Relatório 2º Trabalho Prático

Bernardo Vitorino l48463, Daniel Barreiros l48452, Tomas Antunes l48511

28 de março de 2023

1 Quadrado mágico como um CSP

1.1 Representação do problema

1.1.1 Estados

A representação dos estados seguem a estrutura "e(LNT, LT)", em que LNT é a lista de variáveis não instânciadas e LT é a lista de variáveis instânciadas.

1.1.2 Variáveis

As variáveis são representadas na forma var((X, Y), D, V), onde X e Y são as coordenadas da varíavel, D o dominío e V o valor afetado.

1.1.3 Restrições

O predicado responsável por verificar as restrições é o predicado verifica_restrições, que verifica as coordenadas, as linhas, as colunas e as diagonais, de modo a verificar que todas as coordenadas têm um valor diferente e que a soma dos valores de cada linha, coluna e diagonal é igual.

1.1.4 Estado inicial

O estado inicial usa a estrutura de estados do problema.

```
estado_inicial(e([var((1,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_),
    var((2,1), [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15, ...,
    var((4,4), [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16], _)],[])).
```

1.1.5 Operador sucessor

```
sucessor(e([var(C, D, _)| R], E), e(R, [var(C, D, CX)| E])) :- member(CX, D).
```

1.2 Resolução com o algoritmo de backtracking

```
[var((4,4),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],2),var((3,4),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,4)
,[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((1,4),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((4,3),[1,2,3,4,
5,6,7,8,9,10,11,12,13,14,15,16],1),var((3,3),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,3),[1,2,3,4,5,6,7,8,9,
10,11,12,13,14,15,16],1),var((1,3),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((4,2),[1,2,3,4,5,6,7,8,9,10,11,12,1
3,14,15,16],1),var((3,2),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,2),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
],1),var((1,2),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((4,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((
3,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((1,1),[1,2,
3,4,5,6,7,8,9,10,11,12,13,14,15,16],1)]
```

Figura 1: Output com o algoritmo backtracking

1.3 Resolução com o algoritmo de forward checking

```
e([var((2,1),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_726),var((3,1),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_762),var((
4,1),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_798),var((1,2),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_834),var((2,2),[2,
3,4,5,6,7,8,9,10,11,12,13,14,15,16],_870),var((3,2),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_906),var((4,2),[2,3,4,5,6,
7,8,9,10,11,12,13,14,15,16],_942),var((1,3),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_978),var((2,3),[2,3,4,5,6,7,8,9,10
,11,12,13,14,15,16],_1014),var((3,3),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_1050),var((4,3),[2,3,4,5,6,7,8,9,10,11,12
,13,14,15,16],_1086),var((1,4),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_1122),var((2,4),[2,3,4,5,6,7,8,9,10,11,12,13,14
,15,16],_1158),var((3,4),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],_1194),var((4,4),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
],_1230)],[var((1,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1)])
```

Figura 2: Output com o algoritmo forward checking

1.4 Melhoria de complexidade

Para melhorarmos a complexidade espacial e temporal do algoritmo de pesquisa, decidimos utilizar os dois algoritmos, backtracking e forward checking, em conjunto.

```
[var((4,4),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],2),var((3,4),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,4)
,[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((1,4),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((4,3),[1,2,3,4,
5,6,7,8,9,10,11,12,13,14,15,16],1),var((3,3),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,3),[1,2,3,4,5,6,7,8,9,
10,11,12,13,14,15,16],1),var((1,3),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((4,2),[1,2,3,4,5,6,7,8,9,10,11,12,1
3,14,15,16],1),var((3,2),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,2),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
],1),var((1,2),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((4,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((
3,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((2,1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1),var((1,1),[1,2,
3,4,5,6,7,8,9,10,11,12,13,14,15,16],1)]
```

Figura 3: Output com o algoritmo backtracking e forward checking

1.5 Exemplos

1.5.1 2 por 2

```
[var((2,2),[3],3),var((1,2),[3,4],4),var((2,1),[2,3,4],2),var((1,1),[1,2,3,4],1)]
```

Figura 4: Resolução do cubo mágico 2 por 2 com o algoritmo forward checking e backtracking

1.5.2 3 por 3

```
[var((3,3),[8],8),var((2,3),[8,9],9),var((1,3),[7,8,9],7),var((3,2),[6,7,8,9],6),var((2,2),[5,
6,7,8,9],5),var((1,2),[4,5,6,7,8,9],4),var((3,1),[3,4,5,6,7,8,9],3),var((2,1),[2,3,4,5,6,7,8,9
],2),var((1,1),[1,2,3,4,5,6,7,8,9],1)]
```

Figura 5: Resolução do cubo mágico 3 por 3 com o algoritmo forward checking e backtracking

1.5.3 4 por 4

```
[var((4,4),[15],15),var((3,4),[15,16],16),var((2,4),[14,15,16],14),var((1,4),[13,14,15,16],13)
,var((4,3),[12,13,14,15,16],12),var((3,3),[11,12,13,14,15,16],11),var((2,3),[10,11,12,13,14,15
,16],10),var((1,3),[9,10,11,12,13,14,15,16],9),var((4,2),[8,9,10,11,12,13,14,15,16],8),var((3,
2),[7,8,9,10,11,12,13,14,15,16],7),var((2,2),[6,7,8,9,10,11,12,13,14,15,16],6),var((1,2),[5,6,
7,8,9,10,11,12,13,14,15,16],5),var((4,1),[4,5,6,7,8,9,10,11,12,13,14,15,16],4),var((3,1),[3,4,
5,6,7,8,9,10,11,12,13,14,15,16],3),var((2,1),[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],2),var((1,
1),[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],1)]
```

Figura 6: Resolução do cubo mágico 4 por 4 com o algoritmo forward checking e backtracking

2 Sudoku como um CSP

2.1 Representação do problema

2.1.1 Estados

A representação dos estados seguem a estrutura ”e(LNT, LT)”, em que LNT é a lista de variáveis não instânciadas e LT é a lista de variáveis instânciadas.

2.1.2 Variáveis

As variáveis são representadas na forma var((X, Y), D, V), onde X e Y são as coordenadas da varíavel, D o dominío e V o valor afetado.

2.1.3 Restrições

O predicado responsável por verificar as restrições é o predicado ve_restrições, que verifica as linhas as colunas e os quadrantes, de modo a verificar ques todos os elementos destes três grupos são diferentes.

2.1.4 Estado inicial

O estado inicial usa a estrutura de estados do problema.

```
estado_inicial(e([var((1, 1), [1,2,3,4,5,6,7,8,9], _), ...,
    var((9, 8), [1,2,3,4,5,6,7,8,9], _),
    var((9, 9), [1,2,3,4,5,6,7,8,9], _)],
    [var((1, 2), [1,2,3,4,5,6,7,8,9], 1), ...,
    var((9, 3), [1,2,3,4,5,6,7,8,9], 6),
    var((9, 6), [1,2,3,4,5,6,7,8,9], 3)])).
```

2.1.5 Operador sucessor

```
sucessor(e([var(N,D,_)|R],E),e(R,[var(N,D,V)|E])):- member(V,D).
```

2.2 Resolução com o algoritmo backtracking

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

Figura 7: Output com o algoritmo backtracking

2.3 Resolução com o algoritmo forward checking

```
Fatal Error: global stack overflow (size: 32768 Kb, reached: 32765 Kb, environment variable used: GLOBALSZ)
```

Figura 8: Output com o algoritmo forward checking

Isto acontece pois é excedida a memória disponibilizada para a execução do PROLOG.