Graph Coloring

Michelle Gonzalez Alvarado

CSC 326

Data Structure

Professor: Tatiana Balashova Anderson

**Resources**

Graph Coloring: History, Results and Open Problems - Vitaly I. Voloshin

A Formal Proof of the Four Color Theorem - Limin Xiang

Graph Theory - Princeton University

                            Graph Coloring Introduction

In mathematics, graph theory is the study of graphs, which are mathematical structures used to model pairwise relations between objects. A graph can be made up of vertices, also called nodes or points, which are connected by edges, also called links or lines. In graph theory, graph coloring is a special case when it comes to graph labeling. It is an assignment of labels traditionally called "colors" to elements of a graph subject to certain constraints. In its simplest form, it is a way of coloring the vertices of a graph such that no two adjacent vertices are of the same color, which is called a vertex coloring. Similarly, an edge coloring assigns a color to each edge so that no two adjacent edges are of the same color, and a face coloring of a planar graph assigns a color to each face or region so that no two faces that share a boundary have the same color. In this paper, we will dive into what coloring represents, the history of the four color theorem, what they were trying to prove, algorithms, and applications of coloring.

         What Coloring Represents

Graphs are used to depict ”what is in conflict with what”, and colors are used to denote the state of a vertex. So, more precisely, coloring theory is the theory of “partitioning the sets having internal irreconcilable conflicts” because we will only count “good” colorings. Coloring represents matching regions together. For example, coloring two countries of a map in such a way that no two countries that have a common border receive the same color. When talking about coloring, it dates back to Guthrie trying to color the map of countries. Francis Guthrie, who was a [South African](https://en.wikipedia.org/wiki/South_African) mathematician, tried to color the map of counties of England noticed that four colors suffice. He asked his brother, Frederick if it was true that any map can be colored using four colors in such a way that adjacent regions (i.e. those sharing a common boundary segment, not just a point) receive different colors. Frederick Guthrie then communicated the conjecture to De Morgan, who was a British [mathematician](https://en.wikipedia.org/wiki/Mathematician) and [logician](https://en.wikipedia.org/wiki/Logician). The first printed reference is due to Cayley, who was a was a prolific [British](https://en.wikipedia.org/wiki/United_Kingdom_of_Great_Britain_and_Ireland) [mathematician](https://en.wikipedia.org/wiki/Mathematician)  in 1878.

  Two Color Graphs and Bipartite

A Bipartite Graph also called a bi-graph, is a set of graph vertices, i.e, points where multiple lines meet, decomposed into two disjoint sets, meaning they have no element in common, such that no two graph vertices within the same set are adjacent. is a graph whose vertices can be divided into two independent sets, U and V such that every edge (u, v) either connects a vertex from U to V or a vertex from V to U. In other words, for every edge (u, v), either u belongs to U and v to V, or u belongs to V and v to U. We can also say that there is no edge that connects vertices of the same set. The special rule for bipartite graphs is that only two different colors are allowed.

History of The Four Color Theorem

When mentioning about coloring the countries on a map, If we denote the countries by points in the plane and connect each pair of points that correspond to countries with a common border by a curve, we obtain a planar graph. The celebrated Four Color Theorem asks if every planar graph can be colored with 4 colors. It seems to have been mentioned for the first time in writing in an 1852 letter from A. De Morgan to W.R. Hamilton, who was an Irish mathematician. Nobody thought at that time that it was the beginning of a new theory.

What They Were Trying to Prove

 The first ”proof” was given by Kempe, London barrister and amateur mathematician in 1879. It stood for more than 10 years until Percy Heawood, a mathematician, in the year 1890 found a mistake. Heawood proved that five colors are enough to color any map. The Four Color Theorem became one of the most difficult problems in Graph Theory, besides colorings it stimulated many other areas of graph theory. Generally, coloring theory is the theory about conflicts: adjacent vertices in a graph must always have distinct colors, i.e. they are in a permanent conflict. As mentioned previously, if we have a “good” coloring, then we respect all the conflicts. If we have a “bad” coloring, then we have a pair of adjacent vertices colored with the same color. This looks like having a geographic map where some two countries having common border are colored with the same color. Graphs are used to depict “what is in conflict with what”, and colors are used to denote the state of a vertex. So, more precisely, coloring theory is the theory of “partitioning the sets having internal irreconcilable conflicts” because we will only count “good” colorings.

