Comparison of Genetic algorithm, Particle Swarm optimization and their hybrid for graph coloring problem.

Pankhudi Jain[#], Anik Bhattacharya[#], Anindya Jyoti Pal[#]

[#] *Heritage Institute of Technology*
Kolkata

[1]pankhudi.0804@gmail.com
[2] anikbhattacharya93@gmail.com
[3] aj.pal@heritageit.edu

Abstract— **In this paper, we have compared two evolutionary algorithms, namely Genetic algorithm and Particle Swarm Optimization along with their hybrid. We have used the heuristic algorithms and devised a hybrid to solve the graph coloring problem. Although a near optimal solution can be achieved using the conventional evolutionary algorithms, their hybrid provides us within proved results. We have tested the three algorithms on some benchmark graphs, namely the DIMACS, and presented a comparative study of the chromatic numbers obtained together with the time taken by each algorithm to run. The study shows that the hybrid algorithm performs better either by improving the near optimal solution achieved or by improving the time complexity of one or both of it's parent algorithms.**

Keywords— **graphs, Genetic Algorithm, Particle Swarm Optimization, DIMACS, graph color.**

I. Introduction

The graph coloring problem is a topic that has been the subject of intense research by computer scientists and mathematicians for decades. The problem requires us to color the vertices or nodes of a graph with the minimum number of color possible, subject to the constraint that no two adjacent vertices be colored with the same color.

What makes the graph coloring problem so intriguing is the fact that it is an NP complete problem [1]. By definition, NP complete are the problems which fall in the categories of NP and NP hard problems. The problems that lie in the intersection of NP and NP hard are said to fall under the NP complete category. [2] The difficulty while trying to solve these type of problems is that it is not known whether a solution of the problem exists or not. Also the time required to solve such problems increases rapidly with the size of problem. Hence higher the number of vertices of the graph more will be the time it will take to reach an optimum solution.

The increasing time complexity with the increase in problem size due to the NP completeness property of the problem makes heuristic algorithms a better solution for these kind of problems than deterministic algorithms. Although they produce near optimal solution (and not the optimum solution) in some cases, the reduced time complexity makes them attractive for obtaining minimum color required for graphs with large set of vertices and edges.

Graph coloring problem has many practical applications including, but not limited to [6] timetable scheduling [3], assignment of mobile radio frequency [4], map coloring [4] and register allocation [5].Due to the challenging nature of the problem and the wide range of applications it has, we decided to hybridize two heuristic algorithms that not only finds a near optimal solution but also takes minimal amount of time to do that.

Evolutionary algorithms are population based heuristic algorithms which generate random probable solutions, and then modifies that to reach an optimum value. Usually, evolutionary algorithms use techniques which are inspired from the biological concepts of evolution, and uses techniques like population generation, selection, recombination, mutation etc. to generate a fitter population which is analogous to the nearest optimal result possible.

Let us take an undirected graph G= (V, E). The vertices v are partitioned into k-subsets $C_{i,}$ i= {1…k) such that no two adjacent vertices have the same color, i.e., are in the same subset. The graph coloring problem is to find the smallest value of k possible. The smallest possible value of k is known as the chromatic number $\chi(g)$ of graph G. If two adjacent vertices a and bare colored with the same color, vertices a and b are said to be conflicting vertices, the edge {a, b} is called a bad edge, and x is called a conflicting color. The k-coloring of a graph is said to be legal if there are no bad edges. The graph coloring problem is to determine the smallest value of k for which we will find a legal k-coloring of G.

Both Genetic algorithm (GA) and Particle Swarm Optimization (PSO) are used in function optimization and in this paper we have first implemented them to get near optimal solutions for the graph coloring problem. We then devised a hybrid algorithm and observed that the hybrid performs relatively better than the generic algorithms.

II. Previous work

As already mentioned, the graph coloring problem has intrigued computer scientists from all over the world. Hence over the years a lot of work has been done in trying to solve the problem for different graphs. We have discussed in brief the works that have included GA and PSO to solve the graph coloring problem.

