BCC 325 - Inteligência Artificial - Prova 2

1. Considere as variáveis $A, B, C \in \{1, 2, 3\}$ e as restrições $A < B$ e $B < C$.

   (a) Quantas são atribuições totais possíveis.

   (b) Liste todas as atribuições totais possíveis.

   (c) Quantas dessas atribuições satisfazem todas as restrições?

   (d) Aponte quais atribuições não seriam verificadas se fosse utilizada a verificação de restrições em atribuições parciais do backtracking.

2. Responda as questões a seguir com base no algoritmo DFS solver abaixo.

Algorithm 1 DFS_SOLVER(V_s, C_s, context)

```
c_e ← {c ∈ C_s | c pode ser avaliado no context}
if context viola alguma restrição em c_e then
    return {}
else if V_s = {} then
    return {context}
else
    selecione variável var ∈ V_s
    sols ← {}
    for all val ∈ domínio(var) do
        sols ← sols ∪ DFS_SOLVER(V_s \ {var}, C_s \ c_e, {var = val} ∪ context)
    end for
    return sols
end if
```

   (a) Em que momento ocorre a verificação de restrições parciais?

   (b) Justifique por que isso melhora a eficiência do algoritmo em comparação ao *generate-and-test*.

3. Considere o problema com variáveis A, B, C e restrições:

$$A < B, B > C, C > A.$$

$$D(X) = \{1, 2, 3, 4\}, X \in \{A, B, C\}.$$

   (a) Modele este problema como um CSP formal (variáveis, domínios, restrições).

   (b) Represente o problema como uma rede de restrições.

4. Considere o CSP

$$A < B, B > C.$$

   com domínios iniciais

$$D(A) = D(B) = D(C) = \{1, 2, 3, 4\},$$

   e o algoritmo GAC abaixo:

```
def GAC(G,D):
    to_do = deepcopy(G)
    while to_do:
        a = to_do.pop(0)
        X = a['X'], c = a['c']
        Y = c['Scope'] - {X}
        Y = Y.pop()
        ND = set()
        for v in D[X]:
            for y in D[Y]:
                if c['Cs'](v,y):
                    ND.add(v)
                    break
        if ND != D[X]:
```

```
            D[X] = ND
            for edge in G:
                if X in edge['c']['Scope'] and edge['X'] != X:
                    to_do.append(edge)
    return D
```

Usando este algoritmo:

(a) Liste todos os arcos inicialmente inseridos na fila `to_do`.

(b) Execute manualmente o processo de revisão dos domínios, mostrando a lista *to_do* a cada iterção e a remoção de valores.

(c) Apresente os domínios finais $D'(A)$, $D'(B)$ e $D'(C)$ após a conclusão do GAC.

5. O GAC pode apresentar 3 tipos de saídas possíveis. Quais são elas e o que elas dizem sobre o problema?

6. Considere a implementação de *backtracking* que você desenvolveu para resolver o Sudoku no Problema Prático. A instância abaixo será utilizada como base para a simulação:

5	3	.	.	7	.	.	.	.
6	.	.	1	9	5	.	.	.
.	9	8	.	.	.	.	6	.
8	.	.	.	6	.	.	.	3
4	.	.	8	.	3	.	.	1
7	.	.	.	2	.	.	.	6
.	6	.	.	.	.	2	8	.
.	.	.	4	1	9	.	.	5
.	.	.	.	8	.	.	7	9

Simule manualmente as **três primeiras chamadas recursivas** do seu algoritmo de backtracking, descrevendo:

(a) Qual célula vazia é escolhida em cada chamada.

(b) Quais valores são testados para essa célula, indicando quais são imediatamente rejeitados por violarem regras do Sudoku (linha, coluna ou subgrade).

(c) Em que momento a recursão avança para a próxima célula vazia e em que momento ocorre retorno (backtrack) devido à falta de valores possíveis.

A sua resposta deve refletir o comportamento exato do algoritmo que você implementou, incluindo a ordem dos testes e a lógica usada para verificar consistência.

7. Considere sua implementação do problema dos *missionários e canibais*, utilizando a representação de estados

$$(M_E, C_E, B)$$

onde M_E e C_E são, respectivamente, o número de missionários e canibais na margem esquerda, e $B \in \{E, D\}$ indica a posição do barco.

Utilize o estado inicial padrão:

$$(3, 3, E)$$

e considere que a ação geradora de sucessores (`successors(state)`) segue exatamente as regras implementadas por você no Trabalho Prático.

Simule manualmente as **três primeiras expansões de estados** produzidas pelo seu algoritmo de **BFS** (Busca em Largura), descrevendo:

(a) O estado selecionado da fronteira em cada iteração.

(b) Todos os sucessores gerados a partir desse estado, indicando quais foram descartados por violarem as restrições do problema (por exemplo, canibais superando missionários em alguma margem).

(c) A ordem em que os estados válidos são inseridos na fronteira.

(d) O conteúdo da fronteira após cada uma das três expansões.

Sua resposta deve refletir exatamente as regras de geração de sucessores que você implementou, incluindo restrições, ordem de ações e política de inserção na fronteira adotada no seu código.