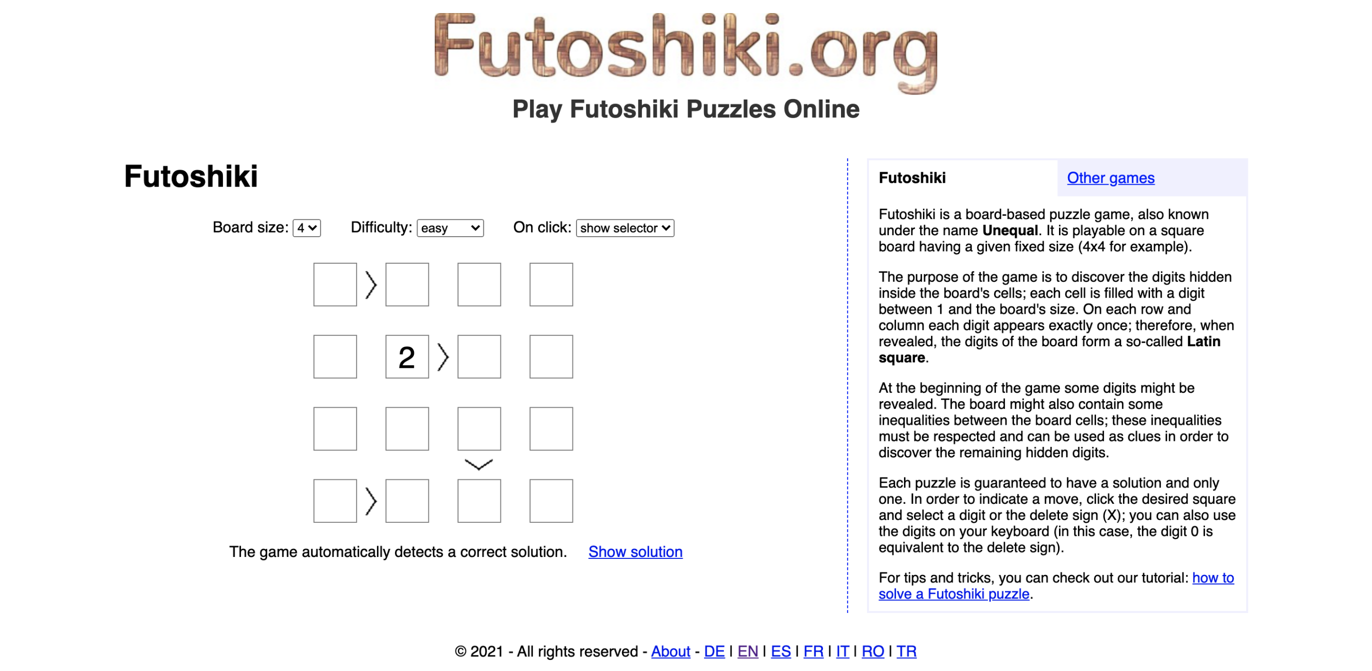
PG47030 - António Jorge Nande Rodrigues - Mestrado em Engenharia Informática

Em baixo segue a foto do puzzle resolvido neste SMT Solver.

Posteriormente será apresentada a foto da solução.



Futoshiki

Board size: 4 Difficulty: easy On click: show selector

The game automatically detects a correct solution. Show solution

Futoshiki Other games

Futoshiki is a board-based puzzle game, also known under the name **Unequal**. It is playable on a square board having a given fixed size (4x4 for example).

The purpose of the game is to discover the digits hidden inside the board's cells; each cell is filled with a digit between 1 and the board's size. On each row and column each digit appears exactly once; therefore, when revealed, the digits of the board form a so-called **Latin square**.

At the beginning of the game some digits might be revealed. The board might also contain some inequalities between the board cells; these inequalities must be respected and can be used as clues in order to discover the remaining hidden digits.

Each puzzle is guaranteed to have a solution and only one. In order to indicate a move, click the desired square and select a digit or the delete sign (X); you can also use the digits on your keyboard (in this case, the digit 0 is equivalent to the delete sign).

