Report

Introduction to Artificial Intelligence

Task 2: Constraint Satisfaction Problem

Grzegorz Tomaka, 245669

Sudoku CSP:

Formulation for sukoku board 9X9 where columns are numereted from 1 to 9 and rows are marked by {A, B,…,I}

Possible **Variables**: {A1,…..,A9,B1,……..,B2, ….., I1,…..I9} |V| = 81

Domains: {1,2,3, ….., 9}

Constraints: 27 constraints

- Alldif(A1 , A2 , A3 , A4 , A5 , A6 , A7 , A8 , A9)
- ……..
- Alldif(A1, B1, C1, D1, E1, F1, G1, H1, I1)
- ……..
- Alldif(A1, A2, A3, B1, B2, B3, C1, C2, C3)
- ……

Analysis:

Sudoku id: 11 , difficulty level : 2

11 | 2.0 | .....2.59..8..46..79...6.1..274..86...3.7154....8.....6.....38.....1....

```
SOLUTION
[3, 4, 6, 1, 8, 2, 7, 5, 9]
[1, 5, 8, 7, 9, 4, 6, 2, 3]
[7, 9, 2, 3, 5, 6, 4, 1, 8]
[5, 2, 7, 4, 3, 9, 8, 6, 1]
[9, 8, 3, 6, 7, 1, 5, 4, 2]
[4, 6, 1, 8, 2, 5, 9, 3, 7]
[6, 3, 9, 2, 4, 8, 1, 7, 5]
[2, 1, 5, 9, 6, 7, 3, 8, 4]
[8, 7, 4, 5, 1, 3, 2, 9, 6]
```

	Expanded nodes	Time
Backtracking	1679	0.5422275999999999
Backtracking with Forward Checking:	469	0.3088433999999999
Backtracking with Minimum Remainin Value Heuristics:	296	0.19357239999999987
Backtracking with MRV and Forward Checking:	276	0.2362613

Sudoku id: 21 level 4

21 | 4.0 | 32.……..8…..9..569.3.2...4.69..1...7....2.35..4……..5….1….74...7..8.3

```
SOLUTION
[3, 2, 9, 1, 5, 4, 7, 6, 8]
[6, 4, 8, 2, 3, 7, 1, 5, 9]
[7, 1, 5, 6, 9, 8, 3, 4, 2]
[5, 8, 7, 4, 2, 6, 9, 3, 1]
[1, 9, 4, 8, 7, 3, 6, 2, 5]
[2, 6, 3, 5, 1, 9, 4, 8, 7]
[4, 7, 2, 3, 8, 1, 5, 9, 6]
[8, 3, 1, 9, 6, 5, 2, 7, 4]
[9, 5, 6, 7, 4, 2, 8, 1, 3]
```

	Expanded nodes	Time
Backtracking	20101	5.1649325
Backtracking with Forward Checking:	2679	2.1974981000000007
Backtracking with Minimum Remainin Value Heuristics:	392	0.13030700000000017
Backtracking with MRV and Forward Checking:	334	0.27728670000000033

Sudoku id: 36 level 7

36 | 7.0 | 7.3.8126..2...75……6…..87.2..1.4..7...2.1….6…..9.4..7….2..8……..4.

```
SOLUTION
[7, 9, 3, 5, 8, 1, 2, 6, 4]
[6, 2, 4, 9, 3, 7, 5, 8, 1]
[8, 1, 5, 4, 6, 2, 7, 9, 3]
[9, 8, 7, 3, 2, 4, 6, 1, 5]
[4, 3, 6, 7, 1, 5, 9, 2, 8]
[1, 5, 2, 8, 9, 6, 4, 3, 7]
[5, 6, 9, 1, 4, 8, 3, 7, 2]
[3, 4, 1, 2, 7, 9, 8, 5, 6]
[2, 7, 8, 6, 5, 3, 1, 4, 9]
```

	Expanded nodes	Time
Backtracking	14622	4.8633044
Backtracking with Forward Checking:	4405	5.0490747
Backtracking with Minimum Remainin Value Heuristics:	467	0.3032497999999997
Backtracking with MRV and Forward Checking:	435	0.48135940000000055

