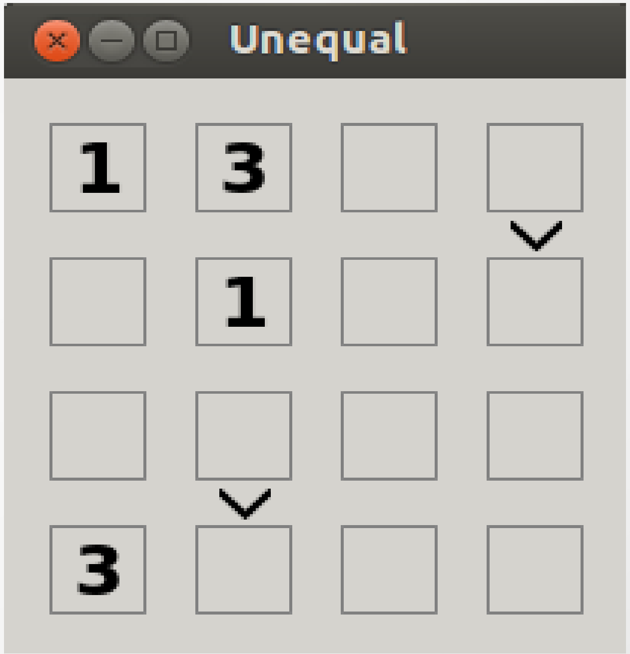
FUTOSHIKI

Aluno: Bruno Barreto



FUTOSHIKI

Futoshiki (不等式), se trata de um quebra-cabeça de lógica japonês, Seu nome significa “desigualdade”. Também é escrito Hutoshiki (usando Kunrei-shiki).

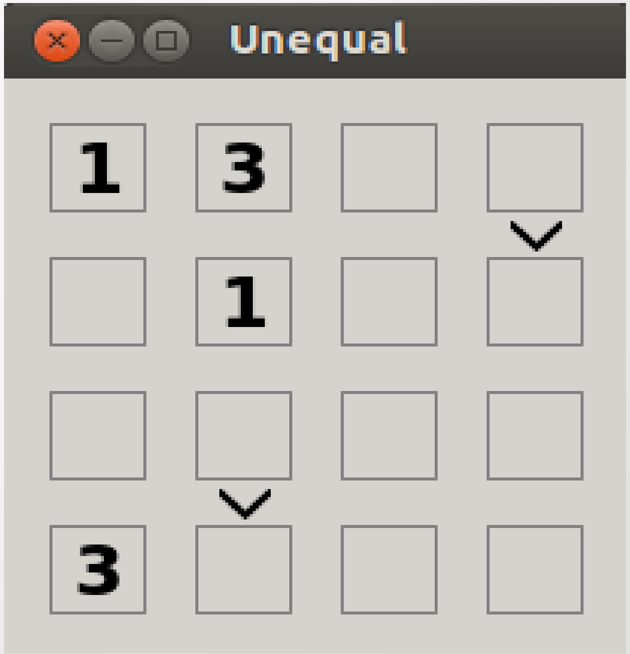
Uma instância do Futoshiki é dada omo um *grid* n x n de células de tal forma que algumas destas células estão vazias e outras estão preenchidas com um número inteiro presente em [n] = $\{1, 2, \ldots, n\}$, e alguns pares de células adjacentes possuem um símbolo de desigualdade (“<” ou “>”) indicando que estas células vizinhas estão sujeitas a esta restrição.

FUTOSHIKI

Dada uma instância, um solucionador é convidado a preencher as células que estão vazias com valores inteiros presentes em [n] de modo que os n^2 inteiros no *grid* satisfaçam as seguintes condições.

Condição do quadrado latino: em cada liha e em cada coluna, cada um dos inteiros em [n] aparecem exatamente uma vez.

Condição de desigualdade: quando duas células adjacentes possuem um sinal de desigualdade entre elas, os inteiros atribuídos às duas células devem satisfazer a desigualdade.



FUTOSHIKI

Futoshiki faz parte de um grupo de problemas que podem ser resolvidos por métodos de satisfação de restrições do inglês *constraint satisfaction problems* (CSPs), ou seja, um problema matemático definido como um conjunto de objetos cujo estado dos mesmos deve satisfazer uma série de restrições.

