8-1-2024

ON THE COMPUTABLE LIST CHROMATIC NUMBER AND COMPUTABLE COLORING NUMBER

Seth Campbell Thomason
Southern Illinois University Carbondale, seth.thomason.math@gmail.com

ON THE COMPUTABLE LIST CHROMATIC NUMBER AND COMPUTABLE COLORING NUMBER

by

Seth Thomason

B.S., Southern Illinois University Carbondale, 2023

A Thesis
Submitted in Partial Fulfillment of the Requirements for the
Master of Science Degree

School of Mathematical and Statistical Sciences
in the Graduate School
Southern Illinois University Carbondale
August 2024

THESIS APPROVAL

ON THE COMPUTABLE LIST CHROMATIC NUMBER AND COMPUTABLE COLORING NUMBER

by

Seth Thomason

A Thesis Submitted in Partial

Fulfillment of the Requirements

for the Degree of

Master of Science

in the field of Mathematics

Approved by:

Dr. Wesley Calvert, Chair

Dr. Lindsey-Kay Lauderdale

Dr. Michael Sullivan

Graduate School
Southern Illinois University Carbondale
July 3, 2024

AN ABSTRACT OF THE THESIS OF

Seth Thomason, for the Master of Science degree in Mathematics, presented on July 3, 2024, at Southern Illinois University Carbondale.

TITLE: ON THE COMPUTABLE LIST CHROMATIC NUMBER AND COMPUTABLE COLORING NUMBER

MAJOR PROFESSOR: Dr. W. Calvert

In this paper, we introduce two new variations on the computable chromatic number: the computable list chromatic number and the computable coloring number. We show that, just as with the non-computable versions, the computable chromatic number is always less than or equal to the computable list chromatic number, which is less than or equal to the computable coloring number.

We investigate the potential differences between the computable and non-computable chromatic, list chromatic, and coloring numbers on computable graphs. One notable example is a computable graph for which the coloring number is 2, but the computable chromatic number is infinite.

ACKNOWLEDGMENTS

I am very grateful to Dr. Calvert for the guidance he has provided on this project, as well as the many important resources he has directed me towards. His help towards completing this thesis has been invaluable.

I would also like to thank my parents for the support that they've offered in all my educational pursuits, including my thesis. Without it, I would not have been able to pursue a Master's degree.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION TO THE COMPUTABLE VARIANTS

A computable function is a function from $\mathbb{N}$ to $\mathbb{N}$ which can be encoded in a computer program. That is, there is a computer program which when given any $n \in \mathbb{N}$, will output $f(n)$. This definition turns out to be essentially independent of the model of computation used: C, Java, Python, and almost any other programming language will allow for the exact same set of computable functions, as will formal mathematical approaches to computation such as Turing machines and lambda calculus.

The use of $\mathbb{N}$ as the domain and codomain of the function are for simplicity. Writing a natural number in binary and interpreting it as a finite binary string, we can encode any kind of data we want, including words, images, and finite mathematical structures. In fact, since programs in programming languages are written as strings of text, we can even encode computer programs as natural numbers.

A computable set is a set $S \subset \mathbb{N}$ for which there is a computer program that, when given any $n \in \mathbb{N}$, will output True if $n \in S$ and False if $n \notin S$. Equivalently, S is computable if there is a computable function f such that $n \in S \implies f(n) = 1$ and $n \notin S \implies f(n) = 0$.

Many countable sets such as $\mathbb{N} \times \mathbb{N}$ and the set $\mathcal{P}_\omega(\mathbb{N})$ of finite subsets of $\mathbb{N}$ have computable bijections with $\mathbb{N}$. Later in this paper we refer to a subset S of $\mathbb{N} \times \mathbb{N}$ being computable. This can be interpreted directly as meaning that there exists a computer program which takes 2 inputs and determines membership in S, but in order to use existing knowledge about computable functions it is convenient to associate the elements of $\mathbb{N} \times \mathbb{N}$ with natural numbers in a computable way, in which case a computable subset of $\mathbb{N} \times \mathbb{N}$ is one associated with a computable subset of $\mathbb{N}$. We will also later refer to a function $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ being computable, this means that it is associated with a computable function $f : \mathbb{N} \to \mathbb{N}$.

If $f_1, f_2, \ldots$ is a sequence of functions from $\mathbb{N}$ to $\mathbb{N}$, then an enumeration of this sequence is another function $F : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that $F(n, x) = f_n(x)$ for all n, x. A computable enumeration is a computable such function, and it is clear that only sequences of computable functions can have computable enumerations.

By a diagonal argument, there is no computable enumeration of every computable function. Despite this, it is possible to enumerate every possible computer program. In a programming language like Python, programs are just finite combinations of symbols. Most combinations of valid symbols will not produce valid programs, but any time there is an error, we can say the program returns 0. However, it is impossible to prevent programs which run infinitely without terminating.

This motivates the idea of a partial computable function, which is a partial function which can be encoded in a computer program. Input values for which the program runs forever without halting are excluded from the domain of this function. By creating a program which is capable of emulating other programs (for example, a universal Turing machine), we can create a computable enumeration of partial computable functions.

It is impossible to avoid repetition in this enumeration. In fact, every possible computable function must show up an infinite number of times.

Whenever we deal with computability, we will assume the vertex set of our graph is $\mathbb{N}$.