Sudoku id: 43 level 8

43 | 8.0 | ……..8.4.6..3..6…2.4.9.95…..1….3.2….6.51.8..3…7…9..4…….2.63…

```
SOLUTION
[1, 5, 7, 3, 4, 9, 6, 2, 8]
[2, 4, 9, 6, 7, 8, 3, 1, 5]
[6, 8, 3, 1, 2, 5, 4, 7, 9]
[3, 9, 5, 2, 8, 6, 7, 4, 1]
[8, 7, 1, 9, 3, 4, 2, 5, 6]
[4, 2, 6, 7, 5, 1, 9, 8, 3]
[5, 3, 4, 8, 9, 7, 1, 6, 2]
[9, 6, 8, 4, 1, 2, 5, 3, 7]
[7, 1, 2, 5, 6, 3, 8, 9, 4]
```

	Expanded nodes	Time
Backtracking	24339	4.7963408
Backtracking with Forward Checking:	3943	1.7435356999999998
Backtracking with Minimum Remainin Value Heuristics:	699	0.1370331999999994
Backtracking with MRV and Forward Checking:	646	0.32370060000000045

Sudoku: level 9 – unresolved

Conclusions: Results show that backtracking method are able to solve CSP sudoku problem in reasonable time. However, using additionally Forwad Checking we can significantly reduce expanded nodes and sometimes also execution time (in sudoku id 36 time with Forward Checking time was longer) . The next attempt to reduce expanded nodes was using Minimum Remaining Value Heuristics and it turned out to be more effective. The bactracking with MRV could reduce expanded nodes even 30x times (sudoku id 36). By combining these two previous optimization method I obtained the best result in terms of expanded-nodes for each sukoku. I think the backtracking with forward checking and MRV is the best method from analyzed by me because it reduces expanded nodes the most what seems to be more crucial factor than execution time.

Fill-in puzzle

CSP formulation explained using the example

			#

Domain = {boat, art, need, ban, ore, ate }

In my idea of a solution I am looking for variables separately for horizontal variables and vertical ones

H_var

[0, 0]	**[0, 1]**	**[0, 2]**	**[0, 3]**
[1 ,0]	**[1, 1]**	**[1, 2]**	**#**
[2, 0]	**[2, 1]**	**[2, 2]**	**[2, 3]**

Horizontal variables: (for each cell with the second value equal 0)

H0, H1, H2

Ex. [0, 1]
The first value ('0') is the number of variable
The second value ('1') is responsible for indication of the letter number in variable.

V_var

[0, 0]	**[1, 0]**	**[2, 0]**	**#**
[0 ,1]	**[1, 1]**	**[2, 1]**	**#**
[0, 2]	**[1, 1]**	**[2, 2]**	**#**

Vertical variables: (for each cell with thes second value equal 0)

V0, V1, V2

Based on these tables I create domains and constraints

Domains (D):

D(H0) = D(H2) = All words of length 4 from Domain = {boat, need}
D(H1) = D(V0) = D(V1) = D(V2) = All words of length 3 from Domain = {art., ban, ore, ate }

Constraints:

- **C1 H0[0] (first letter in horizontal variable nr 0) = V0[0] (first letter in vertical variable nr 0)**
- **C2 H0[1] = V1[0]**
- **C3 H0[2] = V2[0]**
- **C4 H1[0] = V0[1]**
- **C5 H1[1] = V1[1]**
- **C6 H1[2] = V2[1]**
- **C7 H2[0] = V0[2]**
- **C8 H2[1] = V1[2]**
- **C9 H2[2] = V2[2]**

Puzzle 0

Backtracking

```
SOLUTION
['b', 'o', 'a', 't']
['a', 'r', 't', '#']
['n', 'e', 'e', 'd']
Expanded nodes:
7
Time:
0.00017600000000000254
```

Backtracking with forward checking

```
SOLUTION
['b', 'o', 'a', 't']
['a', 'r', 't', '#']
['n', 'e', 'e', 'd']
Expanded nodes:
6
Time:
0.00018800000000000067
```

Puzzle 1

Backtracking

```
SOLUTION
['#', '#', 'D', 'A', 'G', '#', '#']
['#', '#', 'A', 'R', 'I', 'D', '#']
['E', 'D', 'I', 'T', '#', 'O', 'R']
['V', 'E', 'S', 'I', 'C', 'L', 'E']
['O', 'N', '#', 'C', 'L', 'E', 'F']
['#', 'S', 'I', 'L', 'O', '#', '#']
['#', '#', 'O', 'E', 'D', '#', '#']
Expanded nodes:
25
Time:
0.0014295000000000002
```

