

# Universidade Federal de Santa Catarina

Centro Tecnológico  
Departamento de Informação e Estatística  
Ciências da Computação

Enzo da Rosa Brum

Relatório de paradigmas de programação

Florianópolis  
2023

# 1 Problema

Neste trabalho, foi decidido criar um resolvidor de sudoku comparativo (vergleichssudoku). As regras do jogo são:

1. Células na mesma linha não podem ter o mesmo valor
2. Células na mesma coluna não podem ter o mesmo valor
3. Células na mesma região não podem ter o mesmo valor
4. Os sinais »”entre duas células indicam que o número na célula para a qual a seta aponta é menor do que o número na outra célula.

## 2 Solução

### 2.1 Algoritmo

Para resolver o sudoku comparativo, foi necessário reduzir o jogo à um problema de cobertura exata e usar o "Algorithm X" para resolver tal problema.

### 2.2 Problema de cobertura exata

Um problema de cobertura exata pode ser descrito pela seguinte pergunta: Dada uma matriz de 0s e 1s, ela possui um conjunto de linhas contendo exatamente um 1 em cada coluna?

Por exemplo, a matriz

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

forma um conjunto solução com as linhas 2 e 3. Enquanto as linhas 1, 2, e 4 não podem ser consideradas solução devido à presença de múltiplos 1s na coluna 3.

### 2.3 Dancing Links

Em [1], Knuth introduz um de resolver problemas de cobertura exata de forma rápida e eficiente por meio de estrutura chamada de "Dancing Links" e um algoritmo conhecido como "Algorithm X". Resumidamente, o algoritmo descrito por Knuth se aproveita da remoção e reinserção eficiente em listas encadeadas para realizar um backtracking com grande eficiência. Por exemplo, essas operações permitem

$node \rightarrow left \rightarrow right = node \rightarrow right;$   $node \rightarrow right \rightarrow left = node \rightarrow left;$   
remover e

$node \rightarrow left \rightarrow right = node;$   $node \rightarrow right \rightarrow left = node;$

reinsere um nó em  $O(1)$ . Nesse sentido, Knuth propõe a utilização de duas listas circulares para representar a matriz binária de um problema de cobertura exata, onde cada nó representaria um 1 na matriz.

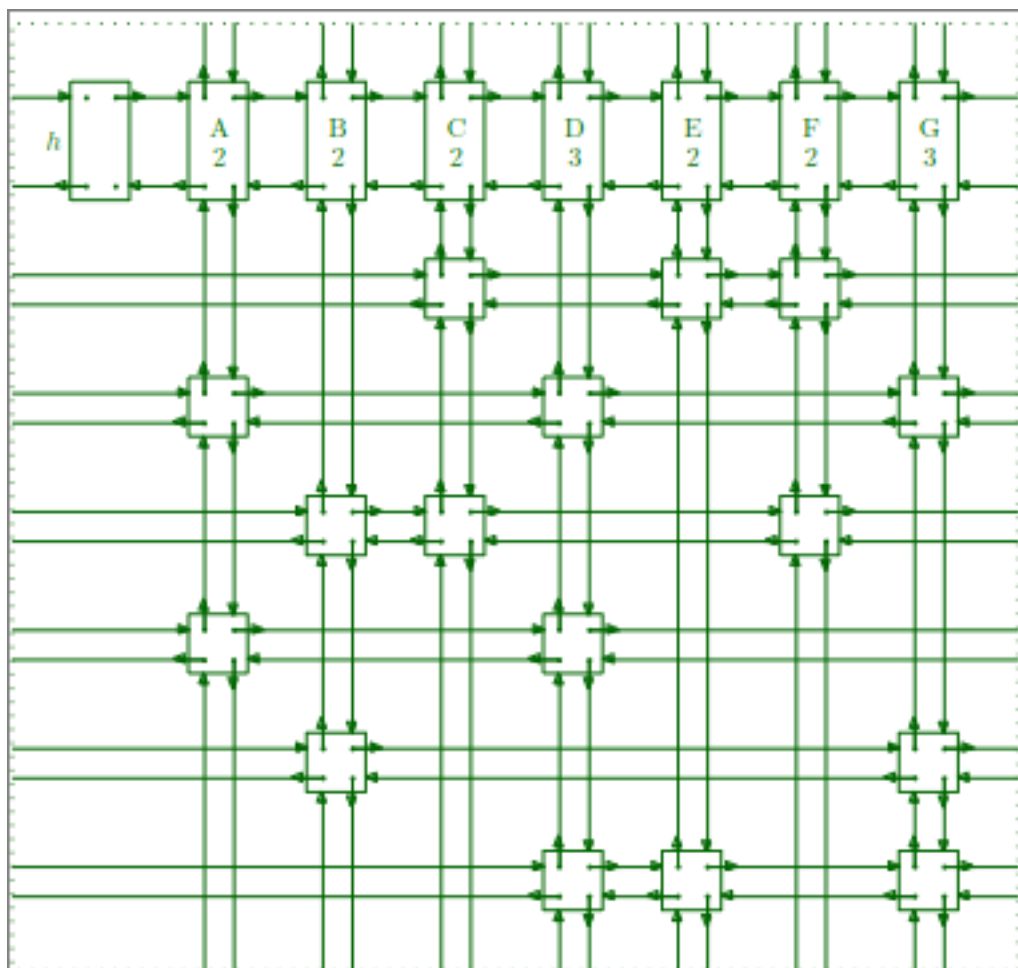
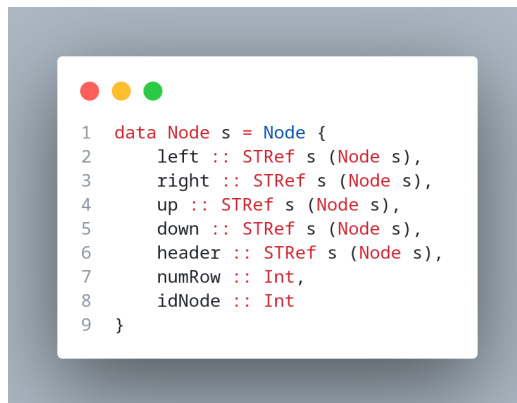