An n-coloring of G is a function $f : V(G) \to \{1, 2, \ldots, n\}$. It is proper if no two adjacent vertices are assigned the same color.

Some common (isomorphism classes of) graphs which we will use are as follows:

K_n is a complete graph on n vertices

$K_{m,n}$ is a complete bipartite graph with partite sets of size m and n

C_n is a cyclic graph on n vertices

$\Theta_{a,b,c}$ is a theta graph, which consists of 2 vertices connected by 3 disjoint paths of lengths a, b, and c.

Definition 1.1. *A graph G is computable if there is a computable subset of $\mathbb{N} \times \mathbb{N}$ that corresponds to the edges of G.*

Another way of thinking about this is that there is a program which can take in any two vertices and decide if they are adjacent. At first, this may seem to suggest that the graph has nice computational properties in general but this is not the case. In [1], Bean shows that there exists a computable, connected, planar graph which has a proper 3-coloring, but which has no computable, proper m-coloring for any $m \in \mathbb{N}$.

Definition 1.2. *The computable chromatic number $\chi^c(G)$ of a computable graph G is the smallest natural number m such that G has a proper, computable m-coloring. If there is no such m, we say $\chi^c(G) = \infty$.*

Recall that a coloring is a function from $V(G) = \mathbb{N}$ to $\mathbb{N}$, so a computable coloring is a computable such function. Then, for the graph given in [1], $\chi(G) = 3$ but $\chi^c(G) = \infty$.

Proposition 1.3. *For every graph G, $\chi(G) \leq \chi^c(G)$.*

This is clear as if $\chi^c(G) = k$, G has a proper, computable k-coloring, which is also a proper k-coloring, so $\chi(G) \leq k$.

Definition 1.4. *A list assignment L for a simple graph G is a function that assigns to each vertex v in G a list of colors $L(v) \in \mathcal{P}_\omega(\mathbb{N})$.*

Definition 1.5. *A proper L-coloring of G is a proper coloring f of G such that $f(v) \in L(v)$ for each $v \in V(G)$.*

List coloring was introduced independently by Erdős, Rubin, and Taylor [3] and by Vizing [7]. We follow [3] for notation. One situation where list coloring is applicable is the process of trying to finish a proper coloring of a graph in which some vertices are already fixed. Coloring a list assignment which assigns $\{1, \ldots, m\}$ to every vertex is analogous to ordinary proper coloring. In order to define the list chromatic number the sizes of the lists are identical, though merely imposing a minimum required list size is also equivalent.

Definition 1.6. *An m-assignment for G is a list assignment for G where every list has size m.*

Definition 1.7. *A graph G is m-choosable if for every m-assignment L there is a proper L-coloring of G. The list chromatic number $\chi_\ell(G)$ is the smallest $m \in \mathbb{N}$ such that G is m-choosable.*

One fact about the list chromatic number is that it is always greater than or equal to the chromatic number. This is because by choosing an m-assignment which assigns the list $\{1, 2, \ldots, m\}$ to every vertex we obtain a situation which is analogous to ordinary graph coloring.

Proposition 1.8. *For every graph G, $\chi(G) \leq \chi_\ell(G)$.*

Now that we have a definition of list chromatic number, we can define a computable analogue of it.

Definition 1.9. *A graph G is computably m-choosable if for every computable m-assignment L there is a computable proper L-coloring of G. The computable list chromatic number $\chi_\ell^c(G)$ is the smallest $m \in \mathbb{N}$ such that G is computably m-choosable.*

Again here, computable list assignments and computable L-colorings are computable as functions: recall they map $\mathbb{N} \to \mathcal{P}_\omega(\mathbb{N})$ and $\mathbb{N} \to \mathbb{N}$ respectively.

We will also introduce a computable analogue of another variant on the chromatic number: the coloring number.

Definition 1.10. *The coloring number* $\mathrm{Col}(G)$ *of a graph G is the smallest number k such that there exists a well-ordering of the vertices of G in which every vertex is preceded by at most $k-1$ of its neighbors.*

The coloring number of a graph is the smallest upper bound on the chromatic number that can be obtained using a greedy argument. Given a transfinite sequence in which

every vertex is preceded by at most $k-1$ neighbors, we can k-color the vertices in the order prescribed by the sequence, at each step choosing a different color than we used on any preceding neighbors. Thus, $\chi(G) \leq \mathrm{Col}(G)$. For the exact same reason, the coloring number is also an upper bound on the list chromatic number.

Proposition 1.11. *For every graph G, $\chi_\ell(G) \leq \mathrm{Col}(G)$.*

We continue this inequality chain one further, note that $\Delta(G)$ is the maximum degree of G.

Proposition 1.12. *For every graph G, $\mathrm{Col}(G) \leq \Delta(G) + 1$.*

Any vertex ordering will suffice to witness this.

By combining Propositions 1.8, 1.11, and 1.12, we obtain the following.

Theorem 1.13. *For every graph G, $\chi(G) \leq \chi_\ell(G) \leq \mathrm{Col}(G) \leq \Delta(G) + 1$.*

Note that this holds for graphs of any cardinality.

Later, we will show this inequality holds for the computable versions of the invariants as well.

The coloring number is also closely related to another concept called degeneracy.

