Trabalho Prático 3 - Algoritmos 1

Victor Hugo Silva Moura - 2018054958

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte - MG – Brasil

1 Introdução

O sudoku é um puzzle de lógica baseado, em sua versão tradicional, na colocação de números em uma grade nxn. Os números são colocados de forma que nenhum número seja repetido nas linhas, nas colunas e em subquadrantes de tamanho n cada.

Tradicionalmente os quadrantes do sudoku são quadrados perfeitos, ou seja, um sudoku 9x9 tem quadrantes 3x3 por exemplo, que é um quadrado perfeito. Porém existem versões que não são perfeitas, como por exemplo um sudoku 8x8 onde cada quadrante tem tamanho 2x4, o que não é um quadrado perfeito. Para este trabalho, essas versões também serão utilizadas.

O objetivo do trabalho é resolver o problema do Sudoku por meio de uma transformação possível para ele, que é o problema da Coloração de Grafos. Nesse problema, dado um grafo $S(V, A)$ deve-se encontrar o menor número de cores que podem ser utilizadas para colorir o grafo sem que haja repetições de cores em vertices adjacentes. No caso do trabalho, não se busca o menor número k de cores possíveis, uma vez que já se conhece que $\chi(S) = n$, sendo $\chi(S)$ o número de cores necessárias para colorir o grafo S.

Sendo assim, é necessário transformar o sudoku em um grafo que respeite as restrições do sudoku (vértices na mesma linha, coluna ou bloco não podem ter a mesma cor). A montagem desse grafo será discutida na seção de Implementação. Como o problema de Coloração de Grafos é NP-Completo, é necessário utilizar uma solução aproximada, visto que a solução exata exige um tempo de computação exponencial, o que a torna inviável.

2 Implementação

Para a implementação desse problema, foi utilizada uma transformação do Sudoku para um problema de Coloração de Grafos, assim como dito na introdução. Essa transformação é feita de forma a respeitar as restrições do sudoku (vértices na mesma linha, coluna ou bloco não podem ter a mesma cor).

Para montar uma instância da Coloração de Grafos utilizando o sudoku, o primeiro passo é transformar cada célula em um vértice do grafo. Sendo assim, um sudoku nxx gera um grafo de nxn vértices para a coloração. O próximo passo é fazer as conexões do grafo de modo que essas conexões permitam com que as restrições do sudoku sejam mantidas. Para cada vértice do grafo, o mesmo é conectado aos vértices que representam elementos da mesma coluna, linha ou bloco. Dessa forma, como na coloração dois vértices adjacentes não podem ter a mesma cor, estamos garantindo no sudoku que nenhuma linha, coluna ou bloco terá números repetidos. Caso tenha, quer dizer que dois vértices adjacentes foram coloridos da mesma cor, o que não é possível. Abaixo temos uma imagem do grafo fornecida no enunciado do trabalho:

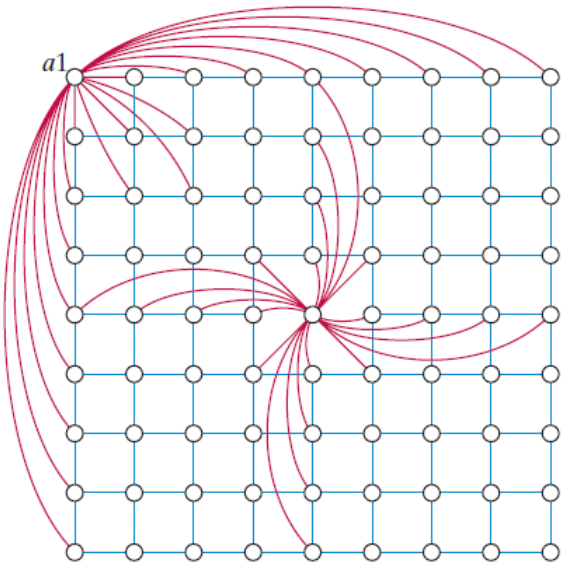

Figura 1: Exemplo de parte do grafo para um sudoku 9x9. Note que o vértice $a1$ está conectado a todos os vértices de sua linha coluna e bloco. O mesmo vale para o vértice central do grafo.

Sendo assim, agora vamos à solução para o sudoku. A solução desenvolvida utiliza a seguinte heurística: "*para cada novo preenchimento de cores no grafo, escolha o vértice com menos opções disponíveis, que ainda não tenha sido preenchido, e preencha-o com a primeira cor disponível para ele*". Em outras palavras, para cada vez que uma nova cor tiver que ser preenchida no grafo, o vértice com a menor opção de cores disponível, ou seja, o vértice cujo seus vizinhos tem a maior variedade de cores (impossibilitando que essas cores sejam escolhidas para esse vértice), e que ainda esteja descolorido é escolhido para ser preenchido. A cor com a qual ele será preenchido é a primeira cor disponível para ele. Exemplo: Suponha que as cores para a coloração são {"Azul", "Verde",

"Amarelo", "Vermelho"} e o vértice que será preenchido não pode ser preenchido com as cores "Azul" e "Vermelho". A primeira cor disponível, seguindo a ordem das cores do grafo é a cor "Verde" e é com essa cor que o vértice será preenchido.

O preenchimento só termina com duas condições, que são as condições de solução do sudoku. Elas são:

- Se o sudoku estiver completamente preenchido. Nesse caso a execução termina e é retornado ao usuário que uma solução foi encontrada.
- Se ao tentar colorir algum vértice, não houverem opções de cor disponíveis para ele. Nesse caso, é retornado ao usuário que a solução não foi encontrada pela heurística.

Para o primeiro caso, se o sudoku estiver completo, não existe nenhum vértice no grafo que não foi colorido, o que indica que a coloração terminou e, por consequência, o sudoku. Além disso, para chegar nesse estado, nenhum vértice do grafo foi colorido com a mesma cor de um vértice adjacente, o que indica que no sudoku nenhuma célula foi preenchida com o mesmo número de outra célula na mesma linha, coluna ou bloco, devido à construção do grafo.

Para o segundo caso, se algum vértice não possui cor para ele, isso significa que durante o processo de coloração, todos os vértices adjacentes a ele foram coloridos de forma que todas as cores foram utilizadas. Dessa forma, o único jeito de reverter isso seria por meio de *backtracking*. Porém, a heurística proposta não utiliza essa técnica. Assim, ao chegar nesse estado, não há solução possível para a coloração, e, por consequência, para o sudoku.

3 Análise de Complexidade

3.1 Complexidade de Tempo

3.2 Complexidade de Espaço

4 Provas de Corretude

5 Análise Experimental

Referências