Control and Cybernetics

vol. **28** (1999) No. 2

The smallest hard-to-color graphs for the classical, total and strong colorings of vertices

by

Marek Kubale, Krzysztof Manuszewski

Technical University of Gdańsk,
Foundations of Informatics Dept.,
Narutowicza 11/12, 80-952 Gdańsk
e-mail: {kubale, manus}@pg.gda.pl

Abstract: Let $\mathbf{A}(G)$ be the number of colors used by algorithm $\mathbf{A}$ to color the vertices of graph G. A graph G is said to be *hard-to-color* (HC) (resp. *slightly* HC) if for every (resp. some) implementation of the algorithm $\mathbf{A}$ we have $\mathbf{A}(G) > \chi(G)$, where $\chi(G)$ is the chromatic number of G. The study of HC graphs makes it possible to design improved algorithms trying to avoid hard instances as far as possible. Hard-to-color graphs are also good benchmarks for the evaluation of existing and future algorithms and provide an alternative way of assessing their quality. In this paper we demonstrate the smallest HC graphs for the best known coloring heuristics in classical applications, as well as when adapted to the chromatic sum coloring and strong coloring of vertices.

Keywords: benchmark, chromatic number, chromatic sum, graph coloring, hard-to-color graph, NP-completeness, strong coloring.

1. Introduction

Graph coloring has been an attractive field for many years, with several thousands of papers dealing with the study of chromatic properties of graphs. As people were becoming used to applying the tools of graphs theory to the solution of real-world technological and organizational problems, chromatic models appeared as a natural way of tackling many practical situations.

Unfortunately, the problem of optimal coloring the vertices of a graph is very hard. It was shown that not only determining the value of the *chromatic number* $\chi(G)$ for arbitrary graph G is NP-hard but also finding an n^ϵ-approximation for this problem remains NP-hard for any $\epsilon < 1/7$. Thus the construction of a polynomial-time approximate coloring algorithm with a polylogarithmic of graph coloring heuristics becomes increasingly important and this paper is devoted to this topic.

On the whole, the performance of graph coloring heuristics is studied by giving asymptotic results. These are usually the performance guarantee and the time complexity. Both functions tell us what one can expect at worst as the number of vertices $n \to \infty$, but we really don't know what is going on at the other end of the scale, say when $n \leq 10$. For this reason Hansen and Kuplinsky (1991) introduced the concept of hard-to-color graphs. The aim of studying such graphs is threefold. First, analyzing hard-to-color graphs makes it possible to obtain improved algorithms which avoid hard instances as far as possible (see Manuszewski, 1997). Second, it enables to search for small benchmarks, which are an indispensable tool in evaluating comprehensive families of graph coloring algorithms (cf. Głazek et al., 1997). Third, it provides a more sensitive way of assessing their efficiency as compared to the performance guarantee (the larger graph the better algorithm), since the overwhelming majority of coloring heuristics have asymptotically the same linear function of performance guarantee.

To tackle the problem more formally we need several definitions. Let $G = (V, E)$ be a graph with $|V| = n$ vertices and $|E| = m$ edges. A *graph coloring algorithm* $\mathbf{A}$ is an algorithm which, when applied to any graph G, produces a coloring of the vertices of G. The number of colors that $\mathbf{A}$ uses on G is denoted by $\mathbf{A}(G)$. The *performance guarantee* $\mathbf{A}(n)$ of $\mathbf{A}$ is the maximum value of $\mathbf{A}(G)/\chi(G)$ taken over all graphs G on n vertices. Ideally, we would like $\mathbf{A}$ to be efficient and $\mathbf{A}(G)$ to be equal (or very close) to $\chi(G)$ for any graph G. We shall see that this is impossible even on very small graphs. Given a heuristic algorithm $\mathbf{A}$, a graph G is said to be *slightly hard-to-color* (SHC) with respect to $\mathbf{A}$ if for some instance of algorithm $\mathbf{A}$ the number $\mathbf{A}(G)$ satisfies $\mathbf{A}(G) > \chi(G)$. We similarly define a *hard-to-color* (HC) graph as one for which every application of the algorithm (i.e. no matter what choice is made to break ties) results in a nonoptimal coloring. Given a family $\mathbf{F}$ of graph coloring algorithms, graph G is called a *benchmark* for $\mathbf{F}$ if G is HC for every algorithm in $\mathbf{F}$. Moreover, we define *smallest* graphs with this respect. More formally, in the case of HC graphs among all graphs $G = (V, E)$ we are looking for a graph $G' = (V', E')$ which realizes

