Using a Genetic Algorithm Approach to Solve the Chromatic Number Problem

Anand Kumar

Chimanbhai Patel P.G. Institute of Computer Applications, Gujarat University, Ahmedabad

kumaranandkumar@gmail.com

ABSTRACT

The least number of colors needed to color a graph G is called its chromatic number, $\chi(G)$. For example the chromatic number of a complete graph Kn of n vertices (a graph with an edge between every two vertices), is $\chi(Kn) = n$. A graph that can be assigned a (proper) k-coloring is k-colorable, and it is k-chromatic if its chromatic number is exactly k. This paper presents a genetic algorithm approach to solve the "Chromatic Number Problem". The "Chromatic Number Problem is NP complete problem and - one of the hardest problems in the class NP (non-deterministic polynomial problems). A genetic algorithm is simply the algorithm used to simulate evolution. It takes candidate solutions, selects some of the best using user-defined evaluation functions, applies user-defined transformations (often called mutation and crossover, but implementations of these depend on the problem), and makes new candidate solutions. Genetic Algorithms are being used extensively in optimization problem as an alternative to traditional heuristics. It is an appealing idea that the natural concepts of evolution may be borrowed for use as a computational optimization technique, which is based on the principle "Survival of the fittest" given by "Darvin". In this paper I have used genetic algorithm as an optimization technique to provide the solution for "Chromatic Number Problem". I have tried to show that genetic algorithm is an alternative solution for this NP hard problem where conventional deterministic methods are not able to provide the optimal solution.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Language Constructs and Features – *abstract data types, structure, Recursion, Linked List..*

General Terms

Algorithms.

KEY WORDS

Genetic Algorithm, Four colour map, graph, GA operators, NP-complete.

1 INTRODUCTION TO GENETIC ALGORITHM

Genetic algorithms (GA) is a powerful, robust search and optimization tool, which work on the natural concept of evolution, based on natural genetics and natural selection.. Its working principle is completely different, compared to classical search and optimization methods. Because of its broad applicability, ease of use, and global perspective, GAs have been increasingly applied to various search and optimization problems in the recent past. Genetic algorithms are typically implemented as a computer simulation in which a population of abstract representations (called chromosomes) of candidate solutions (called individuals) to an optimization problem evolves toward better solutions.

Traditionally, solutions are represented in binary as strings of 0s and 1s, but different encoding are also possible and in this paper I have used alphabets (A,B,C,D…) represent chromosomes. The evolution starts from a population of completely random individuals and happens in generations. This process is repeated generation by generation until the required result is not found. Most importantly of all, GAs are critically dependent on the fitness function. If we can't define the fitness function, then we can't apply a GA.

Work flow of GA

1. Initialization of parent population
2. Evaluation
3. Selection
4. Crossover/recombination
5. Mutation
6. Evaluate child and Go to step 3 until termination criteria satisfies

- **Initialization** of parent population means to generate the *M* number of solution string known as parent population which is mostly random
- **Evaluation** means give fitness to each of the solution. It is very important part of the algorithm based on the nature of problem and the requirement of the solution. It varies problem to problem.
- **Selection** means to select some of the best fit chromosomes from parent population according to some selection criteria (eg. Roulette wheel selection). Fit solution are likely to survive and bad solution are likely to die off
- **Crossover/Recombination** means to exchange partial solution between pair of selected solution with some probability
- **Mutation** means change the value of an allele of solution with some small probability value, it is a motivation to explore new point in the solution space

Simple Genetic Algorithm

```
{
    initialize population;
    evaluate population;
        while TerminationCriteriaNotSatisfied
          {
            select parents for reproduction;
            perform recombination and mutation;
            evaluate population;
          }
}
```

2 CHROMATIC NUMBER PROBLEM

The four color theorem (also known as the four color map theorem) states that given any plane separated into regions, such as a political map of the states of a country, the regions may be colored using no more than four colors in such a way that no two adjacent regions receive the same color. Two regions are called adjacent only if they share a border segment, not just a point. Each region must be contiguous: The four color theorem was the first major theorem to be proven using a computer, and the proof is not accepted by all mathematicians because it would be unfeasible for a human to verify by hand.

The "Chromatic Number Problem" in graph-coloring problem is an NP complete problem - one of the hardest problems in the class NP (non-deterministic polynomial problems).

