Int. Journal of Math. Analysis, Vol. 4, 2010, no. 50, 2473 - 2479

Coloring Programs in Graph Theory

[1]Akhlak Mansuri, [2]Vijay Gupta and [3]R. S. Chandel

[1]Department of Applied Science
Kailash Chandra Bansal College of Technology
Bhopal, (M.P.) India
mansuri_akhlaak@yahoo.com

[2] Department of Applied Mathematics, UIT, RGPV, Bhopal (M.P.) India
vkgupta_12873@rediffmail.com

[3]Department of Mathematics, Govt. Geetanjali Girls College, Bhopal (M.P.) India
rs_chandel2009@yahoo.co.in

Abstract. Purpose of this paper to introduce the two new heuristic graph coloring programs which are based on known heuristic algorithms, have already been introduced. First one is adaptation of the Largest Degree Ordering algorithm, and second one is a adaptation of the Saturation Degree Ordering algorithm. These two new programs planned in this paper, is compared practically with a few of the known heuristic graph coloring algorithms such as; Largest Degree Ordering, First Fit, Saturated Degree Ordering and Incident Degree Ordering. In this comparison, for the number of used colors, the result is found that the proposed programs are better than the original results.

Keywords: Graph coloring algorithms, degree based ordering, first-fit algorithm.

INTRODUCTION:

A graph $G = (V, E)$ consists two sets where one is the set of vertices and another is the set of edges such that each edges is associated with an un ordered pair of vertices and graph coloring is one of the most useful models in graph theory. Graph coloring is the way of coloring the vertices of a graph with the minimum number of colors such that no two adjacent vertices share the same color. For example binary search tree needs two colors. There is a wide application of graph coloring such as; registration allocation, pattern matching, time tabling and scheduling, frequency assignment, numerical computation, and estimation of sparse Jacobins.

The coloring of a graph $G = (V, E)$ is a mapping $c: v \rightarrow f$, where "f" is a finite set of colors, such that if $v_1, v_2 \varepsilon E$ then $c(v_1) \neq c(v_2)$. In other words, adjacent vertices are not assigned the same color. A coloring using at most k-colors is called a (proper) k-coloring. The smallest number of colors needed to color a graph G is called a chromatic number, $\chi(G)$. A graph can be assigned a (proper) k-coloring is k-colorable *i.e.* $\chi(G) \leq k$, and it is a k-chromatic if its chromatic number is exactly k *i.e.*. $\chi(G) = k$.

In this paper the two new heuristic graph coloring programs which is based on known heuristic algorithms, have already been introduced. First one is an adaptation of the Largest Degree Ordering algorithm, and second one is an adaptation of the Saturation Degree Ordering algorithm. These two new programs planned in this paper, is compared practically with a few of the known heuristic graph coloring algorithms such as; Largest Degree Ordering, First Fit, Saturated Degree Ordering and Incident Degree Ordering. In this comparison, for the number of used colors, the result is found that the proposed algorithms are better than the original results.

Graph Coloring Algorithms: There are many heuristic sequential techniques for coloring a graph. One of them is the Greedy Graph Coloring. Greedy coloring heuristics build a coloring by repeatedly extending a partial coloring of the graph. A graph is said to be partially colored if a subset of its vertices is validly colored. Greedy coloring heuristics concentrate on carefully picking the next vertex to color and the color for that vertex. In these heuristics, once a vertex is colored, its color never changes. For graphs arising from a number of applications, it has been demonstrated that these heuristics are often able to find colorings that are within small additive constants of the optimal coloring [2, 5]. Below, we explain the first fit and degree based ordering techniques.

First Fit: The First Fit coloring algorithm is fed the set of vertices in some arbitrary order. The algorithm sequentially assigns each vertex the lowest legal color. First Fit has the advantage of being very simple and very fast. In other words, First Fit is an O(n)-time algorithm.

Degree based ordering: A better strategy than simply picking the next vertex from an arbitrary order is to use a certain selection criterion for choosing the vertex to be colored among the currently uncolored vertices. Such a strategy, depending on the nature of the selection criterion, has a potential for providing a better coloring than First Fit. In the following paragraphs we discuss some of the well-known, effective selection strategies.