$$|E'| = \min\{|E| : \text{ for every instance of } \mathbf{A}, \mathbf{A}(G) > \chi(G), \text{ and } |V| = n_0\},$$

where $n_0 = \min\{|V| : \text{ for every instance of } \mathbf{A}, \mathbf{A}(G) > \chi(G)\}$. A similar definition applies to SHC graphs. For simplicity, by $HC(\mathbf{A})$ (resp. $SHC(\mathbf{A})$) we denote the set of all smallest HC (resp. SHC) graphs for algorithm $\mathbf{A}$. Similarly, $HC(\mathbf{F})$ is the set of all smallest benchmarks for family $\mathbf{F}$. For readability, we drop braces if any of the sets above is 1-element. Obviously, it may happen that $HC(\mathbf{A}) = \emptyset$ and $HC(\mathbf{F}) = \emptyset$, as it is the case for a naive sequential

In this paper we consider three different models of vertex coloring. These are: classical, total and strong. In all these models the problem remains NP-hard. In Section 2 we formally describe heuristic algorithms under study. Section 3 is devoted to classical coloring. We give the smallest HC and SHC graphs for particular algorithms. Also, we give two benchmarks for some families of these algorithms. In Section 4 we adapt the considered algorithms to the chromatic sum problem. The need for considering this graph invariant follows from its numerous applications in task scheduling, resource allocations and $VLSI$ layout design. We give here the smallest HC and SHC graphs for the total coloring algorithms as well as two benchmarks. The final Section 5 is devoted to strong coloring of vertices. We list all known the smallest HC/SHC graphs and the only benchmark for this model which is presently known. Most of the graphs given in this article have been obtained by exhaustive computational search.

2. Algorithms

The first heuristic considered is a well-known *largest-first* (**LF**) algorithm. In the **LF** method the vertices are arranged in non-increasing order of their degree. This algorithm was given by Welsh and Powell (1967), in a slightly different but equivalent form. Its time complexity is $O(m + n)$ and performance guarantee $O(n)$. Algorithm **LF** optimally colors very simple classes of graphs only, e.g. odd cycles, odd wheels. Hansen and Kuplinsky (1991) showed that $SHC(\mathbf{LF})$ is path P_6 and $HC(\mathbf{LF})$ is a so-called *envelope*.

The **LF** sequential algorithm can be improved in a number of ways. One of them is revising the principle of vertex ordering. In a *smallest-last* (**SL**) method the vertices are ordered sequentially so that v_i has the minimum degree in the subgraph of G induced by vertices $v_1, \ldots, v_i$ $(1 \leq i \leq n)$. The **SL** algorithm with various refinements is due to Matula et al. (1972). As previously, its complexity is $O(m+n)$ and performance $O(n)$. Algorithm **SL** optimally colors polygon trees and k-trees. Kubale et al. (1997) showed that $SHC(\mathbf{SL}) = prism$ and $HC(\mathbf{SL}) = prismatoid$.

Another improvement over the **LF** algorithm is by using a dynamic ordering of vertices, as proposed by Brélaz (1979) in his method $DSATUR$. The *saturation LF* (**SLF**) algorithm, as we call it for short, repeatedly chooses an uncolored vertex v such that its neighbors represent a maximum number of color classes and puts v in the first color class where it fits. Algorithm **SLF** runs in $O((m + n) \log n)$ time and has performance guarantee $\mathbf{SLF}(n) = O(n)$. **SLF** optimally colors bipartite graphs, cacti and polygon trees. Recently, Janczewski et al. (2000) found the smallest SHC/HC graphs for this algorithm.