The problem can be formalized as follows. An assignment of colors to a graph G = (V, E) is a mapping C: V → S, where S is a finite set of colors, (S == 4) which are normally represented as integers, such that if (v, w) € E then C(v) ≠ C (w); in other words, the same color is not assigned to adjacent vertices. Given a graph, the problem is to find the minimum number of colors needed for coloring the graph referred as the chromatic number of the given graph.

3 PROBLEM PRESENTATION

To solve the Chromatic Number Problem, I have considered a map of 10 regions (Fig 1.) where each region represents a separate country or state. The objective of this research is to use the concept of genetic algorithm to colour these entire region such that no two adjacent region have the same colour. The restriction is to use only four colours.

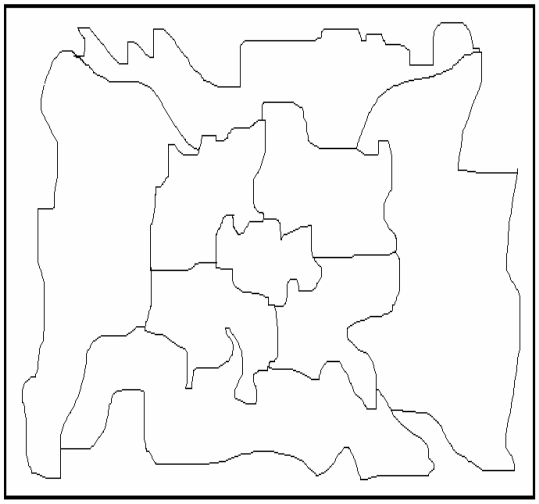

Figure:1

The entire regions have been marked with numbers to represent uniquely (Fig 2.) The region (10) represents the background of the map, which will also have different colour to visualize all the adjacent regions as 6,7,8 and 9 from the Fig 2.

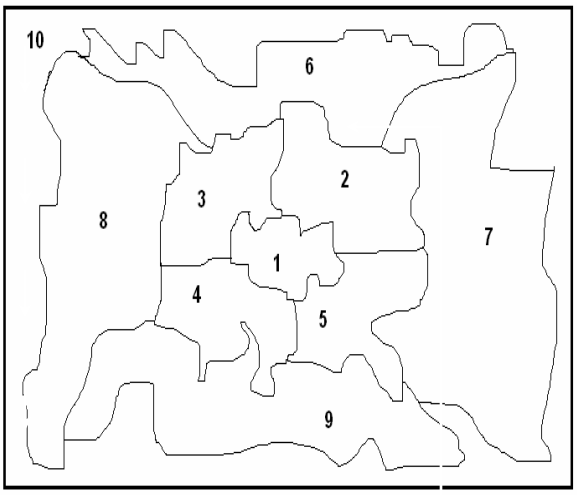

Figure: 2.

All these regions have connected to represent the problem in the form of a graph (Fig 3)

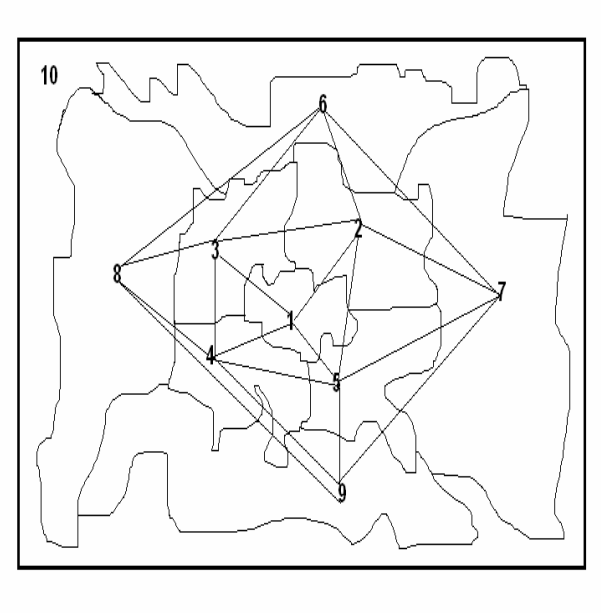

Figure: 3.