Largest Degree Ordering: Ordering the vertices by decreasing degree, proposed by C. Avanthay, A. Hertz, N. Zufferey [1], was one of the earliest ordering strategies. This ordering works as follows. Suppose the vertices $v_1, v_2, \ldots, v_{i-1}$ have been chosen and colored. Vertex v_i is chosen to be the vertex with the maximum degree among the set of uncolored vertices. Intuitively, Largest Degree Ordering provides a better coloring than First Fit since during each iteration it chooses a

vertex with the highest number of neighbors which potentially produces the highest color. Note that this heuristic can be implemented to run in $O(n^2)$.

Saturation Degree Ordering: Saturated degree ordering was proposed by E. Falkenauer [4] and is defined as follows. Suppose that vertices $v_1, v_2,\ldots,v_{i-1}$ have been chosen and colored. Then at step i, vertex v_i with the maximum Saturation degree is selected. The saturation degree of a vertex is defined as the number of differently colored vertices the vertex is adjacent to. For example, if a vertex v has degree equal to 4 where one of its neighbors is uncolored, two of them are colored with color equal to 1, while the last one is colored with color equal to 3, then v has saturation degree equal to 2. While choosing a vertex of maximum saturation degree, ties are broken in favor of the vertex with the largest degree. Intuitively, this heuristic provides a better coloring than Largest Degree Ordering since it first colors vertices most constrained by previous color choices. The heuristic can be implemented to run in $O(n^2)$ E. Falkenauer [4] .

Incident Degree Ordering: A adaptation of the Saturated Degree Ordering heuristic is the Incident Degree Ordering introduced by E.K. Burke, B. McCollum, A. Meisels, S. Petrovic, R. [5] This ordering is defined as follows. Suppose that vertices $v_1, v_2,\ldots,v_{i-1}$ have been chosen and colored. Vertex v_i is chosen to be a vertex whose degree is a maximum in the subgraph of G induced by the vertex set $\{v_1, v_2,\ldots.v_{i-1}\}\cup\{v_i\}$. In other words, a vertex v_i with the maximum Incident degree is chosen at step i. The Incident degree of a vertex is defined as the number of its adjacent colored vertices. Note that it is the number of adjacent colored vertices and not the number of colors used by the vertices that is counted. The vertex v in the example of the above subsection has Incident degree equal to 3.Tie-breaking in Incident Degree Ordering is as in the case of Saturated Degree Ordering. The Incident Degree Ordering vertex ordering has the advantage that its running time is a linear function of the number of edges [8]. In other words, Incident Degree Ordering is an O(n)-time algorithm.

Proposed Programs: In this study, two new heuristic graph coloring Programs have been introduced based on known algorithms. These two proposed new programs are described in the following sections.

Proposed Program 1: In this program, we modified the Largest Degree Ordering algorithm by combining it with the Incident Degree Ordering. The program works as Largest Degree Ordering but when we found that there are two nodes having the same degree, the Incident Degree Ordering was used to choose between them. There are two criteria for chosen the vertex to be colored:

- The number of vertices connected to the vertex Largest Degree Ordering.
- The number of colored vertices connected to the vertex Incident Degree Ordering.

```
# include <iostream.h>
#include <conio.h>
void main ( )
```

```
{
Int n, i, I, d, large, count,  No. of colored Vertices=0, colored[500], a[500], ID[500];
cout<< "Enter the number of Vertices";
cin>> n;
cout<< "Enter the degree of each Vertex";

for(i=1; i<=n; i++ )
{
cin>>a[i];
}

While( No. of colored Vertices<n)

{
large=-1;
for( I=1; I<=n; I++)
{
If (!colored[I])
{
d=a[I];
if(d>large)
{
large=d;
count =I;
}
if(d=large)
if(ID[I]>ID[count])
count =I;

color(count);
No. of colored Vertices++;
}
cout<<"The No. of Colored Vertices";
cout<<No. of colred Vertices;
}

}
```

This heuristic can be implemented to run in $O(n^2)$ as Largest Degre Ordering.

Proposed Program 2: In this program, we combine both the Saturated Degree Ordering, and The Largest Degree Ordering .The program works as the Saturated Degree Ordering, but when

there are two nodes having the same degree, we use the Largest Degree Ordering to choose between them. So there are two criteria to choose the next node to be colored:

- The number of colors surrounding the vertex, Saturated Degree Ordering.