Yet another refinement of sequential algorithms is given by an interchange procedure as follows. Whenever a new color is going to be introduced for a vertex v_i $(4 \leq i \leq n)$ consider all possible pairs α and β of already used colors. If in each component of the graph induced by vertices colored with α and β the vertex v_i

of the bipartite components so that v_i is now joined to only color, say α. Then v_i can be given color β and no new color need be used. By an *LF with interchange* (**LFI**) algorithm we mean a sequential algorithm with interchange procedure applied to the list of vertices arranged in the *LF* order. The **LFI** algorithm can be run in time $O(mn)$. By an *SL with interchange* (**SLI**) algorithm we mean the sequential **SL** algorithm with interchange procedure incorporated into it. Johnson (1974) showed that $\mathbf{LFI}(n) = \mathbf{SLI}(n) = O(n)$. Also, the **SLI** algorithm can be implemented to run in time $O(mn)$. Similarly, by **SLFI** we mean the **SLF** algorithm with interchange procedure. It has asymptotically the same complexity and performance as its predecessors.

The penultimate heuristic algorithm considered here is due to Johnson (1974). This is an implementation of a general independent sets method in which the vertices of G are considered in some order, assigning a node v_i $(1 \leq i \leq n)$ to the color class 1 whenever it is not joined to a vertex already colored with 1. When no more vertices of a k-colorable graph can be colored with 1, all the vertices of color 1 (whose number is at least $\lfloor \log_k n \rfloor$) are removed and the coloring process continues likewise with color 2 on the remaining subgraph, etc. Johnson proposed the following greedy method for choosing vertices for consecutive independent sets. Starting with $V_1 = \emptyset$, set V_1 is augmented at each step by a minimum degree vertex of the subgraph generated by the non-neighbors of the current members of V_1. Then the process is repeated for the subgraph $G(V - V_1)$, and so on. We shall call this method a *greedy independent sets* (**GIS**) algorithm. **GIS** can be implemented in time $O(mn)$. In contrast to the previous methods, **GIS** is an algorithm with sublinear performance guarantee $O(n/\log n)$.

Leighton (1979) proposed a method, called *recursive LF* (**RLF**), which combines the strategy of algorithm **LF** with the structure of algorithm **GIS**. Here the rule of selecting the vertices to consecutive independent sets is only slightly different from that of **GIS**. Namely, instead of taking vertices with few uncolored neighbors, **RLF** takes vertices that have many neighbors among uncolored nodes adjacent to already colored vertices. The performance guarantee of this algorithm is still $O(n)$ and the time complexity is $O(n^3)$.

3. Classical coloring

The classical vertex coloring problem has particularly large number of applications in the area of computer science and electronics engineering. List of practical applications is too long to be cited here, so we refer the reader to Kubale (1998). Below in Table 1 we give the smallest SHC/HC graphs for every algorithm of the previous section. In addition, each SHC graph is accompanied by a vertex ordering that leads to a nonoptimal solution. The reader can easily verify that obeying this ordering results in a coloring using strictly more than the chromatic number of colors. For example, coloring P_6 in the given LF

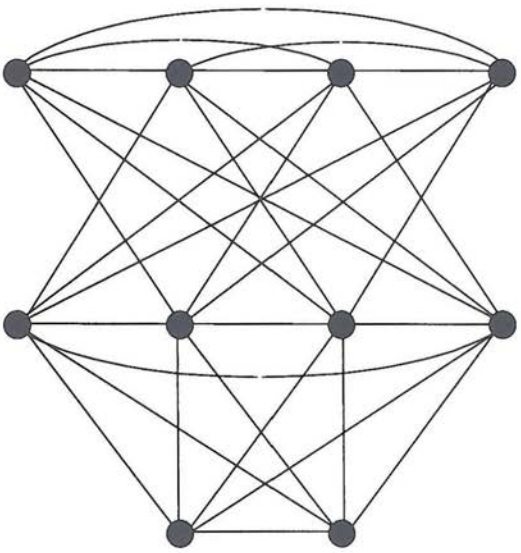


Figure 1. The smallest benchmark for the family F_S.