Definition 1.14. *A graph G is k-degenerate if every subgraph of G has a vertex with degree at most k. The degeneracy of G, denoted $\mathrm{d}(G)$, is the largest k for which G is k-degenerate.*

For the purposes of this paper, if $\mathrm{Col}(G)$ or $\mathrm{d}(G)$ are not finite, we will simply say $\mathrm{Col}(G) = \infty$ or $\mathrm{d}(G) = \infty$.

A natural way to look at the degeneracy of a graph is to picture removing vertices of degree k or less one at a time. If the graph is eventually empty, then the graph is k-degenerate. Otherwise, it is not, and the connected components that remain are called the k-cores of the graph.

This initially may not seem related to the coloring number, but from Lick and White [6] we have the following.

Proposition 1.15. *If G is a finite graph, then* $\mathrm{Col}(G) = \mathrm{d}(G) + 1$.

The only reason they differ by one is that the definition of coloring number explicitly includes a plus one. Aside from that, the process of removing vertices adjacent to at most k vertices yet to be removed is exactly the same as the process of adding vertices adjacent to at most k vertices already placed, only in reverse. A more formal proof follows.

Proof. If G is k-degenerate, an ordering which witnesses $\mathrm{Col}(G) \leq k+1$ can be constructed backwards. Take the subgraph induced by the set of vertices not yet in the ordering, choose one with degree at most k, and place it at the beginning of the ordering.

On the other hand, if G is not k-degenerate, find a subgraph where every vertex has degree at least $k+1$. Under any vertex ordering of G, whichever element of the subgraph is last to be enumerated will have at least $k+1$ preceding neighbors. Thus, $\mathrm{Col}(G) > k+1$. □

This correspondence also gives an efficient way of calculating $\mathrm{Col}(G)$ of any finite graph G, along with finding the order witnessing the smallest value. To find $\mathrm{Col}(G)$, repeatedly remove the vertex of lowest degree from G until it is empty. The highest degree of a vertex while it is being removed is $\mathrm{Col}(G) - 1$, and an order witnessing that can be obtained by reversing the order in which the vertices were removed.

However, this does not necessarily hold for infinite graphs. A tree where every vertex has degree k will have a degeneracy of k, but a coloring number of only 2, as the ordering can be built up inductively starting at a root.

The other direction also fails. Erdős and Hajnal [4] give an example of a graph which has a coloring number of 4, but for which every finite subgraph has a coloring number of 3 or less. Jura [5] provides a helpful illustration of this graph in the Appendix, from which we can see that in any subgraph, the vertex with the lowest index will have degree at most 2. Thus, this graph has degeneracy 2 but coloring number 4.

To understand this difference better, it is helpful to introduce, from Jura [5], the linear order coloring number.

Definition 1.16. *The linear order coloring number* $\mathrm{Col}_{LO}(G)$ *of a graph* G *is the smallest* $k \in \mathbb{N}$ *such that there exists a linear ordering of the vertices of* G *in which each vertex is preceded by at most* $k-1$ *neighbors.*

For this paper, if no such k exists we will consider $\mathrm{Col}_{LO}(G) = \infty$.

Since $\mathrm{Col}(G)$ is such a minimum taken only across well-orderings, the following is clear.

Proposition 1.17. *For every graph* G, $\mathrm{Col}_{LO}(G) \leq \mathrm{Col}(G)$.

Since on a finite set, all linear orderings are well-orderings, we have the following.

Proposition 1.18. *For every finite graph* G, $\mathrm{Col}_{LO}(G) = \mathrm{Col}(G)$.

In [5], the following is proved.

Proposition 1.19. $\mathrm{Col}_{LO}(G) \leq k$ *if and only if* $\mathrm{Col}_{LO}(H) \leq k$ *for every finite subgraph* H *of* G.

The following is a little more difficult.

Proposition 1.20. *For every graph* G, $\mathrm{Col}_{LO}(G) \leq \mathrm{d}(G) + 1$.

Proof. We claim that a graph is k-degenerate if and only if there is a vertex ordering which witnesses $\mathrm{Col}_{LO}(G) \leq k+1$ whose order type is dual to a well-ordering.

Let G be k-degenerate. Form a transfinite sequence $\{v_\alpha\}$ by letting v_α be the vertex of smallest degree after the vertices $\{v_\lambda \mid \lambda < \alpha\}$ are removed. This is a vertex well-ordering in which each vertex is succeeded by at most k neighbors, so by reversing it we obtain an ordering witnessing $\mathrm{Col}_{LO}(G) \leq k+1$, whose order type is dual to a well-ordering.

Now, let G have an ordering witnessing $\mathrm{Col}_{LO}(G) \leq k+1$, whose order type is dual to a well-ordering. Let H be an arbitrary subgraph of G. Choose the vertex in H which is last in this ordering. Since it has at most k preceding neighbors, and all vertices in H precede it, its degree in H is at most k. Thus, G is k-degenerate. □

Interestingly, Col_{LO} still fits in between $\chi_\ell(G)$ and $\mathrm{Col}(G)$.

Proposition 1.21. *For every graph G, $\chi_\ell(G) \leq \mathrm{Col}_{LO}(G)$.*

Proof. Let $\mathrm{Col}_{LO}(G) = k$. Then, for every finite subgraph H of G, $\mathrm{Col}_{LO}(H) \leq k$. By Proposition 1.18, this means $\mathrm{Col}(H) \leq k$, and by Proposition 1.11, $\chi_\ell(H) \leq k$. Finally, we will later prove Proposition 5.3, which proves that since $\chi_\ell(H) \leq k$ for every finite subgraph H of G, $\chi_\ell(G) \leq k$. □

Now, we introduce a result from Erdős and Hajnal [4].