Backtracking with forward checking

```
SOLUTION
['#', '#', 'D', 'A', 'G', '#', '#']
['#', '#', 'A', 'R', 'I', 'D', '#']
['E', 'D', 'I', 'T', '#', 'O', 'R']
['V', 'E', 'S', 'I', 'C', 'L', 'E']
['O', 'N', '#', 'C', 'L', 'E', 'F']
['#', 'S', 'I', 'L', 'O', '#', '#']
['#', '#', 'O', 'E', 'D', '#', '#']
Expanded nodes:
18
Time:
0.001160399999999999
```

Puzzle 2

Backtracking

```
SOLUTION
['Z', 'K', 'O', 'Q', 'S', '#', 'Z', 'G', 'N']
['Q', 'E', 'D', 'T', '#', 'Z', 'O', 'S', 'T']
['O', 'G', '#', '#', '#', 'W', 'G', 'K', 'T']
['#', '#', 'Z', 'G', 'L', 'L', 'T', 'R', '#']
['Q', 'Z', 'Z', 'T', 'F', 'Z', 'O', 'C', 'T']
['#', 'D', 'O', 'F', 'O', 'F', 'U', '#', '#']
['L', 'T', 'T', 'F', '#', '#', '#', 'G', 'A']
['R', 'G', 'U', 'L', '#', 'H', 'O', 'T', 'L']
['Z', 'G', 'H', '#', 'L', 'V', 'O', 'F', 'U']
Expanded nodes:
215
Time:
0.0371152
```

Backtracking with forward checking

```
SOLUTION
['Z', 'K', 'O', 'Q', 'S', '#', 'Z', 'G', 'N']
['Q', 'E', 'D', 'T', '#', 'Z', 'O', 'S', 'T']
['O', 'G', '#', '#', '#', 'W', 'G', 'K', 'T']
['#', '#', 'Z', 'G', 'L', 'L', 'T', 'R', '#']
['Q', 'Z', 'Z', 'T', 'F', 'Z', 'O', 'C', 'T']
['#', 'D', 'O', 'F', 'O', 'F', 'U', '#', '#']
['L', 'T', 'T', 'F', '#', '#', '#', 'G', 'A']
['R', 'G', 'U', 'L', '#', 'H', 'O', 'T', 'L']
['Z', 'G', 'H', '#', 'L', 'V', 'O', 'F', 'U']
Expanded nodes:
32
Time:
0.004469999999999998
```

Puzzle 3

Backtracking

```
SOLUTION
['V', 'E', 'R', 'N', 'E', '#', 'E', 'R', 'N', 'E', '#', 'F', 'I', 'G', 'S']
['O', 'R', 'I', 'O', 'N', '#', 'R', 'O', 'A', 'D', '#', 'A', 'L', 'E', 'C']
['L', 'I', 'P', 'I', 'D', '#', 'S', 'O', 'N', 'G', '#', 'R', 'E', 'N', 'O']
['T', 'E', 'E', 'S', '#', 'P', 'E', 'T', '#', 'I', 'N', 'T', 'U', 'I', 'T']
['#', '#', '#', 'E', 'R', 'A', 'S', '#', 'O', 'N', 'E', '#', 'M', 'E', 'S']
['T', 'E', 'U', 'T', 'O', 'N', '#', 'I', 'R', 'E', 'S', '#', '#', '#', '#']
['A', 'U', 'N', 'T', 'Y', '#', 'A', 'V', 'A', 'S', 'T', '#', 'B', 'O', 'X']
['B', 'R', 'I', 'E', '#', 'K', 'N', 'O', 'T', 'S', '#', 'C', 'O', 'I', 'R']
['S', 'O', 'T', '#', 'S', 'N', 'O', 'R', 'E', '#', 'C', 'R', 'E', 'N', 'A']
['#', '#', '#', '#', 'T', 'I', 'D', 'Y', '#', 'Q', 'U', 'I', 'R', 'K', 'Y']
['D', 'O', 'G', '#', 'A', 'T', 'E', '#', 'V', 'E', 'R', 'T', '#', '#', '#']
['E', 'X', 'H', 'O', 'R', 'T', '#', 'K', 'I', 'D', '#', 'I', 'N', 'S', 'T']
['L', 'I', 'A', 'R', '#', 'I', 'R', 'A', 'N', '#', 'S', 'C', 'O', 'N', 'E']
['E', 'D', 'N', 'A', '#', 'N', 'I', 'L', 'E', '#', 'O', 'A', 'S', 'I', 'S']
['S', 'E', 'A', 'L', '#', 'G', 'O', 'E', 'S', '#', 'S', 'L', 'E', 'P', 'T']
Expanded nodes:
1085062
Time:
1763.64185
```

