Towards the Formal Verification of Wigderson's Algorithm

Siraphob Phipathananunth
ben.phipathananunth@yale.edu
Yale University
USA

Abstract

We present progress towards the formal verification of Wigderson's graph coloring algorithm in Coq. We have created a library of formalized graph theory that aims to bridge the literature gap between introductory material on Coq and large-scale formal developments, while providing a motivating case study. Our library contains over 180 proven theorems. The development is available at https://github.com/siraben/coq-wigderson.

CCS Concepts: • **Theory of computation** → **Program verification**; **Program analysis**; **Graph algorithms analysis**.

Keywords: formal verification, coq, graph coloring, approximation algorithms

ACM Reference Format:
Siraphob Phipathananunth. 2023. Towards the Formal Verification of Wigderson's Algorithm. In *Companion Proceedings of the 2023 ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity (SPLASH Companion '23), October 22–27, 2023, Cascais, Portugal.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3618305.3623600

1 Introduction

Graph theory has diverse applications in mathematics, computer science, and other fields. It has also been a focus of formal verification, including results such as the four-color theorem [7]. However, for graph theory in Coq, we note that there is a significant literature gap between introductory material and such large-scale efforts, which we aim to address by introducing a tractable case study from approximation algorithms.

To do this, we have developed a framework from scratch for graph-theoretic results and graph coloring algorithms, and demonstrated its utility in making progress towards formally verifying Wigderson's algorithm. Although its proof of correctness is not difficult, this is the first attempt at a formal proof of correctness of Wigderson's algorithm. This library is also written with the intention of being relatively free of dependencies, apart from proof automation with CoqHammer [5], and does not require advanced knowledge of Coq beyond the Software Foundations textbook [1]. Additionally, most of the lemmas are annotated, and the structure of arguments follows that of mathematical proofs, making it a useful pedagogical demonstration of formal verification for students familiar with discrete mathematics.

SPLASH Companion '23, October 22–27, 2023, Cascais, Portugal

ACM ISBN 979-8-4007-0384-3/23/10.
https://doi.org/10.1145/3618305.3623600

2 Background

Wigderson's algorithm [10] is an approximate graph coloring algorithm that colors a 3-colorable graph using at most $O(\sqrt{n})$ colors in polynomial time, and more generally using $O(n^{1-\frac{1}{k-1}})$ colors for a k-colorable graph. It is notable for its simplicity, and the bound was later improved by [3], [8] and [4]. However, latter algorithms utilize techniques such as semidefinite programming which would make verification significantly more involved.

If the graph is not 3-colorable, then the algorithm returns a valid approximation or a constructive witness that the graph is not 3-colorable. In phase 1, while the graph has maximum degree of at least k, we select a highest-degree vertex v, 2-color its neighborhood, and coloring v with the remaining color until no high-degree vertices remain. In phase 2, the rest of the graph is colored uniquely, as shown in line 11 of Algorithm 1. The choice of k that minimizes the number of colors used is $\sqrt{n}$ [10].

To prove correctness, we observe that in a 3-colorable graph, the open neighborhood of any vertex is bipartite, thus 2-colorable. Each iteration of the loop uses at most two additional colors. Then we color the remaining vertices uniquely. Thus, the algorithm gives a coloring that uses at most $3\sqrt{n}$ colors.

3 Design and implementation of our graph library

We base our graph library on Volume 3 of Appel et al. [1]. Directed graphs are represented as functional maps from positive integers (vertices) to sets of positive integers (outward adjacent vertices). To add structure such as undirectedness

Algorithm 1 Wigderson's 3-coloring algorithm [10].

Input: A 3-colorable graph $G(V, E)$

$n \leftarrow |V|$
$i \leftarrow 1$
while max_degree$(G) \geq k$ **do**
  $v \leftarrow$ highest-degree vertex in G
  $H \leftarrow$ subgraph of G induced by neighborhood $N_G(v)$
  2-color H with colors $i, i+1$
  color v with color $i+2$.
  $i \leftarrow i+2$
  $G \leftarrow$ subgraph of G after deleting $N_G(v) \cup \{v\}$
end while
color G with colors $i, i+1, i+2, \ldots, \Delta(G)$ and halt

and irreflexivity, we impose constraints on the respective graph's vertex sets. We also define relations between graphs for derived structures such as subgraphs and neighborhoods.

A key lemma we proved and is used in reasoning about the correctness of phase 1 is shown in Listing 1. It states that if a coloring is complete (total) on a graph, and if that coloring is an $(n+1)$-coloring, then the restriction of the coloring to the neighborhood of an arbitrary vertex results in a complete n-coloring.

```
Lemma nbd_Sn_colorable_n :
  forall (g : graph) (f : coloring) (p : colors)
    (n : nat),
    coloring_complete p g f ->
    n_coloring f p (S n) ->
    forall v ci, M.find v f = Some ci ->
          n_coloring (restrict_on_nbd f g v)
                     (S.remove ci p) n /\
  coloring_complete (S.remove ci p)
                    (neighborhood g v)
                    (restrict_on_nbd f g v).
```

Listing 1. Key lemma for phase 1.