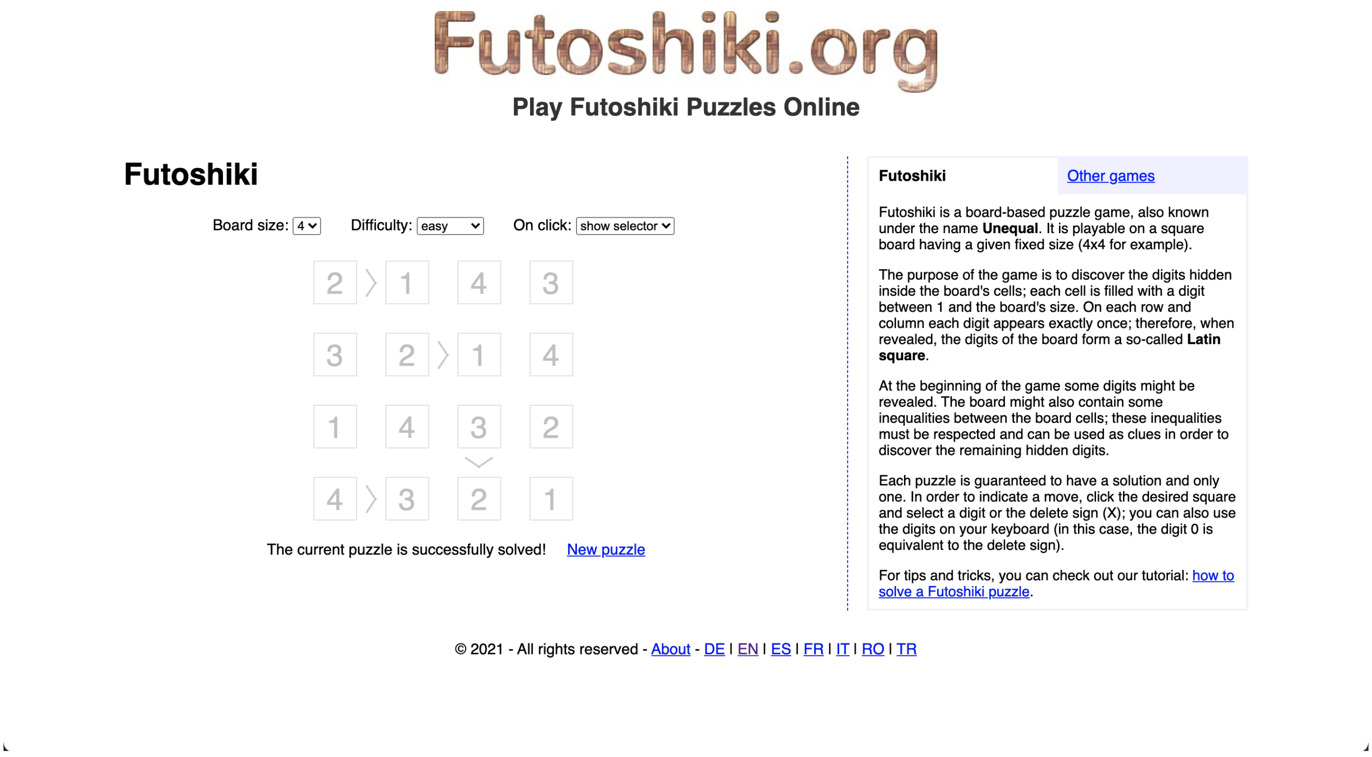
For tips and tricks, you can check out our tutorial: how to solve a Futoshiki puzzle.