Backtracking with forward checking

```
SOLUTION
['V', 'E', 'R', 'N', 'E', '#', 'E', 'R', 'N', 'E', '#', 'F', 'I', 'G', 'S']
['O', 'R', 'I', 'O', 'N', '#', 'R', 'O', 'A', 'D', '#', 'A', 'L', 'E', 'C']
['L', 'I', 'P', 'I', 'D', '#', 'S', 'O', 'N', 'G', '#', 'R', 'E', 'N', 'O']
['T', 'E', 'E', 'S', '#', 'P', 'E', 'T', '#', 'I', 'N', 'T', 'U', 'I', 'T']
['#', '#', '#', 'E', 'R', 'A', 'S', '#', 'O', 'N', 'E', '#', 'M', 'E', 'S']
['T', 'E', 'U', 'T', 'O', 'N', '#', 'I', 'R', 'E', 'S', '#', '#', '#', '#']
['A', 'U', 'N', 'T', 'Y', '#', 'A', 'V', 'A', 'S', 'T', '#', 'B', 'O', 'X']
['B', 'R', 'I', 'E', '#', 'K', 'N', 'O', 'T', 'S', '#', 'C', 'O', 'I', 'R']
['S', 'O', 'T', '#', 'S', 'N', 'O', 'R', 'E', '#', 'C', 'R', 'E', 'N', 'A']
['#', '#', '#', '#', 'T', 'I', 'D', 'Y', '#', 'Q', 'U', 'I', 'R', 'K', 'Y']
['D', 'O', 'G', '#', 'A', 'T', 'E', '#', 'V', 'E', 'R', 'T', '#', '#', '#']
['E', 'X', 'H', 'O', 'R', 'T', '#', 'K', 'I', 'D', '#', 'I', 'N', 'S', 'T']
['L', 'I', 'A', 'R', '#', 'I', 'R', 'A', 'N', '#', 'S', 'C', 'O', 'N', 'E']
['E', 'D', 'N', 'A', '#', 'N', 'I', 'L', 'E', '#', 'O', 'A', 'S', 'I', 'S']
['S', 'E', 'A', 'L', '#', 'G', 'O', 'E', 'S', '#', 'S', 'L', 'E', 'P', 'T']
Expanded nodes:
89
Time:
0.1167544
```