CSPs geralmente apresentam alta complexidade e são custosos exigindo que sejam usados métodos heurísticos e de busca combinatória para que se possa resolver tais problemas em tempo aceitável.

Dentre as técnicas mais usadas estão variantes dos algoritmos de backtracking, propagação de restrições e busca local.

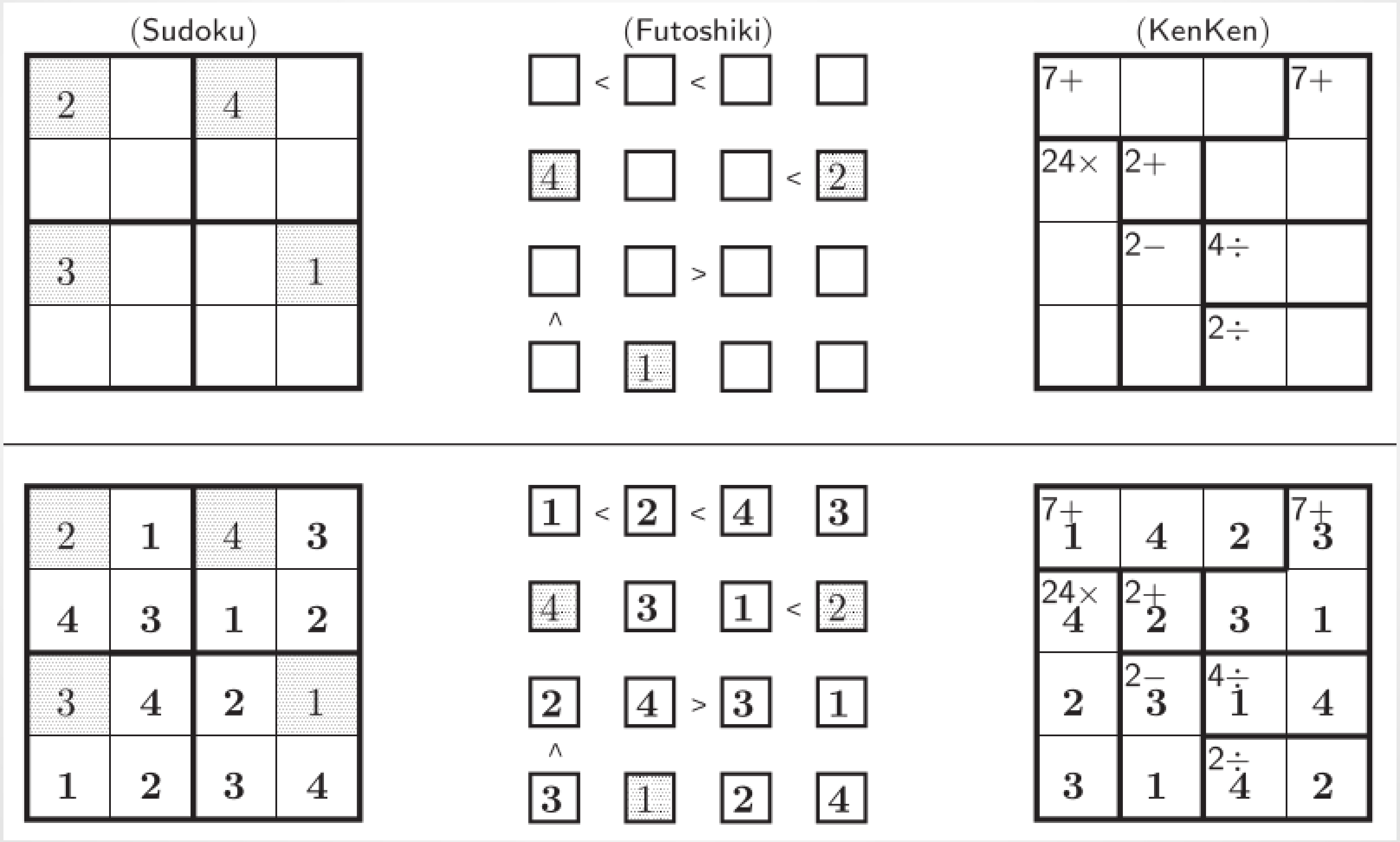
Características Interessantes

- Conjunto Crítico
  - Um conjunto crítico em um quadrado Futoshiki é uma coleção de números e comparações que determina unicamente aquele quadrado Futoshiki, e que não tem subconjuntos próprios que também iria fazê-lo.