© 2021 - All rights reserved - About - DE | EN | ES | FR | IT | RO | TR

```
!pip install z3-solver
from z3 import *
import re
import numpy as np

def printTabuleiro(tabuleiro,sinaisL,sinaisC,size):
 l = 0
 while l < size:
  c = 0
  while c < size:
    print(tabuleiro[l][c],end = "")
    if c != (size - 1) and sinaisL[l][c] != 0:
     if(sinaisL[l][c] == 1):
      print(' > ',end = "")
     else:
      print(' < ',end = "")
    else:
     print('   ',end = "")
    c += 1

  print('')
  c = 0
  while c < size:
    if l != (size - 1) and sinaisC[l][c] != 0:
     if(sinaisC[l][c] == 1):
      print('V',end = "")
     else:
      print('^',end = "")
    else:
     print(' ',end = "")
    print('   ',end = "")
    c += 1

  print('')
  l+=1
 return

s = 0
sV = re.compile(r'SinalV ([12]) (\d+) (\d+)')
sH = re.compile(r'SinalH ([12]) (\d+) (\d+)')
file = open("ex.txt", 'r')
SVertical = dict()
SHorizontal = dict()
data = []
for row in file:
       data.append([int(x) for x in row.split()])
       s+=1

with open("s.txt", 'r') as f:
  file2 = f.read()
  SVertical = np.zeros((s - 1,s), dtype=int)
  SHorizontal = np.zeros((s,s - 1), dtype=int)
  contentV = re.findall(sV, file2)
  contentH = re.findall(sH, file2)

  for l in contentV:
    SVertical[int(l[1])][int(l[2])] = int(l[0])

  for l in contentH:
    SHorizontal[int(l[1])][int(l[2])] = int(l[0])

print("Tabuleiro Inicial")
printTabuleiro(data,SHorizontal,SVertical,s)

# 9x9 matrix of integer variables
X = [ [ Int("x_%s_%s" % (i+1, j+1)) for j in range(s) ]
      for i in range(s) ]

# each cell contains a value in {1, ..., 9}
cells_c  = [ And(1 <= X[i][j], X[i][j] <= s)
             for i in range(s) for j in range(s) ]

# each row contains a digit at most once
rows_c   = [ Distinct(X[i]) for i in range(s) ]

# each column contains a digit at most once
cols_c   = [ Distinct([ X[i][j] for i in range(s) ])
             for j in range(s) ]
```

```
sudoku_c = cells_c + rows_c + cols_c


instance_c = [ If(data[i][j] == 0,
                  True,
                  X[i][j] == data[i][j])
               for i in range(s) for j in range(s) ]

sv = Solver()
sv.add(sudoku_c + instance_c)

for l in range(s):
  for c in range(s-1):
    if(SHorizontal[l][c] == 1):
      sv.add(X[l][c] > X[l][c + 1])
    elif(SHorizontal[l][c] == 2):
      sv.add(X[l][c] < X[l][c + 1])

for l in range(s-1):
  for c in range(s):
    if(SVertical[l][c] == 1):
      sv.add(X[l][c] > X[l + 1][c])
    elif(SVertical[l][c] == 2):
      sv.add(X[l][c] < X[l + 1][c])

if sv.check() == sat:
    m = sv.model()
    r = [ [ m.evaluate(X[l][c]) for c in range(s) ]
          for l in range(s) ]
    print("Tabuleiro Solução")
    printTabuleiro(r,SHorizontal,SVertical,s)
else:
    print ("failed to solve")
```

```
Requirement already satisfied: z3-solver in /usr/local/lib/python3.7/dist-pac
Tabuleiro Inicial
0 > 0   0   0

0   2 > 0   0

0   0   0   0
        v
0 > 0   0   0

Tabuleiro Solução
2 > 1   4   3

3   2 > 1   4

1   4   3   2
        v
4 > 3   2   1
```



Futoshiki.org

Play Futoshiki Puzzles Online

Futoshiki

Board size: 4 Difficulty: easy On click: show selector

```
2 > 1   4   3

3   2 > 1   4

1   4   3   2
        v
4 > 3   2   1
```

The current puzzle is successfully solved! New puzzle

Futoshiki Other games

Futoshiki is a board-based puzzle game, also known under the name **Unequal**. It is playable on a square board having a given fixed size (4x4 for example).

The purpose of the game is to discover the digits hidden inside the board's cells; each cell is filled with a digit between 1 and the board's size. On each row and column each digit appears exactly once; therefore, when revealed, the digits of the board form a so-called **Latin square**.

At the beginning of the game some digits might be revealed. The board might also contain some inequalities between the board cells; these inequalities must be respected and can be used as clues in order to discover the remaining hidden digits.

Each puzzle is guaranteed to have a solution and only one. In order to indicate a move, click the desired square and select a digit or the delete sign (X); you can also use the digits on your keyboard (in this case, the digit 0 is equivalent to the delete sign).

For tips and tricks, you can check out our tutorial: how to solve a Futoshiki puzzle.

✓ 3 s concluído à(s) 23:47