LITERATURE REVIEW: Distributed Graph Coloring

David Worley
School of Computer Science
University of Ottawa
Ottawa, Canada
dworley@uottawa.ca

December 7, 2021

1 Introduction

Graph coloring, the problem of assigning a color to each vertex of a graph such that no two neighbouring vertices share the same color, is one of the most important problems in graph theory. From the proposal of the Four color Theorem in the mid-1850s until today, graph coloring has been a fundamental problem in graph theory, and the focus of an incredible amount of research. The optimization aspect of this problem is identifying what the minimum amount of colors is needed to create such a proper coloring on a graph, and of course implementing fast algorithms to find such colorings.

Furthermore, graph coloring sees many applications in scheduling and identifying groups in social graphs. Unfortunately, the problem of deciding if a graph has a k-coloring (can be colored with k colors) is NP-complete, and thus no sequential, polynomial algorithms for finding graph colorings exist for general cases. This makes finding ways to improve current state-of-the-art algorithms incredibly important, as these algorithms need to run efficiently on graphs with massive scale.

Distributed Graph coloring is an area looking to solve this problem by parallelizing algorithms for graph coloring. While this introduces additional considerations regarding the parallel model under consideration, the speed-ups obtained as a result are strongly beneficial. Thus, the development of simple, efficient distributed graph coloring algorithms is a large research area, with entire books dedicated solely to the topic [1].

2 Literature Review

2.1 Models and Bounds for Distributed Graph Coloring

Currently, the majority of work in distributed graph coloring focuses on finding a k-coloring such that $\Delta + 1 \leq k \leq O(\Delta^2)$, where Δ is the maximum degree of a vertex. The lower bound results from the common fact that any graph where the maximum degree is Δ can be colored with $\Delta + 1$ colors, with simple greedy algorithms able to find such a coloring. The upper bound is a result of Linial who, in 1992, proposed a model for distributed graph algorithms called the LOCAL model [5]. This model is round-based, allowing for each vertex to transmit information to each of its neighbours at the end of each round. Using this model, Linial shows that an $O(\Delta^2)$ coloring can be generated in 1 round in $O(\log^* n)$

time, where $\log^* n$ is the iterative logarithm, or how many times the log function must be applied to n until the result is 1.

This LOCAL model proves to be the simplest to work with, though other common models are also important in the literature. These include the SET-LOCAL model, sometimes called the weak LOCAL model, in which vertices do not have IDs, and so cannot distinguish between the messages of its neighbours. The main assumption algorithms work off if within the SET-LOCAL model is that the algorithm *begins* with a proper coloring [4]. Another common model is the more restrictive CONGEST model, in which the vertices can only transmit $O(\log n)$ data to each neighbour per round. This is particularly restrictive as vertices are typically identified using bit IDs of size $O(\log n)$.

2.2 One Round Color Reductions

While the $O(\Delta^2)$ bound proposed above is large, this is acceptable due to the presence of color-reduction algorithms that can be used to reduce the amount of colors in the coloring. With Linial's algorithm acting as a preprocessing step, color reduction algorithms can focus on reducing the $O(\Delta^2)$ coloring instead. In fact, making this distinction between coloring algorithms and color reduction algorithms is unnecessary, as any coloring algorithm can be considered as a color reduction algorithm from an input $|V|$-coloring, where V is the set of vertices of the input graph (and so each vertex has its own color). These color reduction algorithms can make use of the reduced $O(\Delta^2)$ coloring well to create a smaller coloring in as few rounds as possible, making this output coloring from Linial's algorithm still valuable. Alternatively, color reduction algorithms can focus on specific settings in which a coloring can be reduced greatly in *one* round. Linial himself once again sets strong foundations for this area, presenting an algorithm to reduce a k-coloring of a graph to a $O(\Delta^2 \log m)$-coloring in a single round under his model [5].