The next major contribution came from Birkhoff, [American mathematician](https://en.wikipedia.org/wiki/American_mathematician) best known for what is now called the [ergodic theorem](https://en.wikipedia.org/wiki/Ergodic_theorem), whose work allowed Philip Franklin, who was an American mathematician and professor whose work was primarily focused on analysis, in 1922 to prove that the four color conjecture is true for maps with at most 25 regions. It was also used by other mathematicians to make various forms of progress on the four color problem. We should specifically mention Heesch who developed the two main ingredients needed for the ultimate proof reducibility and discharging. While the concept of reducibility was studied by other researchers as well, it appears that the idea of discharging, crucial for the unavoidability part of the proof, is due to Heesch, and that it was he who conjectured that a suitable development of this method would solve the Four Color Theorem. This was confirmed by Appel and Haken in 1976, when they published their proof of the Four Color Theorem.

In 1912, Birkhoff, working at Princeton, published the following paper: ”A determinant formula for the number of ways of coloring a map”, Birkhoff (1912). In this paper, he introduced a function denoted by P(G, λ) which gives the number of proper colorings of a graph G using the colors from set {1, 2, 3, ...λ}. Since it is a polynomial, it was called the chromatic polynomial of a graph. Suppose the chromatic number of a graph G is χ(G); it means there are no colorings using 1, 2, 3, ... χ(G) − 1 colors, and there is at least one coloring using χ(G) colors. It implies that P(G, 1) = P(G, 2) = ... = P(G, χ−1) = 0 and P(G, χ) , 0. In other words, the numbers λ =1, 2, 3,... χ − 1 all are the roots of P(G, λ) and the number λ = χ(G) is not a root. Birkhoff was motivated by the colorings of maps and his basic goal was the following: prove that for every map = planar graph G, P(G, 4) , 0. That meant that any counter example G to the four color problem must have the root λ = 4 in its chromatic polynomial. It ”only” remained to investigate the behavior of the roots (just one root!) in the chromatic polynomials of planar graphs to solve the famous four color theorem. Birkhoff and others have shown that the chromatic polynomial of a graph on n vertices has degree n, with leading coefficient 1 and constant term 0. Furthermore, the coefficients alternate in sign, and the coefficient of the second term is −m, where m is the number of edges. Generally, since Birkhoff, the chromatic polynomial, as function in variable λ was studied in the form where n is the number of vertices, χ is the chromatic number, ri(G) is the number of feasible partitions induced by colorings with exactly i colors, and λ (i) = (λ−1)(λ−2)...(λ−i+1).

Throughout the 20th century, there were many results about the roots of the chromatic polynomials of planar and other classes of graphs but all they miss the main goal: four color problem. Surprisingly, Birkhoff, Lewis and others, investing so many efforts in study of the roots of chromatic polynomials, did not even characterize the graphs for which all the roots are the integers from the set {1, 2, 3, ...χ − 1} (seems the simplest case!). As sometimes happens in mathematics, new ideas come from ”nowhere”: in 1955, Benzer discovered the linear structure of DNA molecule; motivated by that Hajnal and Suranyi in 1959 introduced and studied interval graphs which are a subclass of chordal graphs. Later, in 1975, it was discovered that chordal graphs have all the roots from the set {1, 2, 3, ...χ−1}. Dmitriev has found that not only chordal graphs have this property, so the ultimate form of the characterization is the following: a graph G is chordal if and only if for every induced subgraph G 0 all the roots of the chromatic polynomial are integers from the set {1, 2, 3, ...χ(G’) − 1}.

The history lesson however, is this: algebraic methods could not help to solve the four color problem but together with combinatorial methods produced many new fruitful scientific directions of research. Sometimes mathematics, and graph theory in particular, looks like a race for generalizations. In order to generalize graph coloring, Erdos and Hajnal in 1966 have introduced hypergraph colorings: the requirement ”adjacent vertices must have different colors” was generalized to ”at least two vertices in hyperedge must have different colors”. This idea was extremely fruitful and led to many generalizations of graph colorings and many hypergraph classes have been discovered. The special attention was paid to bipartite hypergraphs, normal hypergraphs (related to the weak Berge perfect graph conjecture) and extension of graph coloring to many set systems known long ago, like block designs etc. However, in all such generalizations the basic combinatorial problem was to find the chromatic number of a respective hypergraph, i.e. the minimum number of colors.

The problem of finding the maximum number of colors systematically never appeared because for any hypergraph on n vertices the coloring using n colors always existed. From this point of view, the classic coloring theory was the theory for finding the minimum only, i.e. it was evidently asymmetric. The end to this asymmetry was put in 1993 when the concept of mixed hypergraph was introduced (see the search on Google ”mixed hypergraph”). Mixed hypergraph is a triple H = (X,C, D) with vertex set X and two families of subsets, C and D, called C-edges and D-edges respectively. Proper coloring of H is a mapping from X into a set of λ colors in such a way that every C-edge has two vertices of the Common color and every D-edge has two vertices of Different colors. Now the very fundamental problem of colorability appeared: not every mixed hypergraph is colorable. The structure of uncolorable mixed hypergraphs is very general. The first asymptotical investigations created an opinion that coloring theory becomes the theory of uncolorable mixed hypergraphs, and colorability seems to be an island in a darkness.