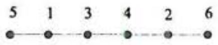
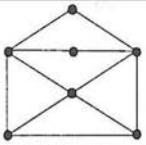
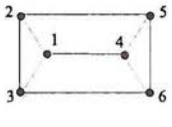
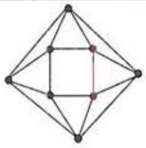
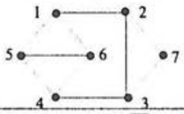
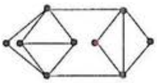
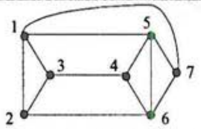
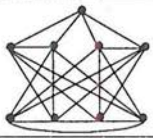
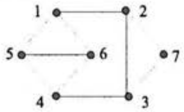
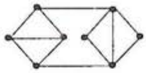
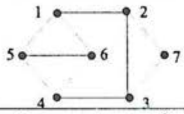
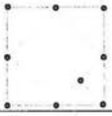
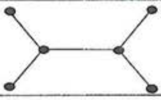
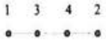
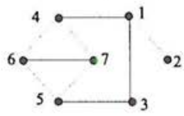
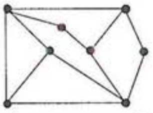
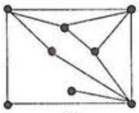
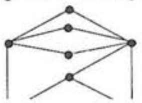
So far we have known two benchmarks for this model of coloring. They are both 10-vertex graphs. The first graph is the smallest benchmark for the family F_S of sequential algorithms with and without interchanging, namely $\mathbf{F}_S = \{\mathbf{LF}, \mathbf{SL}, \mathbf{SLF}, \mathbf{LFI}, \mathbf{SLI}, \mathbf{SLFI}\}$. This benchmark is shown in Fig. 1. Note that $HC(\mathbf{F}_S)$ is a supergraph for each smallest HC graph of every algorithm in family $\mathbf{F}_S$.

The second benchmark is connected with the family of practically applied algorithms without interchange. Namely, these are $\{\mathbf{LF}, \mathbf{SL}, \mathbf{SLF}, \mathbf{GIS}, \mathbf{RLF}\}$. This benchmark is shown in Fig. 2.

4. Total coloring

Given a graph G, the *chromatic sum* $\sum(G)$ is the smallest total of colors among all proper vertex colorings of G using natural numbers. The need for considering this graph invariant follows from its numerous applications in problems arising from task scheduling, resource allocation and $VLSI$ layout design. For example, in scheduling of unit execution time tasks on a set of processors, finding a chromatic sum solution closely corresponds to constructing a mutual exclusion schedule that minimizes both the mean flow time and total completion time.

The chromatic sum problem is NP-hard and remains so even if G is bipartite, as shown by Bar-Noy, Kortsarz (1998). Therefore, in practice we have to use fast approximation algorithms. To this aim we can use any of the classical [illegible]

Method	The smallest graphs		References
	SHC	*HC*	
LF			Hansen, Kuplinsky (1991)
SL			Kubale, Pakulski, Piwakowski (1997)
LFI			Manuszewski (1997)
SLI			Manuszewski (1997)
SLF			Janczewski, Kubale, Manuszewski, Piwakowski (2000)
SLFI			Manuszewski (1997)
GIS			Manuszewski (1997)
RLF		  	Manuszewski (1997)

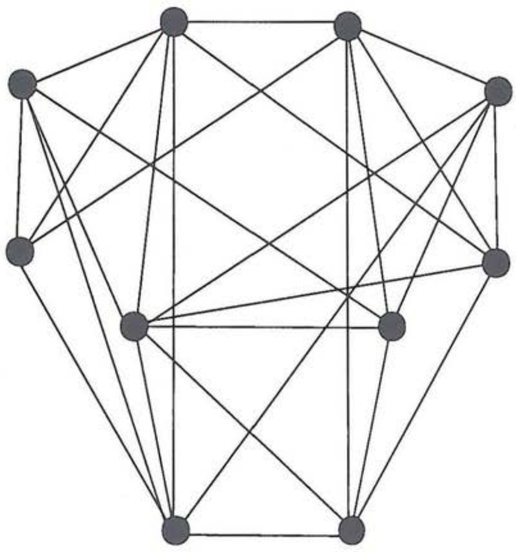


Figure 2. The smallest benchmark for the family of algorithms without interchange.

total coloring of the vertices of G and its sum gives an upper bound on $\sum(G)$. Such algorithms have the same performance guarantee and time complexity, cf. Kubale, Manuszewski (1996). Thus the only difference are the smallest HC and slightly HC graphs. A catalogue of such graphs is given in Table 2. Note that these graphs are smaller than those of Table 1, which confirms the fact that the total coloring problem is harder than the classical one.