Proposition 1.22. *If every finite subgraph G' of G has $\mathrm{Col}(G') \leq k$ with $k \geq 2$, then $\mathrm{Col}(G) \leq 2k - 2$.*

By combining Proposition 1.19 and Proposition 1.18, we can see that $\mathrm{Col}_{LO}(G)$ is the maximum of $\mathrm{Col}(H)$ for every finite subgraph H of G. If no such maximum exists, $\mathrm{Col}_{LO}(G) = \infty$. Thus, we can reformulate Proposition 1.22 in the following way.

Proposition 1.23. *For every non-empty graph G, $\mathrm{Col}(G) \leq 2\mathrm{Col}_{LO}(G) - 2$.*

This bound is also tight, as [4] shows that for every $k \geq 3$ there exists a countable graph G such that $\mathrm{Col}(G) = 2k - 2$ but for every finite subgraph H of G, $\mathrm{Col}(H) = k$.

Now, we will move on to the computable coloring number. We define the computable coloring number $\mathrm{Col}^c(G)$ the same way as the coloring number, except that the vertex ordering must be a computable ordering:

Definition 1.24. *The computable coloring number $\mathrm{Col}^c(G)$ of a graph G is the least number k such that G has a computable vertex ordering $f : \mathbb{N} \to V(G)$ in which each vertex is preceded by at most $k - 1$ of its neighbors.*

Since in this paper $V(G)$ is always $\mathbb{N}$, a vertex ordering is a permutation of $\mathbb{N}$. However, we will consider $f(n)$ to be the nth vertex in the ordering for every n.

One potential problem is that the coloring number $\mathrm{Col}(G)$ quantifies over all vertex well-orders. This raises the concern that there might be a graph with $\mathrm{Col}(G) < \mathrm{Col}^c(G)$ not due to any computability properties of G, but because the vertex ordering which witnesses $\mathrm{Col}(G)$ is a well order of a type other than ω. Fortunately, from [4] we have the fact that when $|V(G)| = \kappa$, there is always an ordering of order type κ which witnesses $\mathrm{Col}(G)$. When $V(G)$ is countable, this means an ordering $f : \mathbb{N} \to V(G)$.

It would be interesting if an analogue of this result were to hold for the computable coloring number.

Question 1.25. *Let G be a graph, let α be a computable ordinal, and let $k \in \mathbb{N}$. Let $f : \alpha \to V(G)$ be any computable well-ordering of the vertices of G such that every vertex is preceded by at most k of it's neighbors. Is $\mathrm{Col}^c(G) \leq k + 1$?*

However, if this fails, it is not clear whether such an ordering would imply a way to properly, computably $(k + 1)$-color the vertices of a graph even in the case that $\alpha = \omega 2$.

Just as with the chromatic number, the computable coloring number is never less than the coloring number.

Proposition 1.26. *For every graph G, $\mathrm{Col}(G) \leq \mathrm{Col}^c(G)$.*

This is clear as $\mathrm{Col}^c(G)$ is a minimum across computable orderings, while $\mathrm{Col}(G)$ is a minimum across all orderings.

On the other hand, whether $\chi_\ell(G) \leq \chi^c_\ell(G)$ is not at all clear. In order to have $\chi^c_\ell(G) < \chi_\ell(G)$, we would need $\chi_\ell(G)$ to be witnessed only by non-computable list assignments. There would also need to be no computable $(\chi_\ell(G))$-assignments which despite being colorable are not computably colorable.

Question 1.27. *Does there exist a graph G for which $\chi^c_\ell(G) < \chi_\ell(G)$?*

These computable graph invariants are easier to understand with respect to each other, the analogue of Theorem 1.13 still holds.

Theorem 1.28. *For every graph G, $\chi^c(G) \leq \chi^c_\ell(G) \leq \mathrm{Col}^c(G) \leq \Delta(G) + 1$.*

Proof. To show $\chi^c(G) \leq \chi^c_\ell(G)$, let $k = \chi^c_\ell(G)$. Let $L(v) = \{1, \ldots, k\}$ for every v in $V(G)$. Then, $L(v)$ must have a computable coloring, which is also a computable k-coloring for G.

To show $\chi^c_\ell(G) \leq \mathrm{Col}^c(G)$, let $k = \mathrm{Col}^c(G)$, let L be a computable k-assignment for G, and let $f : \mathbb{N} \to \mathbb{N}$ be a computable vertex ordering of G in which each vertex is preceded by at most $k - 1$ neighbors. Then, we can L-color G by assigning to $f(1)$ the first color in $L(f(1))$, and assigning to $f(n + 1)$ the first color in $L(f(n + 1))$ not already used by preceding neighbors.

Finally, $\mathrm{Col}^c(G) \leq \Delta(G) + 1$ follows from the fact that in any computable vertex ordering, each vertex will clearly be preceded by at most $\Delta(G)$ of it's neighbors. □

One immediate consequence of this is that any graph with bounded vertex degree has finite computable chromatic number. Another is imposing some additional requirements on any graph G which might satisfy $\chi_\ell(G) > \chi^c_\ell(G)$. G would then satisfy $\chi(G) \leq \chi^c(G) \leq \chi^c_\ell(G) < \chi_\ell(G) \leq \mathrm{Col}(G) \leq \mathrm{Col}^c(G)$.