Puzzle 4

Backtracking

```
SOLUTION
['B', 'H', 'N', 'N', 'C', 'N', '#', 'P', 'P', 'A', 'G', '#', 'A', 'B', 'D', 'O', 'H', 'N', 'J', 'O', 'A', 'M', 'D', '#', 'A', 'B', 'C', 'I', '#', 'L', 'J', 'I']
['H', 'M', 'L', 'K', 'L', 'M', 'L', 'D', 'I', 'K', 'E', '#', 'P', 'O', 'M', 'H', 'M', 'K', 'J', 'C', 'O', 'F', 'E', 'G', 'C', 'A', 'A', 'A', '#', 'F', 'J', 'L']
['F', 'B', 'J', 'G', 'D', 'C', 'H', 'L', 'O', 'F', 'L', 'G', 'O', 'H', 'P', 'E', 'N', 'D', '#', 'F', 'B', 'D', 'A', 'D', 'H', 'O', 'D', 'P', 'L', 'H', 'I', 'B']
['E', 'L', '#', 'M', 'I', 'L', 'P', 'N', 'B', 'O', 'L', 'O', 'N', 'F', 'F', 'M', 'D', 'I', 'O', 'G', 'C', 'P', 'K', '#', 'O', 'B', 'D', 'C', 'J', 'A', 'I', 'E']
['H', 'O', 'P', 'H', 'P', 'I', 'F', '#', 'A', 'P', 'L', 'O', 'F', 'D', 'P', 'D', 'I', 'E', 'E', 'J', 'E', 'L', 'K', 'B', '#', 'E', 'I', 'H', 'C', 'L', 'N', 'O']
['L', 'I', 'A', 'H', 'L', 'J', 'M', 'K', 'N', 'N', 'N', 'N', 'C', 'J', '#', 'P', 'F', 'G', 'M', 'I', 'H', 'N', 'G', 'A', 'A', 'A', 'D', 'L', 'B', 'E', 'F', 'N']
['J', 'G', '#', 'H', 'K', 'B', 'G', 'D', 'D', 'B', 'M', 'O', 'C', 'I', 'A', 'L', 'O', 'M', 'A', '#', 'G', 'B', 'K', 'I', 'D', 'P', 'M', 'N', 'C', 'J', 'K', 'D']
['K', 'N', 'M', 'J', 'B', 'I', 'J', 'A', 'M', '#', 'I', 'N', 'G', 'B', 'N', 'O', 'D', 'N', 'J', 'P', 'I', 'C', 'N', 'M', 'I', 'L', '#', 'G', 'G', 'M', 'M', 'I']
['M', 'K', 'B', 'C', 'C', 'N', '#', 'F', 'L', 'L', 'M', 'D', 'H', 'M', '#', 'D', 'P', 'C', 'J', 'E', 'N', 'M', 'D', 'B', 'M', 'H', 'M', 'B', 'N', 'F', '#', '#']
['F', 'L', 'A', 'G', 'O', 'N', 'P', 'N', 'H', 'L', 'K', 'L', 'B', 'P', 'B', '#', 'K', 'L', 'E', 'I', 'F', '#', 'M', 'E', 'E', 'D', 'N', 'O', 'H', 'E', 'L', 'O']
['N', 'P', 'H', 'A', 'P', '#', 'P', 'C', 'N', 'D', 'J', 'D', 'L', 'L', 'N', 'E', 'O', 'P', 'P', 'L', 'C', 'E', 'D', 'L', 'L', 'C', '#', 'E', 'L', 'E', 'P', 'I']
['D', 'P', 'E', 'G', 'B', 'P', 'K', 'E', 'F', 'E', '#', 'M', 'L', 'J', 'F', 'L', 'L', 'O', 'N', 'L', 'L', 'C', 'O', 'N', '#', 'G', 'E', 'P', 'E', 'N', 'O', 'K']
['#', '#', 'E', 'P', 'D', 'B', 'P', 'A', 'P', 'K', 'B', 'F', 'F', 'I', 'H', 'F', 'P', '#', 'B', 'G', 'M', 'D', 'F', 'D', 'A', 'F', 'E', 'L', 'M', 'M', 'K', 'B']
['E', 'L', 'M', 'I', 'D', '#', 'J', 'F', 'O', 'J', 'M', 'L', 'I', 'L', 'D', 'M', 'M', 'M', 'L', 'H', 'F', 'O', '#', 'O', 'H', 'N', 'G', 'O', 'G', 'L', 'O', 'E']
['M', 'A', 'A', 'M', 'G', 'P', 'O', 'D', 'B', 'K', 'K', 'L', '#', 'C', 'C', 'K', 'D', 'J', 'E', 'N', 'M', 'E', 'A', 'L', 'E', 'G', 'G', 'E', 'N', '#', 'E', 'G']
['N', 'N', 'H', 'B', 'K', 'K', 'M', 'H', 'F', 'A', 'G', 'K', 'I', 'H', 'J', 'H', 'P', '#', 'D', 'P', 'M', 'H', 'A', 'M', 'G', 'I', 'P', 'K', 'A', 'F', 'F', 'L']
['A', 'N', 'A', 'D', 'M', 'O', 'F', 'N', 'C', 'G', 'C', 'M', 'H', '#', 'F', 'H', 'B', 'G', 'I', 'P', 'C', 'F', 'L', 'B', '#', 'B', 'B', 'A', 'C', 'K', 'A', 'F']
['F', 'C', 'O', 'M', 'E', 'B', 'C', 'I', '#', 'B', 'N', 'K', 'A', '#', 'C', 'I', 'M', 'O', 'P', 'L', 'B', 'A', 'M', 'M', 'H', 'F', 'C', 'N', 'K', '#', 'B', 'L']
['H', 'O', 'L', '#', 'P', 'N', 'O', 'L', 'O', 'C', 'H', 'K', 'M', 'D', '#', 'D', 'O', 'A', 'H', 'D', 'I', 'F', 'D', 'F', 'I', 'N', 'C', 'C', 'I', 'F', 'O', 'A']
['C', 'C', 'H', '#', 'N', 'L', 'G', 'K', 'E', 'H', 'C', 'E', 'K', 'C', '#', 'M', 'M', 'L', 'H', 'K', 'C', 'P', 'M', 'D', 'J', 'H', 'M', 'N', 'K', 'A', 'M', 'M']
Expanded nodes:
1617
Time:
1.1005984
```