From Table 2 it follows that path P_3 is the smallest benchmark for family $\{\mathbf{LF}, \mathbf{LFI}, \mathbf{SLF}, \mathbf{SLFI}, \mathbf{RLF}\}$. But we are interested mainly with sequential algorithms. As previously, let $\mathbf{F}_S$ denote the family of sequential algorithms $\{\mathbf{LF}, \mathbf{SL}, \mathbf{SLF}, \mathbf{LFI}, \mathbf{SLI}, \mathbf{SLFI}\}$ and let symbol $\mathbf{F}_I$ stand for the family of coloring algorithms based on independent set rule $\{\mathbf{GIS}, \mathbf{RLF}\}$. Fig. 3 shows the smallest benchmark for family $\mathbf{F}_S$ and Fig. 4 depicts graph $HC(\mathbf{F}_I)$.

5. Strong coloring

Given a graph G, the *strong chromatic number* $\chi_S(G)$ is the minimum number of colors needed to color the vertices of G in such a way that no two vertices at distance at most 2 have the same colors. The need for considering this particular model of coloring follows from its possible applications in cellular telecommunication technology (cf. Kubale, 1998).

The strong coloring problem is NP-hard. Therefore, in practice we have to

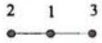
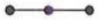
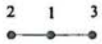
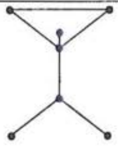
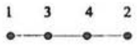
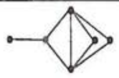
Method	The smallest graphs		References
	SHC	*HC*	
LF, LFI, SLF, SLFI, RLF			Głazek, Kubale, Manuszewski (1997)
SL, SLI, SLFI			Głazek, Kubale, Manuszewski (1997)
GIS			Głazek, Kubale, Manuszewski (1997)

Table 2. A catalogue of the smallest SHC/HC graphs for the total model of coloring.

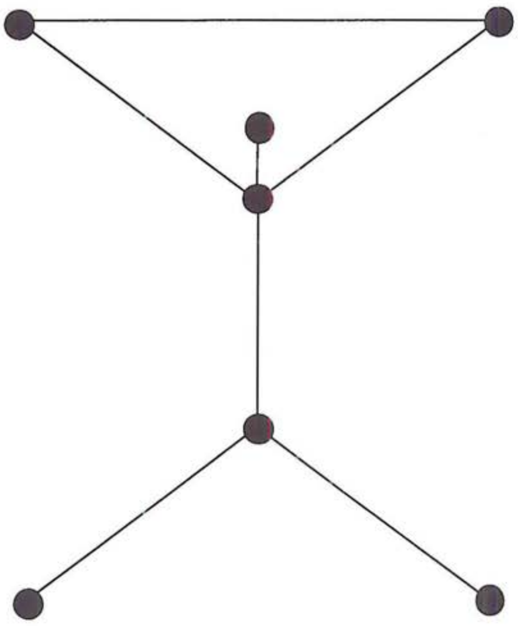


Figure 3. Weight lifter – the smallest benchmark for family $\mathbf{F}_S$.

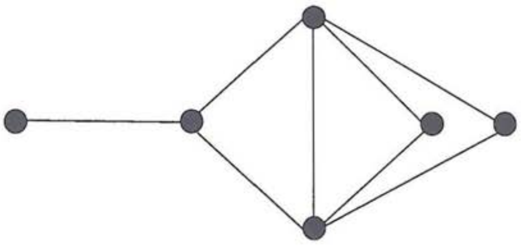


Figure 4. The smallest benchmark for the family of independent sets algorithms.

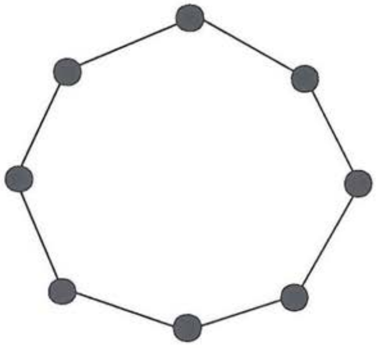


Figure 5. $HC(\{\mathbf{SL}, \mathbf{SLF}\}) = C_8$.

algorithms of Section 2, since $\chi_S(G) = \chi(G^2)$, where G^2 is the *square* of G, i.e. a graph whose adjacency matrix is $A + A^2$. Thus any proper coloring of graph G^2 is a strong coloring of G as well. A catalogue of the smallest HC/SHC graphs for these algorithms when applied to the strong coloring of vertices is given in Table 3.