CHAPTER 2

SEPARATING THE COMPUTABLE INVARIANTS FROM EACH OTHER

Now that our graph invariants are related by a chain of inequalities, it is natural to ask how far each one can be separated from the next. The only case in which $\chi(G) = 1$ is for a graph with no edges, so every component of our chain must be at least 2. Thus, we seek to find examples for which one component on the chain is 2 but the next is ∞.

Proposition 2.1. *There exists a graph G with $\mathrm{Col}(G) = \mathrm{Col}^c(G) = 2$ but $\Delta(G) = \infty$.*

Proof. Let G have edge set $E(G) = \{\{1, v\} \mid v \geq 2\}$. Then, the ordering $f(n) = n$ witnesses $\mathrm{Col}^c(G) = 2$ but $\Delta(G) = \infty$. □

Proposition 2.2. *There exists a graph G with $\chi(G) = \chi^c(G) = 2$ but $\chi_\ell(G) = \chi_\ell^c(G) = \infty$.*

Proof. We create a complete bipartite graph with infinite partite sets. Let $E(G) = \{\{u, v\} \mid u \text{ odd}, v \text{ even}\}$. To show that $\chi_\ell^c(G) \neq k$ for every $k \in \mathbb{N}$, we need only look at finite subgraphs of G. From [3] we have that the graph $K_{m,m}$ is not k-choosable when $m = \binom{2k-1}{k}$. Let L be a list assignment of the subgraph of G induced by $\{1, 2, \ldots, 2m\}$ which is not k-colorable. Any computable extension of L to $\mathbb{N}$ will show $\chi_\ell^c(G) > k$.

In this case we leveraged a well known finite construction for separating $\chi(G)$ and $\chi_\ell(G)$: complete bipartite graphs. Unfortunately, there is no such construction for separating $\chi_\ell(G)$ and $\mathrm{Col}(G)$. □

Theorem 2.3. *If G is a graph with $\chi_\ell(G) = 2$, then $\mathrm{Col}(G) \leq 3$.*

It is helpful if G is connected, so we begin with the following.

Lemma 2.4. $\mathrm{Col}(G) = \sup\{\mathrm{Col}(G_\alpha)\}$*, where $\{G_\alpha\}_{\alpha \in J}$ is the set of connected components of G.*

Proof. $\mathrm{Col}(G)$ is at least as large as each G_α, as an ordering which witnesses $\mathrm{Col}(G) \leq k$ can be restricted to an ordering which witnesses $\mathrm{Col}(G_\alpha) \leq k$. We may assume J is an initial segment of the ordinals. Order the vertices of G first by the connected component G_α they belong to, and then by their position in an ordering which witnesses $\mathrm{Col}(G_\alpha)$. The result is a vertex well-ordering which witnesses $\mathrm{Col}(G) = \sup\{\mathrm{Col}(G_\alpha)\}$. □

Now, we move on to proving Theorem 2.3.

Proof. We may assume G is connected, otherwise analyze its connected components. In [3], it is proved that for finite connected graphs H, $\chi_\ell(H) = 2$ if and only if H is either 1-degenerate or has a 1-core of C_{2n+2} or $\Theta_{2,2,2n}$ for some $n \geq 1$.

Because these cores are themselves 2-degenerate, this is sufficient when G is finite. Infinite graphs with $\chi_\ell(G) = 2$ may not be k-degenerate for any k at all, for example a tree where every vertex has $(k+1)$ neighbors, so we need to modify our approach.

If G has no cycles, then it is a tree. We will prove later in Proposition 3.1 that this means $\mathrm{Col}(G) = 2$, so we are done.

G cannot have infinitely many cycles. Suppose it does, choose 4 of these cycles arbitrarily. Let G' be a finite subgraph of G containing these cycles along with paths connecting them if they are not already connected. G' is a finite connected graph with $\chi_\ell^c(G) = 2$, so the 1-core of G' must be one of the permitted 1-cores. However, every cycle in G' will remain in the 1-core of G', and all of the permitted 1-cores have less than 4 cycles.

If G has finitely many cycles, we can consider a finite subgraph G' which contains every cycle of G, along with paths connecting them. Now the 1-core of G' is precisely the subgraph induced by vertices of G which are contained in cycles, showing that the cycles of G-form an allowed 1-core.

Now, we modify G by identifying this entire core as a single vertex, which turns G into a tree T. We fix this vertex as the root of our new tree, and give an ordering witnessing $\mathrm{Col}(T) = 2$ starting with our root as prescribed by Proposition 3.1. By replacing the root

vertex in this ordering by an ordering for the cycles of G which witnesses a coloring number of 3, we obtain a vertex ordering for G which shows $\mathrm{Col}(G) \leq 3$. □

This opens the door for some questions.

Question 2.5. *For what values $k \geq 3$ does $\chi_\ell(G) = k$ imply an upper bound on $\mathrm{Col}(G)$?*

Thanks to Proposition 1.22 we can restrict our view to finite graphs, an upper bound for the coloring number of finite k-choosable graphs automatically gives a higher upper bound for k-choosable infinite graphs.