A. *Genetic Algorithm*

Biman Ray, Anindya J Pal, Debnath Bhattacharyya and Tai-hoon Kim [6], have implemented genetic algorithm using a special kind of mutation known as multipoint guided mutation. The algorithm, using this technique turns out to be more efficient than the simpler GA. Their work was published in 2010 and all the earlier work on genetic algorithm to solve graph coloring problem is discussed there. The results given in this paper clearly show marked improvements in reaching a near optimal solution. They have tested their results on the same benchmark DIMACS graphs which help establish uniformity. A new operator is introduced in this paper which is called double point Guided mutation and the algorithm that is proposed here changes the performance of the conventional GA significantly. The basic job of the new mutation operator is to reduce colors from multiple locations. Two mutation operators are used here. One, the conventional mutation that does not guarantee an improved solution but helps the solution to get out of local minima and second, the multi point guided mutation that aims at reducing the number of color classes. The main aim of their work was to get a solution to the complex graphs in reasonable time and they have been successful in doing so.

Musa M. Hindi and Roman V. Yampolskiy [7] in 2012 have incorporated the theory of Wisdom of artificial crowd when the result of genetic algorithm does not improve for a fixed number of iterations. They have used two mutations and crossover operators. They use the operators depending on the chromatic numbers of the chromosome. Artificial crowd is being used here only in that situation when after a fixed number of iterations the algorithm doesn't reach a solution then in that case this technique is used. The technique is basically used to decide and remove the bad edges so that a legal solution is reached. The algorithms in this paper were tested on DIMAC benchmark graphs and the paper was concluded by showing better results for a few more graphs than the earlier work. However, results of implementing the algorithms on bigger graphs were not discussed.

B. *Particle Swarm Optimisation*

Not much work has been done in solving graph coloring problem using particle swarm, though we have discussed the few important works done earlier.

Ling-Yuan Hsu, Shi-Jinn Horng , Pingzhi Fan , Muhammad Khurram Khan,Yuh-Rau Wang ,Ray-Shine Run,Jui-Lin Lai, Rong-Jian Chen. [8] in 2010 proposed a new way of implementing particle swarm optimization and they have called it "modified turbulent particle swarm optimization" Although they have implemented it on planar graphs with very few chromatic numbers, their approach can be used on bigger and more complicated graphs like DIMACS graph.

All the significant works done prior to their paper have been discussed in details by them. This paper has implemented it on planar graph using 4 colors, and the result they get is quite accurate and efficient. The new algorithm developed by them uses three strategies which improves the results significantly over the old PSO algorithm. The strategies are walking one strategy, assessment strategy and turbulent strategy. They are used to find out the conflict (illegal colors or bad edges) of each particle, determining the one with the maximum and minimum conflict, which aids in deciding the velocity for the movement of each particle respectively. All this is determined based on some conditions. These three strategies are combined in the PSO algorithm to get the position and velocity of the particles. This technique proves to be more efficient than the generalized PSO.

Jin Qin , Yi-xin Yin and Xiao-juan Ban [9] in their work in 2011 attempted to solve the graph coloring problem by redefining the PSO operators. The main concept introduced in the paper is that of distance instead of discrete space. By using this logic, position and velocity are reinterpreted. To solve the graph coloring problem and to get the near optimal result, they have designed an algorithm that implements the PSO operator, which attempts to find the difference in position of the two particles, or the distance between them as we may call it. Apart from this, a local search is also combined with the algorithm to produce a hybrid version of the algorithm.
Although they haven't tested it on all the DIMACS, the ones they have tried on works properly and the results obtained are competitive enough as compared to other algorithms.

We have gone through quite a few papers that have been published in various journals and conference but none of them till now have implemented the 2 algorithms that we are using and their hybridized version on one platform and compared the results of all of them, even though a lot of work on these algorithms individually has been done previously. Hence we decided to implement them and later combine the two to produce a hybrid, the results of which have been compared and presented in this paper.

III. Evolutionary Algorithm

Evolutionary algorithms have been used in artificial intelligence to solve complex problems. They are used to design computational techniques to solve optimization problems; they do this mainly by imitating the biological process of Evolution. A brief description of the Algorithms used is given below.