Se *S* é um conjunto crítico (*h*, *k*) para um quadrado Futoshiki de tamanho n, onde *h* são os números e *k* as comparações então:

$$2h + 3k \geq 2\,(n - 1).$$

Problemas Semelhantes



Complexidade

- O Futoshiki é NP-difícil e a prova está no artigo [HARAGUCHI 2015] e no relatório, basicamente foi realizada uma redução da extensão parcial do quadrado latino.
- Uma instância do futoshiki terá uma única solução se esta além de possuir as características do futoshiki possuir um conjunto crítico.

FUTOSHIKI

O Futoshiki é NP-Difícil, pois podem haver grids com alto grau de complexidade com apenas 1 solução, tornando a resolução deste custosa computacionalmente. As instâncias podem ser irresolvíveis, possuir dificuldade elevada ou ser fácil (possuir mais de uma solução possível). Exemplo de instâncias:

Fácil

```
0 | 0 | 0 | 0 | 0
v - - - - - - - v
0 > 0 | 0 | 0 | 3
- - - - - - - - ^
0 > 0 < 0 | 2 | 0
- - - - - - - - -
0 | 0 | 0 | 0 | 0
^ - - - ^ - - - -
0 | 0 | 0 | 0 | 0
```

Médio

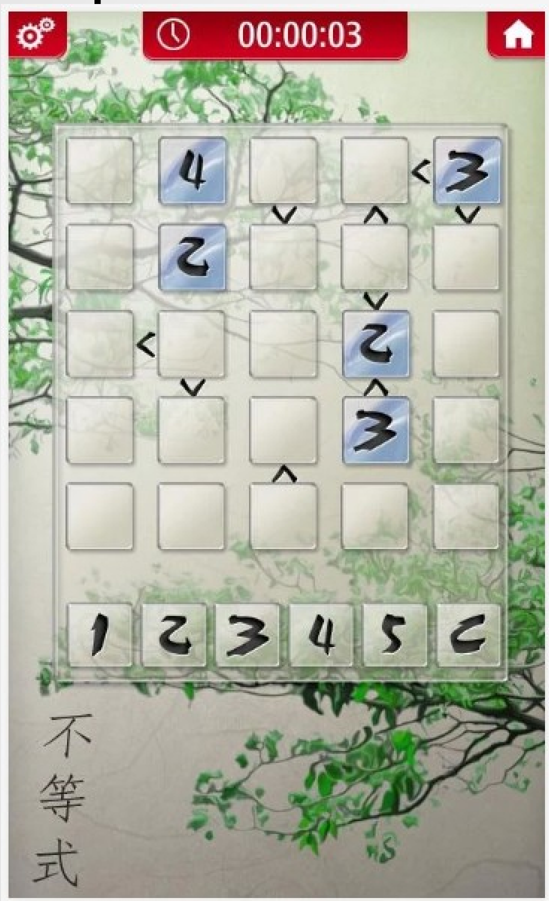
```
0 | 0 | 0 | 0 | 0
v - - - - - v - -
0 | 0 | 0 | 0 | 0
- - ^ - ^ - - - -
0 > 0 < 0 | 0 | 0
- - - - - - - - ^
0 | 0 < 0 | 0 > 0
v - - - - - - - -
2 | 0 | 0 | 0 | 0
```