Expanded nodes:
1617

Time:
1.1005984

Backtracking with forward checking

```
SOLUTION
['B', 'H', 'N', 'N', 'C', 'N', '#', 'P', 'P', 'A', 'G', '#', 'A', 'B', 'D', 'O', 'H', 'N', 'J', 'O', 'A', 'M', 'D', '#', 'A', 'B', 'C', 'I', '#', 'L', 'J', 'I']
['H', 'M', 'L', 'K', 'L', 'M', 'L', 'D', 'I', 'K', 'E', '#', 'P', 'O', 'M', 'H', 'M', 'K', 'J', 'C', 'O', 'F', 'E', 'G', 'C', 'A', 'A', 'A', '#', 'F', 'J', 'L']
['F', 'B', 'J', 'G', 'D', 'C', 'H', 'L', 'O', 'F', 'L', 'G', 'O', 'H', 'P', 'E', 'N', 'D', '#', 'F', 'B', 'D', 'A', 'D', 'H', 'O', 'D', 'P', 'L', 'H', 'I', 'B']
['E', 'L', '#', 'M', 'I', 'L', 'P', 'N', 'B', 'O', 'L', 'O', 'N', 'F', 'F', 'M', 'D', 'I', 'O', 'G', 'C', 'P', 'K', '#', 'O', 'B', 'D', 'C', 'J', 'A', 'I', 'E']
['H', 'O', 'P', 'H', 'P', 'I', 'F', '#', 'A', 'P', 'L', 'O', 'F', 'D', 'P', 'D', 'I', 'E', 'E', 'J', 'E', 'L', 'K', 'B', '#', 'E', 'I', 'H', 'C', 'L', 'N', 'O']
['L', 'I', 'A', 'H', 'L', 'J', 'M', 'K', 'N', 'N', 'N', 'N', 'C', 'J', '#', 'P', 'F', 'G', 'M', 'I', 'H', 'N', 'G', 'A', 'A', 'A', 'D', 'L', 'B', 'E', 'F', 'N']
['J', 'G', '#', 'H', 'K', 'B', 'G', 'D', 'D', 'B', 'M', 'O', 'C', 'I', 'A', 'L', 'O', 'M', 'A', '#', 'G', 'B', 'K', 'I', 'D', 'P', 'M', 'N', 'C', 'J', 'K', 'D']
['K', 'N', 'M', 'J', 'B', 'I', 'J', 'A', 'M', '#', 'I', 'N', 'G', 'B', 'N', 'O', 'D', 'N', 'J', 'P', 'I', 'C', 'N', 'M', 'I', 'L', '#', 'G', 'G', 'M', 'M', 'I']
['M', 'K', 'B', 'C', 'C', 'N', '#', 'F', 'L', 'L', 'M', 'D', 'H', 'M', '#', 'D', 'P', 'C', 'J', 'E', 'N', 'M', 'D', 'B', 'M', 'H', 'M', 'B', 'N', 'F', '#', '#']
['F', 'L', 'A', 'G', 'O', 'N', 'P', 'N', 'H', 'L', 'K', 'L', 'B', 'P', 'B', '#', 'K', 'L', 'E', 'I', 'F', '#', 'M', 'E', 'E', 'D', 'N', 'O', 'H', 'E', 'L', 'O']
['N', 'P', 'H', 'A', 'P', '#', 'P', 'C', 'N', 'D', 'J', 'D', 'L', 'L', 'N', 'E', 'O', 'P', 'P', 'L', 'C', 'E', 'D', 'L', 'L', 'C', '#', 'E', 'L', 'E', 'P', 'I']
['D', 'P', 'E', 'G', 'B', 'P', 'K', 'E', 'F', 'E', '#', 'M', 'L', 'J', 'F', 'L', 'L', 'O', 'N', 'L', 'L', 'C', 'O', 'N', '#', 'G', 'E', 'P', 'E', 'N', 'O', 'K']
['#', '#', 'E', 'P', 'D', 'B', 'P', 'A', 'P', 'K', 'B', 'F', 'F', 'I', 'H', 'F', 'P', '#', 'B', 'G', 'M', 'D', 'F', 'D', 'A', 'F', 'E', 'L', 'M', 'M', 'K', 'B']