Check whether a given graph is bipartite or not using C++

One approach is to check whether the graph is 2-colorable or not using [backtracking algorithm m coloring problem](https://www.geeksforgeeks.org/backttracking-set-5-m-coloring-problem/). In the c++ code, these steps are applied:

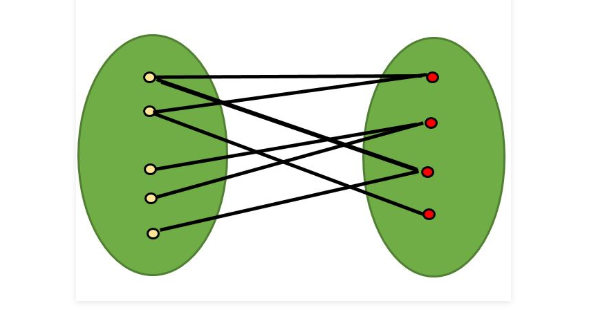
1. Assign RED color to the source vertex (putting into set U).

2. Color all the neighbors with BLUE color (putting into set V).

3. Color all neighbor’s neighbor with RED color (putting into set U).

4. This way, assign color to all vertices such that it satisfies all the constraints of m way coloring problem where m = 2.

5. While assigning colors, if we find a neighbor which is colored with the same color as current vertex, then the graph cannot be colored with 2 vertices (or graph is not Bipartite) Below is our C++ code for our bipartite graph and sample picture:



**Code:**

#include <iostream> //basic I/O

#include <queue>

#define V 4

using namespace std;

// This function returns true if graph

// G[V][V] is Bipartite, else false

bool isBipartite(int G[][V], int src)

{

// Creating a color array to store colors

// assigned to all vertices. Vertex

// number is used as an index in this array.

// The value '-1' of colorArr[i]

// is used to indicate that no color

// is assigned to vertex 'i'. The value 1

// is used to indicate first color

// is assigned and value 0 indicates

// second color is assigned.

int colorArr[V];

for (int i = 0; i < V; ++i)

colorArr[i] = -1;

// Assign first color to source

colorArr[src] = 1;

// Create a queue (FIFO first in first out) of vertex

// numbers and enqueue source vertex

// for BFS traversal

queue <int> q;

q.push(src);

// Run while there are vertices

// in queue (Similar to BFS)

while (!q.empty())

{

// Dequeue a vertex from queue

int u = q.front();

q.pop();

// Return false if there is a self-loop

if (G[u][u] == 1)

return false;

// Find all non-colored adjacent vertices

for (int v = 0; v < V; ++v)

{

// An edge from u to v exists and

// destination v is not colored

if (G[u][v] && colorArr[v] == -1)

{

// Assign alternate color to this adjacent v of u

colorArr[v] = 1 - colorArr[u];

q.push(v);

}

// An edge from u to v exists and destination

// v is colored with same color as u

else if (G[u][v] && colorArr[v] == colorArr[u])

return false;

}

}

// If we reach here, then all adjacent

// vertices can be colored with alternate color

return true;

}

// Driver program to test above function

int main()

{

int G[][V] = { {0, 1, 0, 1},

{1, 0, 1, 0},

{0, 1, 0, 1},

{1, 0, 1, 0}

};

isBipartite(G, 0) ? cout << "Yes, this graph is bipartite" : cout << "No, this graph is not bipartite"; // output message if it is or if it is not bipartite

cout << endl;

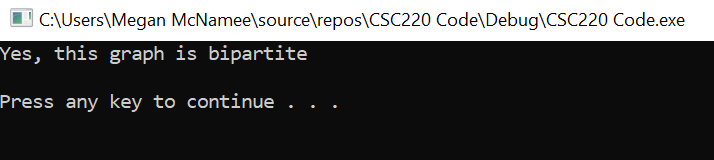
cout << endl;

system("pause");

return 0;

}

//Output Screen:



Like we said before, a graph coloring is a mathematical representation of a set of objects where some pairs of objects are connected to each other.  The sudoku puzzle is a popular game. Mine sudoku has 81 vertices, one for each cell and two vertices are connected by an edge if they cannot be assigned the same value.

Sudoku is closely related to graph theory as a Sudoku puzzle can be solved by considering it as a vertex coloring problem, which is the assignment of colors to the vertices of a graph in a way that no two adjacent vertices have the same color. Given a Sudoku graph where all cells are empty, a Sudoku graph Sud(n), where n 4 is the number of cells, can be established by one-to-one mapping from cells to vertices, adding edges between two vertices if they are in the same row, column or block. Solving a Sudoku puzzle is the same as coloring the whole graph with only n2 colors, e.g. the 4 × 4 Sudoku puzzle has n = 2, and it can be solved by filling the puzzle with numbers 1 to 4, i.e. coloring the whole graph with 22 = 4 colors.