Difícil

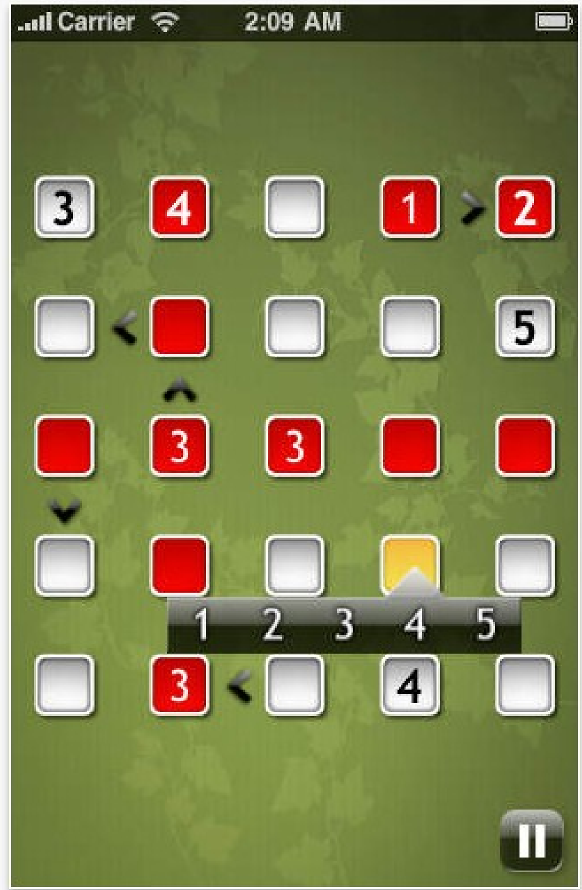
```
0 | 0 | 0 > 0 | 0
- - v - - - - - -
0 | 0 | 0 | 0 | 0
- - v - - - - - -
0 | 0 | 0 | 0 < 0
- - - - - - - - -
1 | 0 | 0 | 0 < 0
- - - - - - - - ^
5 | 1 | 0 | 0 | 0
```

