Projeto Buscas

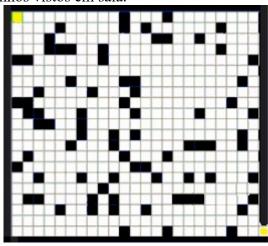
CMC- 15 Inteligência Artificial Prof. Paulo André L. de Castro Trabalho em Grupo de Três ou Quatro Alunos

1. Objetivo

Exercitar e fixar conhecimentos adquiridos sobre Resolução de Problemas através de Busca Heurística, Busca de Melhoria Iterativa (onde o destino é a solução, não o caminho) e sobre Problema de Satisfação de Restrições

2. Descrição do Trabalho

2.1. Do Ponto A ao Ponto B. Crie um agente capaz de encontrar o menor caminho entre os dois pontos amarelos: Início (A): (0,0) e Fim (B): (20,23), (linha,coluna). Os movimentos possíveis são Norte, Sul, Leste, Oeste e tem mesmo custo unitário. As células em preto são inacessíveis para o agente, ele não pode entrar em tais células, nem sair do tabuleiro (grid), qualquer tentativa neste sentido faz com que ele permaneça na mesma posição e tem custo unitário. Modele, implemente e resolva o problema utilizando um dos métodos de busca para encontrar caminhos vistos em sala.



- **2.2**. **Sudoku :** Crie um agente capaz de resolver o jogo chamado Sudoku utilizando algoritmos **de busca heurística, melhoria iterativa ou satisfação de restrição**. (Ver regras do jogo abaixo)
- 2.2.1. Modelagem: Explicite a sua modelagem para resolver o problema
- 2.2.1. Implementação: implemente os algoritmos para a resolução do problema, na linguagem de sua escolha. (Não utilize implementações prontas disponíveis na Internet, nem frameworks que implementem o métodos de resolução escolhido. Você pode usar apenas bibliotecas que facilitem a implementação das estruturas de dados necessárias)
- 2.2.3. Apresentar solução criada pelo programa para pelo menos quatro jogos distintos (ver notebook fornecido no classroom).
- 2.2.4. Contabilize o números de passos necessários (chamadas recursivas, passos de hill climbing, ou número de backtrackings, conforme a solução adotada), imprima o desempenho junta com a solução do problema.

2 5			3		9		1	#	2	5	8	7	3	6	9	4
1				4				#	6	1	9	8	2	4	3	5
4	7				2		8	#	4	3	7	9	1	5	2	6
					+		+	# +				+			+	
	5	2	_	0	1 4		- 1	#	3	9	5	2	7	1	4	8
		ļ.	9	8	1		- Ł	#	7	6	2	4	9	8	1	3
4				3	l			#	8	4	1	6	5	3	7	2
		3	· · ·				2	# +				+			+	
7		3	0			1	7 1	#	1	8	4	3	6	9	5	7
	٦				1 -		3	#	5	7	6	1	4	2	8	9
9	3				6		4	#	9	2	3	5	8	7	6	1

Exemplo de Jogo Sudoku - Início e Solução

Exemplo extraído do site abaixo, no qual você pode ver mais exemplos do Jogo: https://sudoku.com/pt

O jogo é composto de um tabuleiro (9x9). O objetivo é preencher todas células com número de 1 a 9, sem mudar as células já preenchidas inicialmente. Para solucionar o jogo, você deve satisfazer as seguintes condições:

- Cada coluna deve ter os números de 1 a 9
- Cada linha deve ter os números de 1 a 9
- Cada um dos nove quadrados que formam o tabuleiro devem conter os números de 1 a 9.

Seu código deve implementar um método solve que receba um lista de listas com os dados das células inicialmente preenchidas. Células iguais a zero, significam não preenchidas e previamente todas são zeradas.

3. Crie uma célula no seu notebook chamada Teste Final

A célula 'Teste Final' deve ter uma variável final_board definida como acima e ao executar resolver o Sudoku fornecido apresentado o desempenho (ver item 2.2.4) e a solução final.

4. Material a ser Entregue e Prazo

Material: Relatório e Código

Prazo de Entrega: 30/agosto/2024

Relatório do Projeto (arquivo em formato pdf até 4 páginas) com:

Capa: Título: Projeto Buscas e Nomes dos Membros da Equipe

- 1. Objetivo do Trabalho e Descrição da Implementação: Linguagem e IDE utilizados, comentários necessários para a execução do projeto.
- 2. **Resultados Obtidos**: Apresentar os resultados numéricos e/ou qualitativos pedidos na descrição de cada item da descrição do trabalho. Se necessário, apresentar hipóteses adicionais.

Item 2.1 [descrição da solução do item 2.1...]

Item 2.2. [descrição da solução do item 2.2...]

Conclusões: Comentários e sugestões sobre o trabalho (complexidade/facilidade, sugestões, etc.).

Código do Projeto: Notebook com Código-fonte do Sistema formato .ipynb (em Python ou Julia). Caso tenha utilizado outra linguagem (R ou Java, C ou C++). Enviar o código-fonte e binário para execução.

Siga a estrutura do relatório, para organizar o código no notebook

- 1. Equipe do Laboratório
- 2. 1. Problema Encontrar caminho

Códigos correspondentes

2.2. Problema do Sudoku

Códigos correspondentes

3. Teste Final Sudoku

Célula de Código correspondente

OBS: Não enviar link para github ou outros repositórios! O código deve ser enviado como descrito acima.

Bom Trabalho! Prof. Paulo André Castro pauloac@ita.br