The case of the Sudoku puzzle is to find a coloring using only 9 colors. Some algorithms behave better than others and it is usually a tradeoff between runtime complexity and the number of colors used. Currently not all puzzles can be solved; for some puzzles the algorithm cannot find a coloring that uses only nine colors.  An example of a puzzle that cannot be solved is given below.

**Code:**

#include <iostream>

using namespace std;

#define UNASSIGNED 0

#define N 9

bool FindUnassignedLocation(int grid[N][N],

int& row, int& col);

bool isSafe(int grid[N][N], int row,

int col, int num);

bool SolveSudoku(int grid[N][N])

{

int row, col;

if (!FindUnassignedLocation(grid, row, col))

return true;

for (int num = 1; num <= 9; num++)

{

if (isSafe(grid, row, col, num))

{

grid[row][col] = num;

if (SolveSudoku(grid))

return true;

grid[row][col] = UNASSIGNED;

}

}

return false;

}

bool FindUnassignedLocation(int grid[N][N],

int& row, int& col)

{

for (row = 0; row < N; row++)

for (col = 0; col < N; col++)

if (grid[row][col] == UNASSIGNED)

return true;

return false;

}

bool UsedInRow(int grid[N][N], int row, int num)

{

for (int col = 0; col < N; col++)

if (grid[row][col] == num)

return true;

return false;

}

bool UsedInCol(int grid[N][N], int col, int num)

{

for (int row = 0; row < N; row++)

if (grid[row][col] == num)

return true;

return false;

}

bool UsedInBox(int grid[N][N], int boxStartRow,

int boxStartCol, int num)

{

for (int row = 0; row < 3; row++)

for (int col = 0; col < 3; col++)

if (grid[row + boxStartRow]

[col + boxStartCol] == num)

return true;

return false;

}

bool isSafe(int grid[N][N], int row,

int col, int num)

{

return !UsedInRow(grid, row, num) &&

!UsedInCol(grid, col, num) &&

!UsedInBox(grid, row - row % 3,

col - col % 3, num) &&

grid[row][col] == UNASSIGNED;

}

void printGrid(int grid[N][N])

{

for (int row = 0; row < N; row++)

{

for (int col = 0; col < N; col++)

cout << grid[row][col] << " ";

cout << endl;

}

}

int main()

{

int a;

int grid[N][N] = { {0, 1, 0, 0, 5, 0, 8, 0, 9},

  {3, 6, 8, 0, 9, 0, 0, 4, 0 },

  {9, 0, 0, 0, 8, 0, 0, 2, 0 },

  {0, 4, 0, 5, 7, 0, 0, 0, 0 },

  {0, 0, 7, 0, 4, 0, 5, 0, 0 },

  {0, 0, 0, 0, 3, 2, 0, 6, 0 },

  {0, 9, 0, 0, 6, 0, 0, 0, 8 },

  {0, 5, 0, 0, 2, 0, 7, 3, 1 },

  {7, 0, 2, 0, 1, 0, 0, 9, 0 } };

if (SolveSudoku(grid) == true)

printGrid(grid);

else

cout << "No solution exists";

cout << "\nThats mean that our Sudoku looks like graph coloring like this: " << endl;

cout << "\n\t2 1 4 || 3 5 6 || 8 7 9" << endl;

cout << "\t3 6 8 || 2 9 7 || 1 4 5" << endl;

cout << "\t9 7 5 || 4 8 1 || 3 2 6" << endl;

cout << "\t======================="<<endl;

cout << "\t6 4 3 || 5 7 8 || 9 1 2" << endl;

cout << "\t1 2 7 || 6 4 9 || 5 8 3" << endl;

cout << "\t5 8 9 || 1 3 2 || 4 6 7" << endl;

cout << "\t=======================" << endl;

cout << "\t4 9 1 || 7 6 3 || 2 5 8" << endl;

cout << "\t8 5 6 || 9 2 4 || 7 3 1" << endl;

cout << "\t7 3 2 || 8 1 5 || 6 9 4" << endl;

return 0;

}

//Output:



In conclusion, graph coloring is a special case of graph labeling. Our goal was to further explore graph properties and vertex colorings. We have educated ourselves on the great history of graph coloring four color theorem and two color theorem, what they were trying to prove, applications of coloring and coding algorithms to demonstrate this case through C++ code. I am happy that I picked this topic to do research on cause it has exposed us to new ideals we weren’t fully aware of.