A. Genetic Algorithm

It is one of the most well known evolutionary algorithms whose working principle is inspired from the fundamental concepts of genetics, i.e., selection, mutation, crossover etc. First of all, in Genetic algorithm (GA), we need to encode the probable solution to a specific problem in a chromosome like structure. A pool of such chromosomes is created, and their respective fitness is calculated. The main aim after that is to improve the chromosome through crossover and mutation and take the fittest population to the next generation.

For solving the Graph coloring problem, we have implemented the algorithm by initializing chromosomes of the same length as the number of vertices in the graph. Values in these chromosomes will be randomly assigned colors to each vertex. For example, if there is a graph with 6 vertices, a sample chromosome will look like the following:

1	5	3	2	4	6

Fig 1: Chromosome Example

After generating a specific number of such chromosomes, their validity is checked, i.e., it is made sure that no two adjacent vertices of the graph have the same color, and if any such conflict is found where the chromosomes are not valid, the conflicting colors in one of the vertices is replaced by a suitable color.

Once the valid pool is generated, functions like mutation and crossover to generate a fitter population is applied to the chromosome. Fitness in this case is calculated by the number of distinct colors used in the chromosome. For example, in the figure mentioned above, the fitness of that chromosome are 6.

A special type of mutation is used in this case. It is called multi-point special mutation [6]. During mutation, we try to reduce the number of color classes by replacing one particular color each time at two or more places by one of the other colors used to color the vertices [6]. We then check whether the new chromosome generated is valid or not. The new chromosome is introduced into the pool of chromosomes if it is a valid one; otherwise the process is repeated again.

Crossover used here is a simple one; a two point cross over method is used. Two chromosomes are selected, and a random point is chosen. Information after that point is interchanged to generate two new chromosomes. Their fitness is calculated and they are included in the pool. The algorithm to explain how the graph coloring problem was solved is given below.

INPUT: A simple graph represented by an adjacency list or an adjacency matrix.

Step 1) Create the pool of initial chromosomes. A chromosome is created by randomly assigning colors to the vertices of the graph.

Step 2) Predetermine crossover and mutation probability.

Step 3) Calculate the validity of such chromosomes to make sure the chromosomes have a valid coloring by eliminating bad edges.

Step 4) Calculate the fitness of each of the valid chromosome.

Step 5) Continue till the termination condition is met.

- Step 5.1) Perform Crossover with a probability.
- Step 5.2) Ensure validity of chromosome by removing bad edges.
- Step 5.3) Calculate the fitness of the newly generated chromosomes and add them in the pool.
- Step 5.4) Perform mutation with a probability.
  - Step 5.4.1) Either perform an extensive mutation by randomly selecting a color class and replacing it with other colors for all vertices that it was previous colored with
  - Step 5.4.2) Or perform a normal mutation by selecting a single vertex and recolor it with a different color to ensure that the solution is not stuck at a local minima.
- Step 5.5) Ensure validity of new chromosome by removing bad edges.
- Step 5.6) Calculate the fitness and introduce new chromosome in the pool.
- Step 5.7) Sort the chromosome pool according to the fitness and select the best few from the pool for the next generation.
- Step 5.8) Continue till the fitness value of the best possible solution remains unchanged for a significant number of iterations to find and optimal solution.

B. Particle Swarm Optimisation

Particle swarm optimization is an algorithm that works best on numerical optimization problem. A fixed number of particles are assigned their respective positions initially and then made to travel around the solution space with a certain velocity till the best solution is found.

For graph coloring problem, we initialize a fixed number of particles; say 20, with different positions and velocity. Then we check the validity of the particle by making sure no two adjacent vertices has the same color. If such a situation exists, the color of one such conflicting vertex is replaced with a suitable color. The particles in this case are arrays with the same length as the number of vertices, and their positions will be the random colors assigned to each vertex for each individual particle. The fitness of each particle is calculated based on the number of independent colors used. The Global best (Gbest) will the value of the fittest particle. Velocities of the particles are calculated based on the difference of current global best and previous global best of the particles. To improve the solution, we remove the same number of unique color classes from the particle as the velocity and ensure the validity of the particles again. This continues for a fixed number of iteration.

