As appearing in *Information Processing Letters*, Feb. 1993.

A Still Better Performance Guarantee for Approximate Graph Coloring

Magnús M. Halldórsson*
School of Information Science
Japan Advanced Institute of Science and Technology, Hokuriku
Tatsunokuchi, Ishikawa 923-12, Japan

Email: magnus@jaist-east.ac.jp

Abstract

We present an approximation algorithm for graph coloring which achieves a performance guarantee of $\mathcal{O}(n(\log\log n)^2/(\log n)^3)$, a factor of $\log\log n$ improvement.

Keywords: analysis of algorithms, approximation algorithms, graph coloring.

1 Introduction

A coloring of a undirected graph is a partition of the vertices into color classes so that no edge joins two vertices in the same class. The objective is to use as few colors as possible. Given that it is $\mathcal{NP}$-hard to color every graph with the minimum number of colors, we are interested in polynomial time algorithms that provide guarantees on the number of colors they use. The *performance guarantee* of an approximate coloring algorithm is the largest ratio, over all graphs on n vertices, of the number of colors used to the minimum number of colors required.

The first algorithm with a non-trivial performance guarantee was given by Johnson [4]. This performance guarantee of $\mathcal{O}(n/\log n)$ was later improved by Wigderson [6] to $\mathcal{O}(n(\log\log n/\log n)^2)$, and by Berger and Rompel [1] to $\mathcal{O}(n(\log\log n/\log n)^3)$.

We further improve the best performance guarantee known by applying an approximate algorithm for the independent set problem by Boppana and Halldórsson [2] in combination with these earlier results on coloring. The performance guarantee we obtain is $\mathcal{O}(n(\log\log n)^2/\log^3 n)$.

2 Graph Coloring Algorithm

We shall present an algorithm for finding large independent sets in k-colorable graphs. This will directly lead to a good coloring algorithm, as well shall indicate later. For simplicity of presentation, we state the algorithm in its randomized form.

Let G be a k-colorable graph on n vertices, and let A be a largest color class under some optimal coloring of G. Let I be a set of $\log_k n$ randomly selected vertices. Let $\overline{N}(I)$ denote the subgraph induced by vertices neither in I nor adjacent to any vertex in I; alternatively, $\overline{N}(I)$ is the graph obtained from G by removing I, the vertices adjacent to vertices in I, and all incident edges. For a graph S, $|S|$ denotes the number of vertices in S.

Our algorithm is based on the following three observations:

*Research performed at Rutgers University. Supported in part by Center for Discrete Mathematics and Theoretical Computer Science graduate fellowship.

Observation 1

1. *$I \subseteq A$ (and thus $A \subseteq I \cup \overline{N}(I)$), with probability at least $1/n$.*
2. *$\overline{N}(I)$ is k-colorable.*
3. *If $\overline{N}(I)$ is small and $A \subseteq I \cup \overline{N}(I)$, then an independent set algorithm of [2] finds a large independent set in $\overline{N}(I)$.*

The first two observations are from [1]. The first observation holds because the ratio of the number of $\log_k n$-sized vertex sets contained in A to the number of all such sets is at least $1/n$. It tells us that some searching will give us a sizable independent set with certain nice properties. The second observation forms the basis of a recursive algorithm: Apply the same arguments on the graph $\overline{N}(I)$, and merge the result with the independent set I.

The third observation is new to this paper. It allows us to argue that when the independent set algorithm fails to find a large approximation, the size of $\overline{N}(I)$ must be large, facilitating a deeper recursion. We formalize the observation in the following lemma, with $\overline{N}(I) \cup I$ represented by S. We first need the following result.