FUTOSHIKI

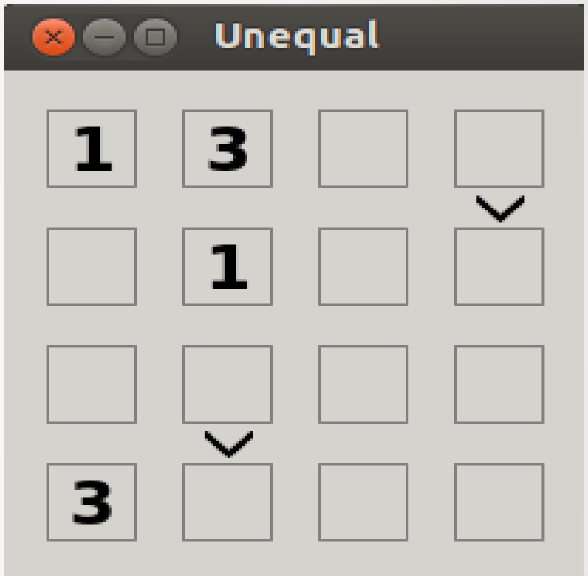
Aplicativos Futoshiki



Android
Futoshiki



Itunes
Pocket Futoshiki



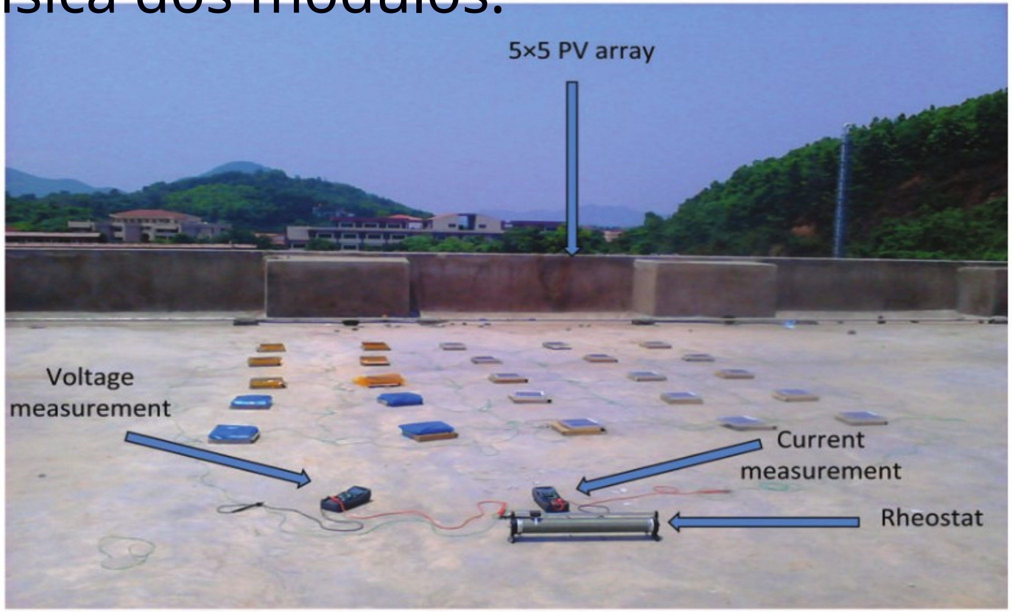
Ubuntu
Unequal

Aplicações

O futoshiki foi recentemente utilizado em um experimento [SAHU 2015] para melhorar a eficiência energética de um sistema fotovoltaico quando este está sujeito a um sombreamento parcial.

As perdas em um gerador fotovoltaico dependem do padrão de sombreamento e a localização física dos módulos.

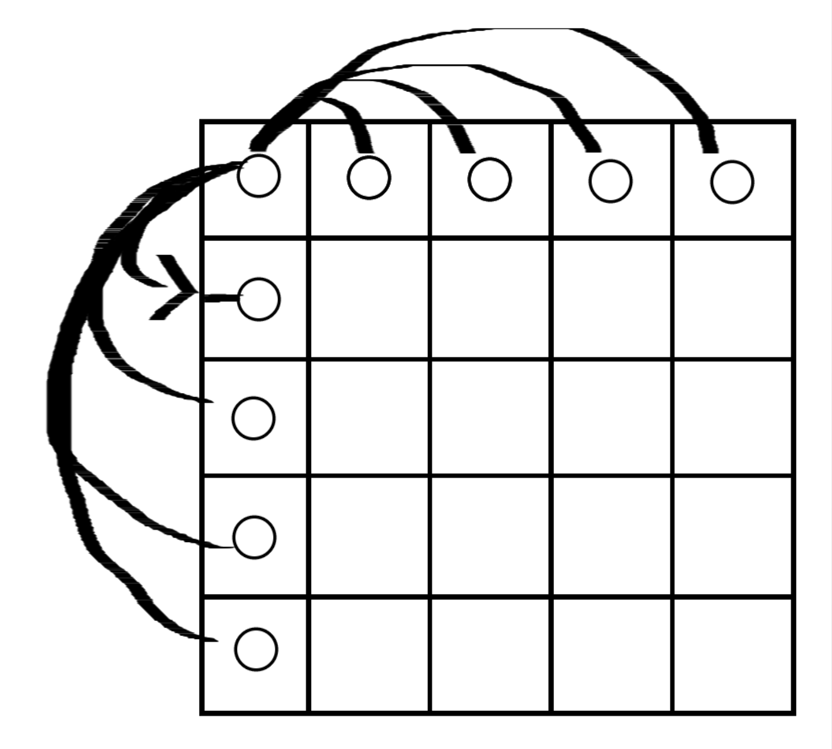
As perdas em um gerador fotovoltaico dependem do padrão de sombreamento e a localização física dos módulos.



Modelagem Matemática

$$Maximize\ z=\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}x_{ijk}$$

Sujeito a:



$$\sum_{valor}^{n} x_{linha,coluna,valor}=1,\forall\, linha,coluna$$

$$\sum_{coluna}^{n} x_{linha,coluna,valor}=1,\forall\, linha,valor$$

$$\sum_{linha}^{n} x_{linha,coluna,valor}=1,\forall\, coluna,valor$$

$$\sum_{valor\,1}^{n} x_{linha\,1,coluna\,1,valor\,1} * valor\,1\geqslant 1\sum_{valor\,2}^{n} x_{linha\,1,coluna\,1,valor\,2} * valor\,2$$

FUTOSHIKI – Backtracking puro

```
Procedimento solveFutoshiki (matrix, row, col)  // backtracking que resolve o problema
  if isSolution(matriz) then  // se é uma solução
    write matriz
  else
    if matriz[row][col] possui peso then
      if coluna chegou no fim then
        solveFutoshiki(matriz, row + 2, 0)        // vai para a próxima linha
      else    //não chegou ao fim da coluna
        solveFutoshiki(matriz, row, col + 2)  //vai para a próxima celula
    else
      for each i = 1, …, n then //para cada celula a possibilidade de adicionar algum valor
        matriz[row][col] = i (peso)  //adiciona peso
        solveFutoshiki(matriz, row, col + 2)      //vai para a próxima celula
        if col chegou ao fim then
          solveFutoshiki (matriz, row + 2, 0)    // vai para a próxima linha
      matriz[row][col] = 0  //backtracking
```