Question 2.6. *Does $\chi_\ell^c(G) = 2$ imply any upper bound on $\mathrm{Col}^c(G)$?*

CHAPTER 3

SEPARATING THE ORDINARY INVARIANTS FROM THE COMPUTABLE INVARIANTS

Now we will look at another kind of separation: separating the graph invariants from their computable counterparts. The strongest way to do this is to find a graph where $\mathrm{Col}(G) = 2$ but $\chi^c(G) = \infty$.

Fortunately, there is a construction based on [1] which works for this purpose. The graph in the paper has $\chi(G) = 3$ and $\chi^c(G) = \infty$, but by performing a similar construction, we can create a forest with $\chi^c(G) = \infty$, at the cost of disconnecting the graph.

Proposition 3.1. *A graph G is a forest if and only if* $\mathrm{Col}(G) = 2$.

Proof. Fix a root of every component of G. Let $(S_0, \leq_0)$ be the set of roots equipped with a well-order. For $n \in \mathbb{N}$, let $(S_n, \leq_n)$ be the set of vertices at distance n from a root equipped with a well-order. Then we well-order the vertices of G by ordering u before v if u is closer to a root than v, or if they are both in the same S_n, ordering them according to $\leq_n$. Then, each vertex v has at most one preceding neighbor: the first step in the path from v to a root.

The other direction is easy. Suppose G had a cycle, then for any vertex ordering the vertex in the cycle which is last is adjacent to both of its neighbors in the cycle. □

Theorem 3.2. *There exists a computable graph G with* $\mathrm{Col}(G) = 2$ *but* $\chi^c(G) = \infty$.

Proof. The following proof is a modification of Bean's proof that there exists a connected, planar, computable graph with chromatic number 3 but infinite computable chromatic number [1]. The construction is also very similar to one from Jura [5], which shows that there is a computable forest for which *any* computable linear ordering must have vertices preceded by arbitrarily many neighbors.

Consider a game between two players: Player 1 and Player 2. The game starts with a set of $n!$ disjoint vertices for some $n \in \mathbb{N}$. Player 2 will color these vertices, and then Player 1 will introduce new vertices, connected to already existing vertices in a way that doesn't form a cycle. Each time Player 1 introduces a new vertex, Player 2 must immediately color it.

Player 1's goal is to force Player 2 to use as many colors as possible, and indeed they can force at least $n+1$ different colors: First, Player 1 will divide the vertices into n groups of size $(n-1)!$ and force Player 2 to use n colors for each group. In order to avoid immediate loss, Player 2 must use the same n colors for every group. Player 1 then wins by creating a vertex and connecting it to a vertex of color i in the ith group for every $1 \leq i \leq n$.

Now, to use this game to construct a graph, fix a computable enumeration of partial computable functions $\{\varphi_e \mid e \in \mathbb{N}\}$. We will construct a graph G in stages: at stage s, start by adding a group of $s!$ new vertices to the graph. The subgraph we will eventually build off these vertices, we will call G_s. For each $i \leq s$, run φ_i for s steps on every vertex of G_i. φ_i will play as player 2, and we play as player 1, meaning we do not take any action until every vertex in G_i is colored, at which point we add an additional vertex into G_i and connect it according to the winning strategy we devised earlier.

This completes the construction. Suppose that G had a computable k-coloring f. We find the first φ_e which is equal to f and for which $e \geq k$. (Note that there are infinitely many such φ_e). The subgraph G_e has been constructed so that it is impossible for φ_e to have colored it using less than $e+1$ colors, which is a contradiction.

G is also computable: to decide if an edge $\{u, v\}$ with $u < v$ is in G, follow the construction of the graph until v is added and check if it is connected to u at that moment. This works because edges between preexisting vertices are never added at later stages. $\square$

Corollary 3.3. *There exists a computable connected graph G with $\mathrm{Col}(G) = 3$ but $\chi^c(G) = \infty$.*

Proof. Take the construction from Theorem 3.2, add a single extra vertex, and connect it to every other vertex. Now, $\mathrm{Col}(G) = 3$ as the graph clearly contains cycles, and we can take any ordering witnessing $\mathrm{Col}(G) = 2$ and add our extra vertex at the beginning to witness $\mathrm{Col}(G) = 3$. □

This is also the best we can do for connected graphs, based on the following well-known result

Proposition 3.4. *If G is computable, connected and $\chi(G) = 2$, then $\chi^c(G) = 2$.*

Proof. Fix a vertex v, color it 1, and color every other vertex w by finding a path from v to w and coloring w with 1 if the path length is even and 2 if it is odd. □

This can be slightly generalized as follows

Corollary 3.5. *If G is computable, composed of finitely many connected components, and $\chi(G) = 2$, then $\chi^c(G) = 2$.*

Proof. By fixing a vertex in every connected component, we can compute which component an arbitrary vertex is in by searching until we find a path to a fixed vertex. Then, we choose colors for all our fixed vertices and proceed the same way as with Proposition 3.4. □

Interestingly, this turns out to hold for the list chromatic number and coloring number as well.