With these foundations, much research focuses on one-round color reduction algorithms that can reduce the colors significantly for certain classes of graphs. For example, one-round color reduction algorithms were developed for directed paths that can reduce k-colorings to 3-colorings in $\frac{1}{2}\log^* n + O(1)$ rounds [3]. This was later reduced into a tight bound of $\frac{1}{2}\log^* n$ [7]. This result is important due to its usefulness as a subroutine of other distributed graph coloring algorithms. Other improvements look towards specific classes of graphs or reductions from k-colorings under some set of assumptions about k. For example, Maus proposes a one-round coloring algorithm that reduces an k-coloring to a $m(\Delta - m + 2)$-coloring, given that $k \geq m(\Delta - m + 3)$, removing m colors from the coloring, where $1 \leq m \leq \Delta/2 + 3/2$ [6].

2.3 Improving Coloring Algorithms

With strong color reduction algorithms, efficient colorings can be obtained through the use of a coloring algorithm followed by repeated color reductions to said graph. Considering our lower bound of $\Delta + 1$, its natural to ask how many rounds a coloring algorithm requires to result in a $(\Delta+1)$-coloring. In 1993, Szegedy and Vishwanathan showed that for algorithms that were locally iterative, a $(\Delta+1)$-coloring algorithm requires $\Omega(\Delta \log \Delta + \log^* n)$ rounds, barring the existence of a special type of coloring whose reduction could be done very efficiently [8]. A locally-iterative algorithm is one where each vertex chooses its next color based solely on the colors of its local neighbourhood, so this bound applied to a large class of the algorithms within the field. For over 25 years, no algorithm could make use of such a special coloring and thus could not beat the proposed *SV Barrier* lower bound. It wasn't until 2021 that this barrier was broken, with a locally iterative algorithm that could compute a $(\Delta + 1)$-coloring with runtime $O(\Delta + \log^* n)$ [2].

2.4 Maus's Algorithm

The algorithm proposed by Maus in [6] is a round based color reduction algorithm scaling between the two bounds presented above as follows. For a given integer $1 \leq k \leq O(\Delta)$, the algorithm generates an $O(\Delta k)$-coloring in $O(\Delta/k)$ rounds through a trial based reduction of an input coloring, like one given by Linial's algorithm. Each vertex of the graph will compute a sequence of k colors and attempt to color itself with this sequence, stopping on the first one that is found to be without conflict. If no colors in the first k are without conflict, then k more are tested in the subsequent round. This process repeats until all vertices are colored.

Since Δ is known and k can be chosen before the algorithm is run, k can always be selected proportionally to Δ to make the algorithm scale from a $O(1)$-round reduction algorithm that generates an $O(\Delta^2)$ coloring, subsuming Linial's algorithm, or an $O(\Delta)$-round algorithm to compute an $O(\Delta)$ coloring. The algorithm is also general enough for other types of graph coloring problems, such as d-defective colorings where vertices are allowed to have the same color as at most d of their neighbours.

References

[1] L. Barenboim and M. Elkin. Distributed graph coloring: Fundamentals and recent developments. *Synthesis Lectures on Distributed Computing Theory*, 4, 07 2013.

[2] L. Barenboim, M. Elkin, and U. Goldenberg. Locally-iterative distributed $(\delta + 1)$-colouring below szegedy-vishwanathan barrier, and applications to self-stabilization and to restricted-bandwith models. *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 437–446, 2018.

[3] R. Cole and U. Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Information and Control*, 70(1):32–53, 1986.

[4] D. Hefetz, F. Kuhn, Y. Maus, and A. Steger. Polynomial lower bound for distributed graph colouring in a weak local model. *Proc. of the 30th International Symp. on Distributed Computing*, pages 99–113, 2016.

[5] N. Linial. Locality in distributed graph algorithms. *Society for Industrial and Applied Mathematics*, 21(1):193–201, 1992.

[6] Y. Maus. Distributed graph colouring made easy. *SPAA '21*, pages 362–372, 2021.

[7] J. Rybicki and J. Suomela. Exact bounds for distributed graph colouring. *SIROCCO 2015*, 9439:46–60, 2015.

[8] M. Szegedy and S. Vishwanathan. Locality based graph coloring. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '93, page 201–207, New York, NY, USA, 1993. Association for Computing Machinery.