The N-regions are 9 nodes connected with others. The region (10) is not shown as the node because of to maintain the simplicity to represent the problem, but it is considered as a node in the implementation. Here each edge between to numbered region shows the adjacency of one region with other one. In (Fig 4). I have used a graph where each region is represented with the numbers 1, 2, 3,4,5,6,7,8,9 and 10. It is shown as ……

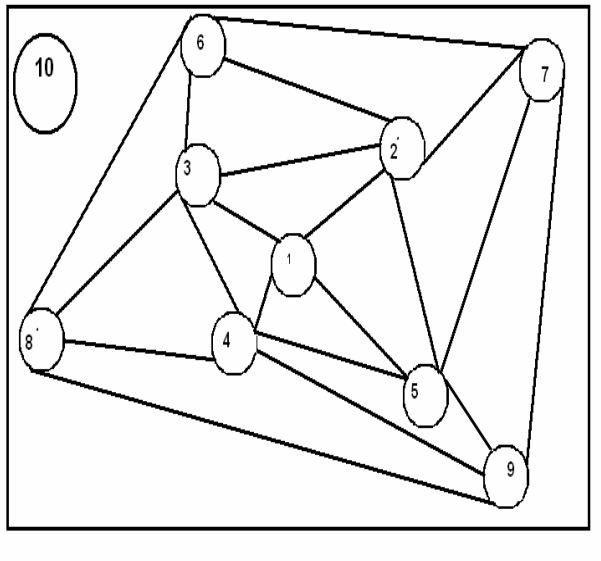

Figure: 4.

This graph is represented with the help of adjacency matrix representation in Table – 1(a). This table contains the value 0 and 1 where 0 represents no path between vertexes while 1 represent path between the two vertexes. Vertices are represented with the numbers 1 to 10 representing each region uniquely including background region 10. Table -1(b) contains the list of adjacent vertices for each of the nodes.

Table-1: **(a) ADJACENCY MATRIX OF THE GRAPH**

Node	1	2	3	4	5	6	7	8	9	10
1	0	1	1	1	1	0	0	0	0	0
2	1	0	1	0	1	1	1	0	0	0
3	1	1	0	1	0	1	0	1	0	0
4	1	0	1	0	1	0	0	1	1	0
5	1	1	0	1	0	0	1	0	1	0
6	0	1	1	0	0	0	1	1	0	1
7	0	1	0	0	1	1	0	0	1	1
8	0	0	1	1	0	1	0	0	1	1
9	0	0	0	1	1	0	1	1	0	1
10	0	0	0	0	0	1	1	1	1	0

(b) NODES WITH ADJACENT VERTEXES

Nodes	Adjacent vertices
1	2, 3, 4, 5
2	1, 3, 5, 6, 7
3	1, 2, 4, 6, 8
4	1, 3, 5, 8, 9
5	1, 2, 4, 7, 9
6	2, 3, 7, 8, 10
7	2, 5, 6, 9, 10
8	3, 4, 6 ,9, 10
9	4, 5, 7, 8, 10
10	6, 7, 8 ,9

4 GENETIC ALGORITHM APPROACH TO SOLVE THE

Problem Initialisation of parent population

Here 4 chromosomes a, b, c and d are generated randomly with the help of a function. It is called parent population. Each chromosome is represented with Alphabet string. Each chromosome is the combination of ten alphabets and each alphabet may be G, Y, S, and B only. These four alphabets represent four different colours

G- Green

Y- Yellow

S- Sky-blue

B- Blue

Each chromosome represents the colour for the node 1 to node 10.

node	1	2	3	4	5	6	7	8	9	10
a	G	S	Y	S	B	B	G	Y	S	Y
b	Y	S	B	G	Y	G	Y	B	G	S
c	G	Y	S	S	B	B	G	G	S	Y
d	G	Y	G	Y	S	B	B	G	Y	S

Evaluation

These four chromosomes are evaluated by a fitness function.

Fitness function

Fitness function is developed to check that how many vertexes have different colours with its adjacent vertices for each set of randomly generated chromosomes.

For each of the vertex (Table -1(b)), its colour is checked with its adjacent vertexes. If the colour of adjacent vertexes are not same as compared to the vertex for which it is compared, fitness is increased by 1. Let us consider the first randomly generated chromosome a,

a	G	S	Y	S	B	B	G	Y	S	Y

This chromosome 'a' represents the colour of node 1 is G, colour of node 2 is S and so on. Now we check the adjacent vertices of node 1, which are 2,3,4,5 from (Table- 1(b)).