The algorithm is explained below:-

```
INPUT: Total no. of vertices and vertices that create edges from file and create adjacency matrix of the graph with number of vertices

Step 1) Create a set of distinct particles with different positions in the solution space by assigning random colors to the vertices for each particle.
Step 2) Ensure the validity of all the particles after calculating the fitness of each chromosomes.
Step 3) Calculate the personal best of all the particles based on the distinct colors used.
Step 4) Calculate the global best (GBest) from the personal bests of all the particles.
Step 4) Repeat till the termination condition is reached.
    Step 4.1) Determine the velocity with which the particles need move
    Step 4.2) For each particle, change its position in the solution space depending on the velocity. This is done by removing as many numbers of colors in each particle as the current velocity of the particles.
    Step 4.3) Ensure all the particles are valid.
    Step 4.4) Calculate their personal best and find the new global best (GBest).
    Step 4.5) Continue till the value of GBest remains unchanged for a significant number of iterations to find an optimal solution.
```

C. Hybrid Algorithm

The hybrid algorithm devised is a crossbreed of particle swarm and genetic algorithm. The two algorithms are combined by creating a pool of chromosomes and another pool of particles at random positions. PSO is then performed on the particles several number of times, and the particle with GBEST is introduced in the pool of chromosomes each time which is then operated by GA.

PSO is performed instead of the extensive mutation operation of the genetic algorithm. Apart from that one change, the genetic algorithm is allowed to run uninterrupted and every time mutation is needed to be performed, we use PSO and then introduce the particle with the GBest value inside the pool of chromosomes.

The algorithm is given below –

```
INPUT: A simple graph represented by an adjacency list or an adjacency matrix.

Step 1) Create an initial pool of chromosomes. A chromosome will be a random assignment of colors to the vertices of the graph.
Step 2) Ensure the validity of all the chromosomes by eliminating bad edges
Step 3) Introduce a certain number of particles into the solution space. A particle is introduced or created by assigning random coloration to the vertices of the graph. It is analogous to a chromosome.
Step 4) Ensure the validity of all the particles.
Step 5) Continue till the termination condition is reached.
    Step 5.1) Perform Crossover with a probability.
    Step 5.2) Ensure validity of chromosome by eliminating bad edges.
    Step 5.3) Calculate the fitness of the newly generated chromosomes and add them in the pool if their fitness is better than the fitness of some of the chromosomes in the existing pool.
    Step 5.4) Perform Mutation or PSO with a probability.
        Step 5.4.1) Either perform PSO on the particles instead of extensive Mutation (This is performed only when the extensive mutation was supposed to be performed with a probability).The particle with the GBEST value is introduced in the pool of chromosomes.
        Step 5.4.2) Or perform normal mutation by selecting a single vertex and replacing its color with a different color to ensure solution is not stuck at the local minima.
    Step 5.5) Sort the chromosomes according to the fitness and choose the first few for the next generation.
    Step 5.6) Continue till the fitness value of the best possible solution remains unchanged for a significant number of iterations.
```

IV. RESULTS AND DISCUSSIONS

We have tested the algorithms on DIMACS graphs [10]. They are benchmark graphs on which testing of graph algorithms can be done.

DIMACS sponsor challenges to determine the performances of algorithms on several problems like graph coloring, travelling salesman, shortest path problem and the like.

We coded our algorithms in ANSI C and the OS we used were FEDORA 21. The processor that was used was 1.4 GHz Intel core i5. As for the results we obtained, we can see that the hybrid algorithm has provided a solution which is very close to the actual chromatic number for majority of the graphs. We have calculated the time needed by the respective algorithms and the chromatic number for these graphs that are generated by these algorithms and we have made the following observations:

1) In terms of the minimum number of colors required to color a graph, or the chromatic number of the graph, the hybrid algorithm, when implemented to solve the graph coloring problem on the graphs mentioned in the table below, performs better than GA 60% of the time and better than PSO 32% of the time.

2) In terms of the time taken to reach the optimal solution for the respective graphs, the hybrid algorithm when implemented to solve the graph coloring problem on the graphs mentioned in the table below, performs better than PSO 68% of the time. The hybrid algorithm however takes more time than GA in all cases expect for one where it takes equal amount of time.