- The number of vertices surrounding the vertex, Largest Degree Ordering.

```
# include <iostream.h>
#include <conio.h>
void main ( )

{
Int n, i, I, d, large, count,  No. of colored Vertices=0, colored[500], SD[500], degree[500];
cout<< “Enter the number of Vertices”;
cin>> n;
cout<< “Enter the degree of each Vertex”;

for(i=1; i<=n; i++ )
{
cin>>SD[i];
}

While( No. of colored Vertices<n)

{
large=-1;
for( I=1; I<=n; I++)
{
If (!colored[I])
{
d=SD[I];
if(d>large)
{
large=d;
count =I;
}
if(d=large)
if(degree[I]>degree[count])
count =I;
}
color(count);
No. of colored Vertices++;
}
cout<<”The No. of Colored Vertices are”;
cout<<No. of colored Vertices;
}

}
```

This heuristic can be implemented to run in $O(n^3)$ as Saturated Degree Ordering.

RESULTS:

We implemented four heuristic graph coloring First Fit, Largest Degree Ordering, Incident Degree Ordering, Saturated Degree Ordering in addition to the two proposed programs. We also used random graphs that have different number of vertices and densities. The results show the First Fit algorithm is the worst algorithm with respect to the number of colors used. Largest Degree Ordering uses smaller number of colors than Incident Degree Ordering and Saturated Degree Ordering uses smaller number of colors than Incident Degree Ordering and Largest Degree Ordering. The proposed **Program 1** (adaptation of Largest Degree Ordering) is better than Largest Degree Ordering in the number of colors used. The proposed **Program 2** (adaptation of Saturated Degree Ordering) is better than Saturated Degree Ordering in the number of colors used. The proposed **Program 2** uses the least number of colors for coloring a graph.

CONCLUSION:

In this paper, we introduced two new heuristic graph coloring programs based on modifying and improving known pervious ones. One of them is an improved modification on Largest Degree Ordering by combining it with the Incident Degree Ordering, and the other one is an improved modification of Saturated Degree Ordering by combining it with the Largest Degree
Ordering. It has been empirically shown that the number of colors used in the proposed program is better than the original.

ACKNOWLEDGEMENT:

I am highly thankful to **Dr. Vijay Gupta** Reader, Deptt. of Mathematics UIT, RGPV (INDIA) & **Dr. R. S. Chandel**, Asst. Professor, Department of Mathematics, GGGC, Bhopal (INDIA) for his kind guidance and support in writing this paper. I am also thankful to **Dr. H. B. Khurasia Sir**, Director KCBCT, Bhopal (INDIA) for his support in the institute.

REFERENCES

[1] **C. Avanthay, A. Hertz, N. Zufferey**, A variable neighborhood search for graph coloring, European Journal of Operational Research 151 (2) (2003) 379–388.

[2] **D. Brélaz**, New methods to color the vertices of a graph, Communications of the ACM 22 (4)(1979) 251–256.

[3] **D. de Werra, C. Eisenbeis, S. Lelait, B. Marmol**, On a graph-theoretical model for cyclic register allocation, Discrete Applied Mathematics 93 (2–3) (1999) 191–203

[4] **E. Falkenauer**, A hybrid grouping genetic algorithm for bin packing, Journal of Heuristics 2 (1) (1996) 5–30.

[5] **E.K. Burke, B. McCollum, A. Meisels, S. Petrovic, R. Qu**, A graph-based hyper heuristic for timetabling problems, European Journal of Operational Research 176 (2007) 177–192.

[6] **I. Blöchliger, N. Zufferey**, A graph coloring heuristic using partial solutions and a reactive tabu scheme, Computers and Operations Research 35 (3) (2008) 960–975.

[7] **M. Chams, A. Hertz, D. de Werra**, Some experiments with simulated annealing for coloring graphs, European Journal of Operational Research 32 (1987) 260–266.

[8] **M. Chiarandini, T. Stützle**, An application of iterated local search to graph coloring, in: D.S. Johnson, A. Mehrotra, M. Trick (Eds.), Proceedings of the Computational Symposium on Graph Coloring and its Generalizations, Ithaca, New York, USA, 2002, pp. 112–125.

[9] **R. Dorne, J.K. Hao**, A new genetic local search algorithm for graph coloring, Lecture Notes in Computer Science *1*498 (1998) 745–754.

[10] **R. Dorne, J.K. Hao**, Tabu search for graph coloring, T-colorings and set Tcolorings, *Meta-Heuristics:* Advances and Trends in Local Search Paradigms for Optimization (1998) 77–92.

Received: June, 2010