To reason about the functions, we make extensive use of Coq's `functional induction` commands [2]. Since graph-coloring algorithms often have non-structural termination criteria, for instance, coloring vertices until a boolean predicate is met, we prove its termination in Coq. In `phase2`, while the graph has nonzero degree, we select its highest-degree vertices one by one to ensure that they are non-adjacent, and color them with the color $n+1$, continuing in this manner until the graph has zero degree.

4 Preliminary verification results

Since the base material from Software Foundations is quite minimal, we have defined and proven more than 180 additional theorems pertaining to various aspects of graph theory, including subgraphs, coloring maps, degrees, and map restrictions. For instance, since the coloring returned by phase 2 is constructed recursively, we have to prove that the valid coloring invariant holds. This further involves proving that unions of disjoint colorings preserve the invariant, and that inducing colorings from sets of pairwise non-adjacent vertices is valid.

Additionally, we make extensive use of CoqHammer [5] to automate mechanical portions of proofs. We almost have a complete proof of the correctness for phase 2. For phase 1, we have proven theorems regarding unions of disjoint 2-coloring maps and the disjointness of iteratively removing highest-degree vertices. Remaining work includes formulating and proving the correctness of a 2-coloring algorithm that is used to color the neighborhoods, which was implicitly assumed in the literature and left postulated for now.

Table 1. Statistics of our graph theory development.

Filename	Code
`subgraph.v`	1145
`coloring.v`	994
`graph.v`	407
`wigderson.v`	228
`restrict.v`	181
`munion.v`	63
Total	3018

5 Related work

There are other libraries of formalized graph theory results in the literature [6], [9]. However, we decided to create our own implementation due to unfamiliarity with the SSReflect library, or unneeded generality. Since our goal is also pedagogical, we want to introduce abstractions as minimally as possible, even if that limits more general statements. It is our hope that eventually these efforts could be combined into a single work.

6 Conclusion

We have made progress towards formally verifying Wigderson's graph coloring algorithm using Coq. Our approach led to the creation of a graph theory library containing over 180 relevant proven theorems. Although the verification is not yet complete, future work will focus on addressing remaining admitted lemmas and expanding the library for verifying other graph approximation algorithms. In this regard, we aim to continue bridging the gap between introductory material and large-scale formal developments in Coq through illustrative examples such as graph coloring. The accompanying code for this work can be found at https://github.com/siraben/coq-wigderson.

Acknowledgments

We would like to especially thank Steven Tschantz for his invaluable guidance and support, and to Jamison Homatas for contributions to an early version of this work.

References

[1] Andrew W. Appel. 2023. *Verified Functional Algorithms*. Software Foundations, Vol. 3. Electronic textbook.

[2] Gilles Barthe and Pierre Courtieu. 2002. Efficient reasoning about executable specifications in Coq. In *International Conference on Theorem Proving in Higher Order Logics*. Springer, 31–46.

[3] Avrim Blum. 1991. *Algorithms for approximate graph coloring*. Ph. D. Dissertation.

[4] Avrim Blum and David Karger. 1997. An $\tilde{O}(n^{3/14})$-coloring algorithm for 3-colorable graphs. *Information processing letters* 61, 1 (1997), 49–53.

[5] Łukasz Czajka and Cezary Kaliszyk. 2018. Hammer for Coq: Automation for Dependent Type Theory. *Journal of Automated Reasoning* 61, 1 (Jun. 2018), 423–453. https://doi.org/10.1007/s10817-018-9458-4

[6] Christian Doczkal and Damien Pous. 2020. Graph theory in Coq: Minors, treewidth, and isomorphisms. *Journal of Automated Reasoning* 64 (2020), 795–825.

[7] Georges Gonthier et al. 2008. Formal Proof—The Four-Color Theorem. *Notices of the AMS* 55, 11 (2008), 1382–1393.

[8] David Karger, Rajeev Motwani, and Madhu Sudan. 1998. Approximate graph coloring by semidefinite programming. *Journal of the ACM (JACM)* 45, 2 (1998), 246–265.

[9] Wang Shengyi. 2020. *Mechanized Verification of Graph-Manipulating Programs*. Ph. D. Dissertation. National University of Singapore (Singapore).

[10] Avi Wigderson. 1983. Improving the performance guarantee for approximate graph coloring. *Journal of the ACM (JACM)* 30, 4 (1983), 729–735.

Received 2023-07-21; accepted 2023-08-25