Fact 1 ([2]) *Given a graph G with independence number at least tn where $t \geq 1/\log n$, the algorithm* CliqueRemoval *will find an independent set of size at least $e^{-1}n^t/t$.*

Lemma 1 *Let S be a subgraph of G on at most $\frac{n}{k}\frac{\log n}{2\log\log n}$ vertices, containing an n/k independent set. Then* CliqueRemoval *returns an independent set of size at least $\log^3 n/6\log\log n$, when applied to S.*

Proof. When applied to S, CliqueRemoval finds an independent set of size at least

$$e^{-1}|S|^{(n/k)/|S|} \cdot |S|/(n/k) = e^{-1}(k/n)e^{(\log|S|)(n/(k|S|)+1)}.$$

This is minimized when $|S|$ is maximized, giving

$$e^{-1}(\log n/2\log\log n)\ n^{2\log\log n/\log n} \ \geq\ e^{-1}\log^3 n/2\log\log n.$$

∎

We now present our algorithm in fig. 2.1.

Theorem 1 *Given a k-colorable graph G,* SampleIS*(G,k) returns an independent set of size at least $\log_k n \log n/(2\max(\log(k \cdot 2\log\log n/\log n), 1))$ in expected polynomial time.*

Proof. The proof is by induction on the size of the graph. The claim holds trivially for the single vertex graph, hence we inductively assume the claim holds for all graphs of size less than n. Each iteration of the loop in the algorithm can conclude in one of three different ways:

a) $|\overline{N}(I)| \geq n/k \cdot \log n/2\log\log n$. Then, by the inductive assumption, SampleIS($\overline{N}(I)$, k) is at least

$$(\log|\overline{N}(I)|)^2/(2\log k\max(\log(k(2\log\log|\overline{N}(I)|/\log|\overline{N}(I)|)),1)).$$

As $|\overline{N}(I)|$ is trivially at most n, and at least the assumed lower bound, this becomes at least

$$\log^2 n/(2\log k\max(\log(k(2\log\log n/\log n)),1)) - \log n/\log k.$$

But the size of I is $\log_k n = \log n/\log k$, hence the size of the result $I \cup$ SampleIS$(\overline{N}(I), k)$ satisfies the claim.

```
SampleIS (G,k)
{ G is k-colorable, |G| = n }
begin
  if |G| ≤ 1 then return G
  forever do
    randomly pick a set I of log_k n nodes
    if I is independent then
      if |N̄(I)| ≥ n/k · log n/2 log log n then
        return (I ∪ SampleIS(N̄(I),k))
      else
        I_2 = CliqueRemoval(N̄(I)) ∪ I
        if |I_2| ≥ log^3 n/6 log log n then return (I ∪ I_2)
        { else I ⊈ A }
      endif
    endif
  od
end
```

Algorithm 2.1: Algorithm for finding independent sets in k-colorable graphs

b) CliqueRemoval returned a set of size $\log^3 n/(6 \log\log n)$, in which case the claim is satisfied.

c) $|\overline{N}(I)| < n/k \cdot \log n/2 \log\log n$, and CliqueRemoval was not successful. Then, according to lemma 1, I could not have been contained in the largest color class of the graph.

After at most expected n iterations, the randomly selected vertex set I will be contained in the largest color class of the graph, at which point only the successful outcomes b) and c) can occur. ∎

The time complexity of each iteration of the loop is equivalent to the complexity of CliqueRemoval or $\mathcal{O}(\chi(G)(n+m))$, where $\chi(G)$ is the chromatic number of the graph and m is the number of edges in the graph. Each invocation of the method executes the loop at most n times on the average, while the number of invocations is proportional to the size of the approximation.

Note that the algorithm can be made deterministic using the method presented in [1]. Partition the vertices of the graph into bins of size $k \log_k n$. By the pigeonhole principle, at least one of the $n/(k \log_k n)$ bins must contain a subset of A of size $\frac{n/k}{n/(k \log_k n)} = \log_k n$, and we can find this subset by exhaustively examining the $\binom{k \log_k n}{\log_k n} \approx n$ such subsets in each bin.

An algorithm for finding independent sets leads directly to a coloring algorithm as follows. Call the independent set algorithm and color the result with the first color, and then remove this set of vertices and incident edges from the graph. Call the independent set algorithm on the resulting graph and color with the second color, and so on. The following lemma, which has been used implicitly numerous times [4, 6, 2, 1], indicates how the approximations relate.