Proposition 3.6. *If G is computable, composed of finitely many connected components, and $\mathrm{Col}(G) = 2$, then $\mathrm{Col}^c(G) = 2$.*

Proof. From Proposition 3.1 we know that G must be a forest. Fix a root in each tree in the forest. Start the ordering by listing every root. Then, repeatedly take the vertex of smallest index which has not been listed yet, find a path connecting it to a root, and append this path to the current list, starting with the first vertex in the path which hasn't already been listed and ending with the vertex of smallest index. This computable ordering witnesses $\mathrm{Col}^c(G) = 2$. □

Proposition 3.7. *If G is computable, composed of finitely many connected components, and $\chi_\ell(G) = 2$, then $\chi_\ell^c(G) = 2$.*

Proof. In the proof of Theorem 2.3, we show that every connected graph with $\chi_\ell(G) = 2$ must either have no cycles or a finite connected structure of cycles. Given a list assignment L, we can computably L-color every cycle in every connected component, since this requires only a finite amount of information. Then, as in the proof of Theorem 2.3, we can identify each structure of cycles as a single vertex. Then, repeatedly take the vertex of smallest index which has not been colored yet, find a path connecting it back to a cycle, and color the vertices of this path greedily, starting with the first vertex which hasn't been colored yet and ending with the vertex of smallest index. This proves $\chi_\ell^c(G) = 2$. □

CHAPTER 4

SEPARATIONS WITH FINITE GAPS

Another way to examine the invariants is to try creating finite gaps.

Proposition 4.1. *For any m and n with $n \geq m \geq 2$ there exists a computable graph G with $\chi(G) = \chi^c(G) = m$ and $\chi_\ell(G) = \chi^c_\ell(G) = n$.*

To prove this, we will need the following well known example in list coloring

Lemma 4.2. *K_{n,n^n} has chromatic number 2, but list chromatic number $n+1$.*

Proof. That K_{n,n^n} is $(n+1)$-choosable comes from $\mathrm{Col}(G) = n+1$, witnessed by any ordering which starts with the size n partite set. That K_{n,n^n} is not n-choosable comes from a list assignment L which assigns disjoint sets to every vertex in the size n partite set. Then, there are n^n possible ways to L-color these vertices. Then, each vertex in the size n^n partite set can be made to restrict exactly one of these colorings. □

For example, in $K_{3,27}$, we can assign the vertices in the size 3 set the lists $\{1,2,3\}, \{4,5,6\}, \{7,8,9\}$. Then, each vertex in the size 27 set would be assigned a list consisting of one element in each of $\{1,2,3\}, \{4,5,6\}$, and $\{7,8,9\}$.

This achieves $\chi(G) = 2, \chi_\ell(G) = n$. To achieve $\chi(G) = m, \chi_\ell(G) = n$ we can take the disjoint union of K_{n,n^n} and K_m. To make our graph infinite, we can add an infinite number of isolated vertices. Since the graph is finite, we don't need to worry about the computable variants being different.

Proposition 4.3. *For any m and n with $n \geq m \geq 2$ there exists a computable graph G with $\mathrm{Col}(G) = \mathrm{Col}^c(G) = m$ and $\Delta(G) + 1 = n$.*

Proof. Let $G = K_{m-1,n-1}$. Using an ordering which starts with the size $m-1$ set, we find $\mathrm{Col}(G) \leq m$. Since the minimum degree of G is $m-1$, the last vertex in any ordering

must have at least $m-1$ preceding neighbors, so $\mathrm{Col}(G) \geq m$. Thus, $\mathrm{Col}(G) = m$, and $\Delta(G)+1 = n$. □

Every invariant can be separated from it's computable counterpart using the same modification of the construction in Theorem 3.2.

Proposition 4.4. *For any m and n with $n \geq m \geq 2$ there exists a computable graph G with $\chi(G) = \chi_\ell(G) = \mathrm{Col}(G) = m$ and $\chi^c(G) = \chi^c_\ell(G) = \mathrm{Col}^c(G) = \Delta(G)+1 = n$.*

Proof. Repeat the construction used in Theorem 3.2 with the following modification: Instead of adding $s!$ new vertices at stage s, we add $(n-1)!$. This guarantees that Player 2 is forced to use n colors, so for any φ_e which is a computable $(n-1)$-coloring of G, there is a subgraph G_e on which it is not proper. Thus, $\chi^c(G) \geq n$. The final vertex in any completed subgraph G_e will be adjacent to $n-1$ previously placed vertices, and every vertex before it will be adjacent to at most $n-2$ previously placed vertices and at most 1 vertex which will be placed in the future. Thus, $\Delta(G)+1 = n$. Once again, if $m = 2$ we are finished, and if not, take the disjoint union with K_m. □

CHAPTER 5

ADDITIONAL REMARKS

So far, we have found examples of graphs which separate the computable invariants from each other in a way unrelated to computability, and we have found one graph which maximally separates the invariants from the computable invariants.

Proposition 5.1. *There exists a graph G with $\chi(G) = 2$, $\chi^c(G) = \infty$, and $\chi_\ell(G) = \infty$.*

Proof. From Proposition 2.2 we can take a graph G_1 with $\chi(G_1) = 2$ and $\chi_\ell(G_1) = \infty$. From Theorem 3.2 we can take a graph G_2 with $\chi(G_2) = 2$ and $\chi^c(G_2) = \infty$. Then, the disjoint union G of G_1 and G_2 will satisfy $\chi(G) = 2, \chi^c(G) = \infty$, and $\chi_\ell(G) = \infty$. □

This graph, which is easy to color, but hard to list color and hard to computably color comes as no surprise. Much more interesting would be a graph which is easy to computably color and easy to list color, but hard to computably list color.

