Class Project Report
Advanced Algorithms CMSC-501- Spring 2018
Virginia Commonwealth University

Sina Ghadermarzi

Introduction

In this project an algorithm has been implemented for coloring graphs in which some of nodes have already assigned with some colors (super graph coloring). This is a greedy algorithm which uses a simple strategy: it processes vertices in an arbitrary order and assigns them with the first available color (one that is not used by its neighbors). It is implemented in C++ and the instruction for compile and use is in Compile_Instructions.txt

Results on random graphs

4 random graphs (Erdős–Rényi) are generated by finding *e* random unique (unordered) pair of nodes. This is done with python program named "random_graph.py". This program creates 4 random graph files in appropriate format. The name of graph files indicates number of nodes, edges and maximum degree of graphs.

For example for random graphs of size 50, 100, 200, 500, the results of applying super graph coloring program on these graphs are summarized in the table below

Size of Graph	(v= 50 e = 500)	(v= 100 e = 1000)	(v= 200 e = 2000)	(v= 500 e = 5000)
Maximum Degree	27	31	31	34
Number of Colors Used	11	11	12	11

Solving Sudoku Puzzle with Super Graph Coloring

The Sudoku problem can be translated to graph coloring problem by assigning one node to each of the cells in the table and connecting a pair of nodes by an edge if they are either on the same row, same column or same block. In the resulting graph, any 9-coloring of the graph corresponds to a valid solution for the Sudoku puzzle. This works because in this coloring, no adjacent nodes have same color. Which means that in the corresponding table, no pair of cells in the same row, column or blocks will have same number. If some of the cells in the table are filled with numbers already, it can be solved by our super graph coloring problem (in which some of vertices have already colored.

If we number the 81 vertices by numbers in 0…80, the cell in row i (i in 0…8) and column j (j in 0…8) corresponds to vertex number $j + i \times 9$.

And for a node number k:

$Row = k / 9$

$Column = k \% 9$

$Block\ Row = Row / 3$

$Block\ Column = Column / 3$

The python program “create_sudoku_graph.py” can generate the graph file for a Sudoku table given in file “sudoku.csv”. The program saves the graph file as “graph_sudoku.csv”. It then calls the graph coloring program and prints the resulting filled Sudoku.

For the Sudoku given in the project description document, unfortunately the greedy algorithm can’t color the graph with only 9 colors and it needs two extra colors. For this Sudoku, the resulting table after coloring looks like this:

5	7	1	2	6	3	10	8	4
4	2	3	1	8	7	9	5	6
9	8	6	4	5	10	3	2	1
1	6	5	3	4	2	7	9	8
3	4	7	9	10	1	2	6	5
2	9	8	5	7	6	1	3	10
8	5	4	7	3	9	6	1	2
6	1	2	10	11	5	8	4	3
7	3	9	6	1	4	5	10	11