Lemma 2 *An iterative application of an algorithm that guarantees finding an independent set of size $f_k(n) = \mathcal{O}(\sqrt{n})$ in a k-colorable graph G, produces a coloring of G with no more than $2n/f_k(n)$ colors.*

Proof. Assume that f is a positive, non-decreasing function. Then, the iterative application of the independent set algorithm produces a coloring with at most $\sum_{i=1}^{n} 1/f_k(i)$ colors, since each node contributes at most $1/f_k(n')$ colors, where n' is the number of nodes remaining in the graph at the time when the node was assigned a color. This discrete integral is at most $t/(t-1) \cdot n/f_k(n)$ when $f_k(n)$ grows no faster than $\mathcal{O}(n^{1/t})$. ∎

Corollary 1 *Graph coloring is* $\mathcal{O}(n(\log\log n)^2/(\log n)^3)$ *approximable.*

Proof. First notice that we can run SampleIS for all values of k, obtaining an approximation at least as good as that of SampleIS(G,χ), where $\chi = \chi(G)$ denotes the chromatic number of G. We shall also run CliqueRemoval, and retain the best result. We color the graph with a repeated application of this combined method for finding independent sets.

By the preceding lemma, when $\chi \leq \log n/2\log\log n$, an application of CliqueRemoval will produce a coloring with at most $\mathcal{O}(n^{1-1/\chi}/\chi)$ colors, while when $\chi \geq \log n/2\log\log n$, using SampleIS will yield a coloring using at most $\mathcal{O}(n\log\chi\max(\log(\chi\cdot 2\log\log n/\log n),1)/(\log n)^2)$ colors. As a function of χ, the ratio of the number of colors used to the chromatic number is monotone decreasing in the former case and monotone increasing in the latter case. Thus it suffices to observe that at the point of discontinuity, $\chi = \log n/2\log\log n$, the value of either function is $\mathcal{O}(n(\log\log)^2/\log^3 n)$. ∎

3 Remarks

In the more than two years that have passed since the development of the algorithm reported here, no method has been found that improves on the performance guarantee. By recent results of Lund and Yannakakis [5], obtaining a performance guarantee of n^ϵ, for some $\epsilon > 0$, is $\mathcal{NP}$-hard. Using results of [2] we can argue that even a $n/\log^3 n$ approximation is not possible within the framework of current approximate coloring algorithms. Also, the fact that on-line algorithms, and some of their extensions, can not provide $n/\log^2 n$ performance guarantee [3] can be taken as a further evidence of the hardness of the approximate coloring problem. We thus conclude with a bold conjecture.

Conjecture 1 *The best possible performance guarantee for graph coloring is* $\Theta(n/\log^c n)$*, for some constant* $c \geq 3$*.*

Acknowledgements

I am indebted to my adviser, Ravi Boppana, for assistance, ideas, and discussions leading to this and related work. I also thank the anonymous referee and Jaikumar Radhakrishnan for suggestions for improving the presentation.

References

[1] B. Berger and J. Rompel. A better performance guarantee for approximate graph coloring. *Algorithmica*, 5(4):459–466, 1990.

[2] R. B. Boppana and M. M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT*, 32(2):180–196, June 1992.

[3] M. M. Halldórsson and M. Szegedy. Lower bounds for on-line graph coloring. In *Proc. 3rd ACM-SIAM Symp. on Discrete Algorithms*, pages 211–216, Jan. 1992.

[4] D. S. Johnson. Worst case behaviour of graph coloring algorithms. In *Proc. 5th Southeastern Conf. on Combinatorics, Graph Theory, and Computing. Congressus Numerantium X*, pages 513–527, 1974.

[5] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. Manuscript, July 1992.

[6] A. Wigderson. Improving the performance guarantee for approximate graph coloring. *J. ACM*, 30(4):729–735, 1983.