So far, we have only known a benchmark for a small family of two algorithms, namely: **SL** and **SLF**. The smallest benchmark for this family is the cycle shown in Fig. 5.

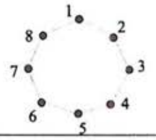
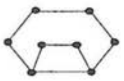
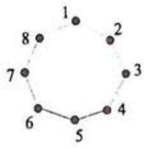
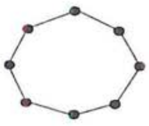
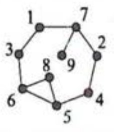
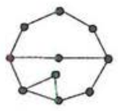
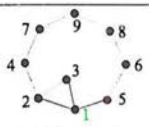
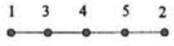
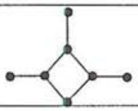
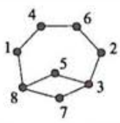
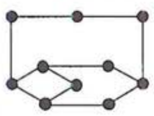
Method	The smallest graphs		References
	SHC	*HC*	
LF			Manuszewski (1997)
SL, SLF			Kubale, Pakulski, Piwakowski (1997)
LFI			Manuszewski (1997)
SLI		Unknown	Manuszewski (1997)
GIS			Manuszewski (1997)
RLF			Manuszewski (1997)

Table 3. A catalogue of the smallest *SHC*/*HC* graphs for the strong model of coloring.

References

BAR-NOY, A., BELLARE, M., HALLDÓRSSON, M.M., SHACHNAI, H., TAMIR, T. (1998) On chromatic sums and distributed resource allocation. *Inform. Comp.*, **140**, 183-202.

BAR-NOY, A., KORTSARZ, G. (1998) Minimum color sum of bipartite graphs. *J. Algorithms*, **28**, 339-365.

BELLARE, M., GOLDREICH, O., SUDAN, M. (1995) Free bits, PCP and non-approximability – Towards tight results. *Proc. 36-th IEEE Symp. on Foundations of Computer Science*, Los Alamitos, 422-431.

BRÉLAZ, D. (1979) New methods to color the vertices of a graph. *Communications of ACM*, **22**, 251-256.

GŁAZEK, W., KUBALE, M., MANUSZEWSKI, K. (1997) The quest for small benchmarks for the chromatic sum problem. *Arch Cont. Sci.*, **6**, 249-260.

HANSEN, P., KUPLINSKY, J. (1991) The smallest hard-to-color graph. *Disc. Math.*, **96**, 199-212.

JANCZEWSKI, R., KUBALE, M., MANUSZEWSKI, K., PIWAKOWSKI, K. (2000) The smallest hard-to-color graph for algorithm DSATUR. (to appear in *Disc. Math.*).

JOHNSON, D.S. (1974) Worst-case behavior of graph coloring algorithms. *Proc. 5-th S-E. Conf. on Combinatorics, Graph Theory and Computing*, Utilitas Mathem., Winnipeg, 513-527.

KUBALE, M. (1998) *Introduction to Computational Complexity and Algorithmic Graph Coloring.* Wydawnictwo GTN, Gdańsk.

KUBALE, M., MANUSZEWSKI, K. (1996) Algorytmy sumarycznego kolorowania grafów (in Polish). *Zesz. Nauk. Pol. Śl., Seria Automat.*, **117**, 213-222.

KUBALE, M., PAKULSKI, J., PIWAKOWSKI, K. (1997) The smallest hard-to-color graph for the SL algorithm. *Disc. Math.*, **164**, 197-212.

LEIGHTON, F.T. (1979) A graph coloring algorithm for large scheduling problems. *Journal RNBS*, 489-506.

MANUSZEWSKI, K. (1997) Grafy algorytmicznie trudne do kolorowania (in Polish). Ph.D. Thesis, Technical University of Gdańsk.

MATULA, D.W., MARBLE, G., ISAACSON, D. (1972) Graph coloring algorithms. In: *Graph Theory and Computing*, Academic Press, New York, 109-122.

WELSH, D.J., POWELL, M.B. (1967) An upper bound for the chromatic number of a graph and its application to timetabling problem. *Comp. J.*, **10**, 85-86.