificial

Trabalho 01 - Implementação
Busca em Profundidade e Largura - O Problema das 8 Rainhas
Prof. Samy Sá, 2019.2

PUBLICAÇÃO: 19/09/2019
PRAZO: 30/09/2019

O objetivo principal desta atividade é realizar a implementação de um problema de busca, bem como das técnicas de busca cega estudadas. Trabalharemos com o Problema das 8 Rainhas por simplicidade de implementação. Este trabalho pode ser feito em DUPLAS. Ao fim deste arquivo, estarão especificados como entregar o trabalho e aspectos relevantes à sua avaliação.

Os primeiros passos do seu trabalho envolvem escrever código para:
1. Instanciar estados do problema
2. Um Gerador de Estados (operadores + controle de redundância por caminho)
3. O Teste de Objetivo

Nesta parte, você pode utilizar qualquer uma das opções de modelagem do problema que discutimos. Cada uma delas tem vantagens e desvantagens que podem vir a ficar mais aparentes apenas durante a implementação dos operadores ou das buscas. Certifique-se de indicar com comentários no código como cada aspecto do problema de busca estaria sendo modelado.

Em seguida, você deve implementar as buscas de Profundidade e Largura sobre este problema, começando sempre pelo tabuleiro vazio. Como não variaremos o estado inicial, isso provavelmente fará com que todas as execuções dos seus códigos para cada busca sejam sempre idênticas, o que facilitará debugging e estudo da execução.

Como este trabalho visa exercitar as técnicas de busca cega, assuma que a resposta que procuramos é uma sequência de jogadas em que adicionaremos uma rainha por vez ao tabuleiro. Isso significa que procuramos um * caminho * do estado inicial até um estado de objetivo. Tendo isso em vista, a saída do seu código deve exibir – em cada iteração da estratégia de gerar e testar – o caminho completo da raiz até o estado que esteja sendo visitado e, em seguida, os novos estados que serão adicionados à árvore naquela iteração. A título de exemplo, o primeiro nó visitado será o estado inicial (a raiz da árvore), que consiste no tabuleiro vazio. Seu programa deve imprimir o caminho que consiste apenas deste estado e, em seguida, com alguma indicação de que esta seria sua vizinhança, imprimir os estados gerados por aplicação dos operadores que você utiliza. Em seguida, o primeiro vizinho gerado será visitado. Nesta iteração, seu programa deve exibir o caminho atual consistindo da raiz + o estado corrente e, como antes, imprimir os estados vizinhos deste que está sendo visitado. Manter o controle dos caminhos que visitamos é importante para o controle de redundância do Gerador de Estados.

Alguns detalhes fundamentais à implementação:
. Pode ser mais simples começar a implementação pela Busca em Profundidade, pois esta visitará apenas um caminho por vez; a utilização de uma pilha o conduzirá para a quase compleção do código.
. Para implementar a Busca em Largura, será necessário manter alguma forma de registro da árvore de busca à medida que esta for revelada. Não é necessário fazer uma implementação de estrutura dinâmica de árvore para tal; basta manter registro dos caminhos parciais da raiz até cada nó na fronteira da busca. A tendência é que você precise de muita memória. Recomenda-se fazer isto através de uma fila com os caminhos parciais cujos destinos não tenham sido visitados. O motivo é

que a implementação usando uma fila com estados não visitados será capaz de encontrar um estado objetivo, mas não conseguirá retornar o caminho da raiz até este objetivo

Entrega:
Seu trabalho deve ser submetido pelo SIPPA na entrada T1 como um arquivo compactado. A submissão deve conter, no mínimo:
. Um arquivo nomeado 'equipe.txt' com matrícula e nomes dos integrantes da equipe;
. Todos os códigos de implementação;
..Um executável, script ou arquivo principal de código a ser interpretado;
. Um arquivo 'help.txt' com instruções para executar seu código; incluindo detalhes sobre eventuais interpretadores e versões requeridos, requisitos de sistema operacional, e operações para iniciar a busca. O ideal é que seu código possa ser executado sem esforço. Caso você implemente outros controles, como um comando para pausar a busca ou avançá-la um passo por vez (podem ser úteis para debugging), indique também neste aquivo como utilizá-los.

Avaliação:
. Até 1 pts pela documentação dos códigos: comente extensivamente
    . Indique o que cada trecho se propõe a resolver;
    . Utilize comentários para detalhar a modelagem que você está implementando.
. Até 2 pts. se a saída dos algoritmos estiver exibida conforme solicitado e de forma compreensível;
. Até 3 pts. se a Busca em Profundidade estiver adequadamente implementada;
. Até 3 pts. se a Busca em Largura estiver adequadamente implementada;
. Até 1 pts por elegância dos códigos e soluções.
    . Código limpo e compreensível, eficiente, etc;

Créditos extras:
. Até 2 pts. Generalize o seu código para o Problema das *n* Rainhas.
    . Neste caso, seu código deve pedir como entrada o número *n* a ser utilizado. Recomenda-se começar pela implementação do problema com $n = 8$ (fixo) e generalizá-lo depois.