['E', 'L', 'M', 'I', 'D', '#', 'J', 'F', 'O', 'J', 'M', 'L', 'I', 'L', 'D', 'M', 'M', 'M', 'L', 'H', 'F', 'O', '#', 'O', 'H', 'N', 'G', 'O', 'G', 'L', 'O', 'E']
['M', 'A', 'A', 'M', 'G', 'P', 'O', 'D', 'B', 'K', 'K', 'L', '#', 'C', 'C', 'K', 'D', 'J', 'E', 'N', 'M', 'E', 'A', 'L', 'E', 'G', 'G', 'E', 'N', '#', 'E', 'G']
['N', 'N', 'H', 'B', 'K', 'K', 'M', 'H', 'F', 'A', 'G', 'K', 'I', 'H', 'J', 'H', 'P', '#', 'D', 'P', 'M', 'H', 'A', 'M', 'G', 'I', 'P', 'K', 'A', 'F', 'F', 'L']
['A', 'N', 'A', 'D', 'M', 'O', 'F', 'N', 'C', 'G', 'C', 'M', 'H', '#', 'F', 'H', 'B', 'G', 'I', 'P', 'C', 'F', 'L', 'B', '#', 'B', 'B', 'A', 'C', 'K', 'A', 'F']
['F', 'C', 'O', 'M', 'E', 'B', 'C', 'I', '#', 'B', 'N', 'K', 'A', '#', 'C', 'I', 'M', 'O', 'P', 'L', 'B', 'A', 'M', 'M', 'H', 'F', 'C', 'N', 'K', '#', 'B', 'L']
['H', 'O', 'L', '#', 'P', 'N', 'O', 'L', 'O', 'C', 'H', 'K', 'M', 'D', '#', 'D', 'O', 'A', 'H', 'D', 'I', 'F', 'D', 'F', 'I', 'N', 'C', 'C', 'I', 'F', 'O', 'A']
['C', 'C', 'H', '#', 'N', 'L', 'G', 'K', 'E', 'H', 'C', 'E', 'K', 'C', '#', 'M', 'M', 'L', 'H', 'K', 'C', 'P', 'M', 'D', 'J', 'H', 'M', 'N', 'K', 'A', 'M', 'M']
Expanded nodes:
123
Time:
0.0483049
```

Expanded nodes: 123
Time: 0.0483049

Conclusions:

Similar to sudoku, the algorithm using backtracking solved all puzzles and the use of forward checking allowed to reduce expanded nodes. The most interesting and very suprising result I obtained for puzzle nr 3 because pure backtracking method turned out to be very inefficient. It expanded 1 085 062 nodes and was highly time-consuming (about half an hour). Using Forward Checking the algorithm expanded only 89 nodes.

Sources:

https://medium.com/my-udacity-ai-nanodegree-notes/solving-sudoku-think-constraint-satisfaction-problem-75763f0742c9

http://zpcir.ict.pwr.wroc.pl/~witold/aiarr/2007_projekty/sudoku/https://courses.edx.org/courses/course-v1:ColumbiaX+CSMM.101x+1T2017/courseware/41cab42cb155498eb00d09102b10a578/9a4c82a0add14d20bbbce9642766d37a/?child=last

https://levelup.gitconnected.com/csp-algorithm-vs-backtracking-sudoku-304a242f96d0

https://web.stanford.edu/class/cs227/Lectures/lec14.pdf