3) Depending on the nature of the graph, the efficiency of the algorithms varies. For those belonging to the category of myciel, the minimal solution is reached but for those belonging to the category of queen and mul_sol, all the algorithms have reached a near optimal solution and not the minimal one.

The results table of the chromatic number of the graphs and the results obtained is given below.

TABLE I
COMPARISION OF THE CHROMATIC NUMBERS

Name of the Graph	*Chromatic number*	*PSO (time is sec)*	*GA (time in sec)*	*Hybrid (time in sec)*
fpsol2.i.3	30	31 (127.26)	32(27.77)	31 (189.77)
le450_25b	25	28 (110.00)	28(12.50)	26 (133.65)
le450_5d	5	18 (124.54)	20(26.70)	17 (148.64)
mulsol.i.1	49	49 (18.18)	53(1.51)	49 (26.16)
mulsol.i.2	31	33 (21.69)	38(1.18)	33 (16.82)
mulsol.i.3	31	33 (22.56)	37(1.93)	33 (19.68)
mulsol.i.4	31	33 (18.11)	37 (1.36)	33 (13.39)
mulsol.i.5	31	33 (23.66)	36(1.60)	32 (23.26)
zeroin.i.1	49	49 (16.78)	50(2.54)	49 (16.48)
zeroin.i.2	30	31 (17.14)	32(4.41)	31 (18.42)
zeroin.i.3	30	31 (15.10)	30(1.46)	30 (19.76)
games120	9	9 (4.75)	9(0.45)	9 (3.23)
miles1000	42	45 (42.15)	45(1.46)	44 (8.68)
miles1500	73	74 (102.76)	73(1.45)	74 (7.80)
miles500	20	21 (9.36)	22(0.95)	21 (4.79)
miles750	31	34 (9.70)	33(0.74)	33 (8.05)
queen5	5	8 (1.52)	8(0.02)	8 (0.21)
queen6	7	9 (1.63)	9(0.04)	9 (0.32)
queen7	7	11 (0.88)	11(0.08)	10 (0.37)
queen9	10	13 (2.84)	13(0.35)	13 (2.26)
queen8_12	12	15 (3.01)	15(0.45)	15 (2.00)
queen13	13	21 (17.91)	24(2.24)	20 (38.98)
myciel3	4	4 (0.02)	4(0.01)	4 (0.01)
myciel6	7	7 (2.43)	7(1.83)	7 (1.90)