Node (1)	2	3	4	5	Fitness
Colour (G)	S	Y	S	B	5

Now it found that colour of node (1) is not similar to any one of its adjacent vertices so its fitness will be 1. Similarly when we apply the same function for rest of the nodes from 2 to 10, we find that node (2), node (5), node (6) and node (7) have different colours with their adjacent vertices.

So fitness will be 5 for the chromosome a.

When the same function is applied to with all the rest chromosomes b, c, and d we find the following fitness..

Parent population	Fitness
a	5
b	3
c	7
d	3

Selection

In genetic algorithm fit solution are likely to survive and bad solution are likely to die off. So some of the best fit chromosomes are selected from parent population according to some selection criteria (e.g. Roulette wheel selection). Selected chromosomes are **a, b, c, and d.**

Crossover/Recombination

Selected solutions are used for crossover. I have considered 1 point cross over.

a	G	S	Y	S	B	B	G	Y	S	Y
b	Y	S	B	G	Y	G	Y	B	G	S

After crossover of a and b following child solutions x and y are found.

x	G	S	Y	S	B	G	Y	B	G	S
Y	Y	S	B	G	Y	B	G	Y	S	Y

Similarly c and d populations are used for crossover

C	G	Y	S	S	B	B	G	G	S	Y
D	G	Y	G	Y	S	B	B	G	Y	S

After crossover of c and d following child solutions p and q are found.

p	G	Y	S	S	B	B	B	G	Y	S
q	G	Y	G	Y	S	B	G	G	S	Y

Mutation

- Change the value of an allele of solution with some small probability value e.g. 1%
- Motivation is to explore new point in the solution space

x	G	S	Y	S	B	G	Y	B	G	S
y	Y	S	B	G	Y	B	G	Y	S	G
p	G	Y	S	S	B	B	B	G	Y	S

Q	G	Y	G	Y	S	B	G	G	S	Y

Evaluation of child population

After applying the fitness function we found the following fitness value for each of the child population

Parent population	Fitness
x	**10**
y	**4**
p	**5**
q	**5**

From the fitness value we found that child population x has the maximum fitness value 10 which is required. It means that all the nodes (regions) will have different colors which were the objective of this study.

If the required solution is not found then this child population replaces the parent population and the entire process is restarted. From selection phase.

After applying the colours recommended by chromosome **x, the resultant map will appear like..**

5 CONCLUSIONS

In this paper, I have proposed a simple Genetic Algorithm approach to solve the "Chromatic Number Problem" which is one of the hardest problems in the class NP (non-deterministic polynomial problems). I have described the problem and GA that encodes the candidate solution to the problem in a novel way. Each chromosome is a string of colours associated with the region of considered graph. The evolutionary approach of GA has shown to be a robust and efficient alternative for the " Chromatic Number Problem". The effectiveness of the methodology however can be increased by applying the various genetic operators and the densely connected regions.

Acknowledgements

My thanks to the reviewers for their cogent and insightful comments.

6 REFERENCES

[1] David E. Goldberg. 1989. Genetic Algorithms in search, optimization and machine learning. Addison-Wesley.0-201-15767-5

[2] Kalyanmoy Deb, 1995. Optimization for engineering design (PHI)

[3] Michael D. Vose. 1999. The simple genetic algorithm : (PHI)

[4] Mitchell, M. (1998). An Introduction to genetic Algorithm. MIT Press. 0-262-113316-4(HB)

Author's Biography

Anand Kumar received the B.Sc degree in Physics from B.R.A University, Muzaffarpur, India, the M.C.A degree from IGNOU, Delhi, India in 2001, the M.Phil degree in Computer Science from Madurai Kamaraj University in 2007 and currently pursuing Ph.D from Saurashtra University Rajkot, India. He is currently a Lecturer in Chimanbhai Patel P.G. Institute of Computer Applications, Gujarat University, Ahmedabad India. He is a senior member of IACSIT Singapore, Member International Association of Engineers Hong Kong and IEEE listed author. He has authored several papers in reputed international journals and International conference proceedings. He has been actively involved in the research of Genetic Algorithm and its implementation in various business domains including network design operation management and NP-hard problems. His current research interests include evolutionary algorithms and implementation in Decision support system and Communication network design.