FUTOSHIKI – Backtracking com poda

```
Procedimento solveFutoshiki (matrix, row, col) // backtracking que resolve o problema
  if isSolution(matriz) then    // se é uma solução
    write matriz
  else
    if matriz[row][col] possui peso then
      if coluna chegou no fim
        solveFutoshiki(matriz, row + 2, 0)        // vai para a próxima linha
      else     //não chegou ao fim da coluna
        solveFutoshiki(matriz, row, col + 2) //vai para a próxima celula
    else
      for each i = 1, …, n then //para cada celula a possibilidade de adicionar algum valor
        // se i (peso) pode ser adicionado em matriz[row][col]
        if isValid(matriz, row, col, i) then
          matriz[row][col] = i (peso)  //adiciona peso
          solveFutoshiki(matriz, row, col + 2)       //vai para a próxima celula
          if col chegou ao fim then
            solveFutoshiki (matriz, row + 2, 0)    // vai para a próxima linha
      matriz[row][col] = 0     //backtracking
```

Backtracking c/ Manipulação de Bits

```
backtrack_bit_manipulation( matrix[1 … n][1 … n] , row, col)
  if( matrix is a solution) then
    write ( matriz )
  else
    for each value [1 … n] then
      mask = 1 << value
      if( value can be added ) then
        matrix[row][col] = value
        rows[row] |= mask
        cols[col] |= mask
        if( backtrack_bit_manipulation(row, col + 1) ) then
          return true
        matrix[row][col] = 0  //backtrack
        rows[row] &= ~mask
        cols[col] &= ~mask
        blocks[block] &= ~mask
   return false;
```

Prova que o Futoshiki é NP

```
isSolution(matrix[1 … n][1 … n] )
    for each i = 1, …, n then
        for each j = 1, …, n then
            for each k = 1, …, n then
                if(matrix[i][j] == matrix[i][k]) then  //se há repetições na linha
                    return 0
                if(matrix[j][i] == matrix[k][i]) then  //se há repetições na coluna
                    return 0
            if(matrix[i][j] == ∅) then   // se há valores nulos
                return 0
    // checagem em relação às regras do jogo (><^v)
    for each i = 1, …, n then
        for each j = 1, …, n then
            if(a restrição de desigualdade entre 2 células adjacentes não é respeitada) then
                return false
    return true  // É um resultado
```

FUTOSHIKI – Fatos interessantes

O Futoshiki é publicado todos os sábados no The Guardian e The Daily Telegraph. Ele aparece diariamente no jornal The Times (exceto aos sábados) e no Dundee Courier.

FUTOSHIKI – Referências

BERTHIER, Denis. Pattern-Based Constraint Satisfaction and Logic Puzzles. arXiv preprint arXiv:1304.1628, 2013.

SUTANTO, Hendy. Combination of BFS and Brute Force Algorithm Implementation in Futoshiki Puzzle Game. Institut Teknologi Bandung.

HARAGUCHI, Kazuya; ONO, Hirotaka. How Simple Algorithms Can Solve Latin Square Completion-Type Puzzles Approximately.Journal of Information Processing, v. 23, n. 3, p. 276-283, 2015.

HARAGUCHI, Kazuya. The number of inequality signs in the design of Futoshiki puzzle. Journal of Information Processing, v. 21, n. 1, p. 26-32, 2013.

BLANK, Shohreh; AZMOODEH, Manoochehr; ALAMO-PRITCHARD, Nathan Frederick. SYSTEM AND METHOD FOR GENERATING AND USING SOLVABLE PUZZLE FORMS. U.S. Patent n. 20,150,137,449, 21 maio 2015.

SAHU, HIMANSHU; NAYAK, SISIR; MISHRA, Sukumar. Maximizing the Power Generation of a Partially Shaded PV Array.

Critical Sets in Futoshiki Squares. https://www.math.brown.edu/~dkatz/futoshiki-talk.pdf