Question 5.2. *Does there exist a graph G for which $\chi^c(G)$ and $\chi_\ell(G)$ are finite but $\chi^c_\ell(G) = \infty$? If so, how small can $\chi^c(G)$ and $\chi_\ell(G)$ be?*

The De Bruijn–Erdős theorem states that if every finite subgraph of G is k-colorable, then G is also k-colorable.

Proposition 1.22, and the fact that that bound is tight are results of Erdős and Hajnal [4], which achieve the best possible equivalent result for the coloring number.

For list coloring, the De Bruijn–Erdős theorem generalizes in it's full strength.

Proposition 5.3. *If G is a graph and every finite subgraph of G is k-choosable ($k \in \mathbb{N}$), then G is k-choosable.*

Proof. Let L be a k-assignment to G. Let $X = \prod_{v \in V(G)} L(v)$ be a topological space with the product topology, where each $L(v)$ is given the discrete topology. Then the elements

of X correspond to L-colorings of G, and X is compact by Tychonoff's theorem. For every edge $e = (u, v)$ in G, let $X_e = \{x \in X \mid x_u \neq x_v\}$. Then,

$$X - X_e = \bigcup_{c \in L(u) \cap L(v)} \{x \in X \mid x_u = c \wedge x_v = c\}$$

Since this is a union of open sets it is open, and hence every X_e is closed.

Note that X_e is the set of colorings which satisfy the edge e. If $S \subset V(G)$ is finite then $\bigcap_{e \in S} X_e$ is the set of L-colorings that satisfy the finite set of edges S. This set is non-empty because restricting L to the subgraph induced by S creates a k-assignment of the subgraph, which can be colored since the subgraph must be k-choosable. Then, the collection $\{X_e\}_{e \in E(G)}$ has the finite intersection property. Since X is compact, that means it has non-empty intersection. Then, any element of this intersection corresponds to a proper L-coloring of G. □

This allows us to find another example of a graph for which $\chi_\ell(G) \leq 5$ but $\chi_\ell^c(G) = \infty$ without doing any construction ourselves. From Dörre [2], we have that every finite planar graph is 5-choosable. Combining this with Proposition 5.3 gives us that infinite planar graphs are also 5-choosable. In [1], a planar graph with $\chi^c(G) = \infty$ is constructed, so this graph must have $\chi_\ell(G) \leq 5$ and $\chi_\ell^c(G) = \infty$.

REFERENCES

[1] Dwight R. Bean. Effective coloration. *The Journal of Symbolic Logic*, 41(2):469–480, 1976.

[2] Peter Dörre. Every planar graph is 4-colourable and 5-choosable a joint proof. *Fachhochschule Südwestfalen (University of Applied Sciences)(unpublished note)*, 2004.

[3] Paul Erdős, Arthur L. Rubin, and Herbert Taylor. Choosability in graphs. In *Proceedings of the West Coast Conference on Combinatorics, Graph Theory and Computing (Humboldt State Univ., Arcata, Calif., 1979)*, volume XXVI of *Congress. Numer.*, pages 125–157. Utilitas Math., Winnipeg, MB, 1980.

[4] P. Erdős and A. Hajnal. On chromatic number of graphs and set-systems. *Acta Mathematica Academiae Scientiarum Hungaricae*, 17(1–2):61–99, 3 1966.

[5] Matthew A Jura. *Reverse Mathematics and the coloring number of graphs*. University of Connecticut, 2009.

[6] Don R. Lick and Arthur T. White. k-degenerate graphs. *Canadian Journal of Mathematics*, 22(5):1082–1096, 1970.

[7] Vadim G Vizing. Coloring the vertices of a graph in prescribed colors. *Diskret. Analiz*, 29(3):10, 1976.

VITA

Graduate School
Southern Illinois University Carbondale

Seth Thomason
seth.thomason.math@gmail.com

Southern Illinois University Carbondale
Bachelor of Science, Mathematics, August 2023

Thesis Paper Title:
On the Computable List Chromatic Number and Computable Coloring Number

Major Professor: Dr. W. Calvert

Publications:

Kaul, Hemanshu & Maxfield, Michael & Mudrock, Jeffrey & Thomason, Seth. (2023). The DP color function of clique-gluings of graphs. Enumerative Combinatorics and Applications. 4. Article #S2R11. 10.54550/ECA2024V4S2R11.

Bui, Manh & Kaul, Hemanshu & Maxfield, Michael & Mudrock, Jeffrey & Shin, Paul & Thomason, Seth. (2023). Non-chromatic-Adherence of the DP Color Function via Generalized Theta Graphs. Graphs and Combinatorics. 39. 10.1007/s00373-023-02633-z.

Becker, Jack & Hewitt, Jade & Kaul, Hemanshu & Maxfield, Michael & Mudrock, Jeffrey & Spivey, David & Thomason, Seth & Wagstrom, Tim. (2022). The DP color function of joins and vertex-gluings of graphs. Discrete Mathematics. 345. 113093. 10.1016/j.disc.2022.113093.

Mudrock, Jeffrey & Thomason, Seth. (2021). Answers to Two Questions on the DP Color Function. The Electronic Journal of Combinatorics. 28. 10.37236/9863.