The graphical representations are given below:-

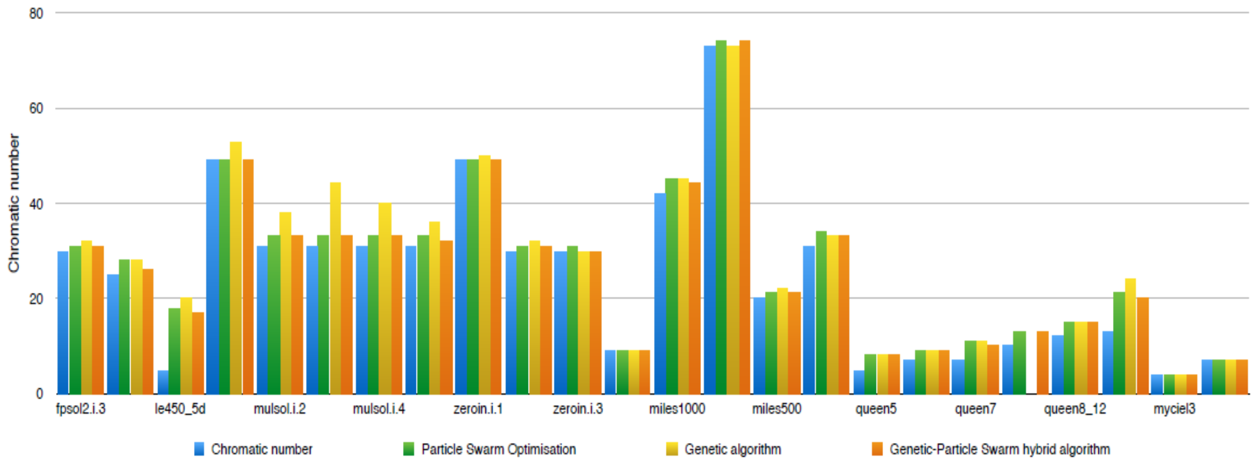

Fig. 1 Graph to show the chromatic numbers obtained

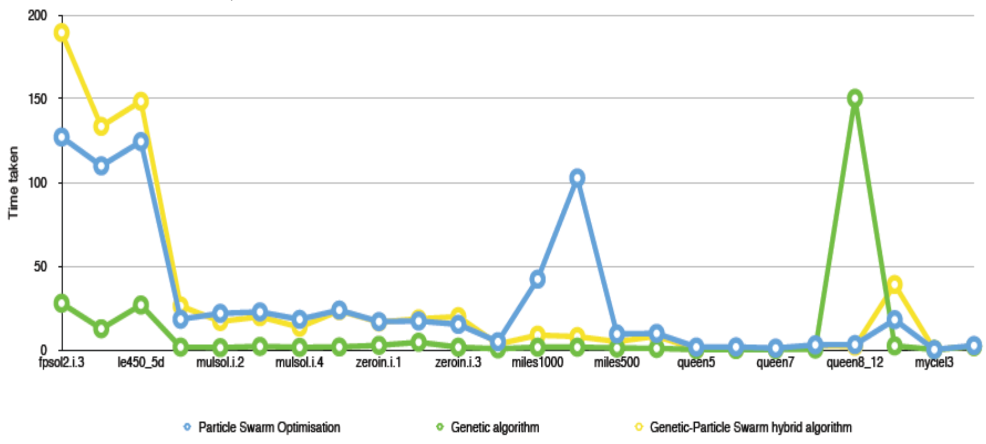

Fig. 2 Graph to show the time taken by the three algorithms

V. Conclusions

In this paper we have tried to generate a hybrid algorithm by combining two of the most well known heuristic algorithms .i.e. Genetic and Particle Swarm . The hybrid algorithm performs better than two original algorithms,

Acknowledgment

The authors gratefully acknowledge the works of all the other researchers whose works have been mentioned earlier. They have provided us with many useful information and tips to complete our work.

References

[1] Gilles Brassard, Paul Bratley, Algorithmics: Theory and Practice. Englewood Cliffs, New Jersey : Prentice hall, 1989.
[2] https://en.wikipedia.org/wiki/NP-complete /
[3] D. C. Wood, "A technique for coloring a graph applicable to large scale time-tabling problems", Computer Journal, vol. 12,1969, pp. 317-319.
[4] http://www.geeksforgeeks.org/graph-coloring-applications/
[5] F. C. Chow and J. L. Hennessy, "Register allocation by priority based coloring", Proceedings of the ACM Sigplan 84 symposium on compiler construction New York,1984, pp. 222-232.
[6] Biman Ray, Anindya J Pal, Debnath Bhattacharyya , and Tai-Hoon Kim, "An efficient ga with multipoint guided mutation for graph coloring problems", International Journal of Signal Processing, Image Processing and Pattern Recognition 3, no. 2 ,2010,: pp. 51-58.
[7] Musa M. Hindi, and Roman V. Yampolskiy , "Genetic Algorithm Applied to the Graph Coloring Problem", Proc. 23rd Midwest Artificial Intelligence and Cognitive Science Conf, 2012, pp. 61-66.
[8] Hsu, Ling-Yuan, Shi-Jinn Horng, Pingzhi Fan, Muhammad Khurram Khan, Yuh-Rau Wang, Ray-Shine Run, Jui-Lin Lai, and Rong-Jian Chen, "MTPSO algorithm for solving planar graph coloring problem.", *Expert systems with Applications* 38, no. 5 , pp: 5525-5531,2011, 2010.
[9] Qin, Jin, Yi-xin Yin, and Xiao-juan Ban, "Hybrid discrete particle swarm algorithm for graph coloring problem." *Journal of Computers* 6, no. 6,pp: 1175-1182,2011.
[10] http://mat.gsia.cmu.edu/COLOR/ins tances.html#XXSGB