Figura 1: Dancing Links. Fonte: [1]

A screenshot of a code editor window with a light blue background. At the top left of the editor are three colored circles: red, yellow, and green. The code is written in Haskell and defines a data structure for a node in a graph. It uses STRefs for mutable pointers to other nodes. The code is as follows:

```
1 data Node s = Node {
2     left :: STRef s (Node s),
3     right :: STRef s (Node s),
4     up :: STRef s (Node s),
5     down :: STRef s (Node s),
6     header :: STRef s (Node s),
7     numRows :: Int,
8     idNode :: Int
9 }
```

Figura 2: Implementação da estrutura Node.

Para implementar os nós, o código acima foi utilizado. `left`, `right`, `up`, `down` e `header` são ponteiros para outros nós. Como o haskell não possui referências mutáveis por padrão, foi necessário utilizar o monad ST para implementar a lista da forma idealizada por Knuth.

### 3 Rodando o programa

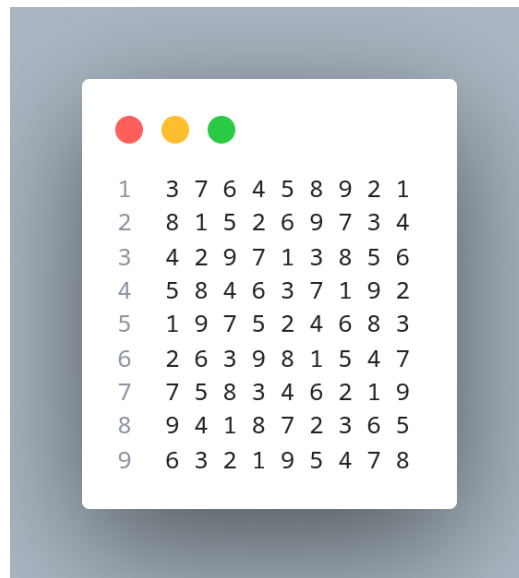
Para compilar o programa, use o comando `make all` para gerar o arquivo executável. Depois disso, use `./main <arquivo de input` para rodar o programa.

Um arquivo de input equivale à solução de um sudoku qualquer e deve estar no seguinte formato:

Quanto a saída do programa, o resultado do resolvidor é impresso no terminal em formato semelhante ao arquivo de input e, logo após isso, uma linha é impressa indicando se a solução encontrada pelo programa é a mesma que está presente no arquivo de input.

### 4 Dificuldades encontradas

A escolha de utilizar `Dancing Links` ao invés de um backtracking mais simples fez com que a implementação do trabalho levasse muito mais tempo que o necessário, o que por sua vez, me deixou com pouquíssimo tempo para escrever este relatório. Espero fazer justiça à esse algoritmo maravilhoso quando for implementá-lo para o trabalho 2. (Ou não, fazer um documento explicando este



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

Figura 3: Arquivo de input.

algoritmo provavelmente levaria mais tempo do que a implementação...)

## Referências

- [1] Donald E. Knuth. Dancing links. *Millennial Perspectives in Computer Science*, 2000:187, 2000.