FINDING DOMATIC PARTITIONS IN INFINITE REGULAR GRAPHS

MATTHEW JURA, OSCAR LEVIN, AND TYLER MARKKANEN

Abstract. We investigate the apparent difficulty of finding domatic partitions in graphs using tools from computability theory. We consider nicely presented (i.e., computable) infinite regular graphs and show that even if the domatic number is known, there might not be any algorithm for producing a domatic partition of optimal size. However, smaller domatic partitions can be constructed. We consider various approaches to this question. Additionally, we establish similar results for total domatic partitions.

1. Introduction

A set D of vertices in a graph G is called *dominating* if every vertex of G not in D is adjacent to a vertex in D. A partition of the vertices into n dominating sets is called a *domatic n-partition*, and the largest such n is called the *domatic number* of the graph. Not surprisingly, finding the domatic number of a graph is non-tractable—deciding whether a graph has domatic number 3 is NP-complete [2]. Thus actually producing a domatic partition of optimal size is computationally hard. The goal of this paper is to understand *why* domatic partitions are so difficult to produce.

We will consider infinite regular undirected graphs, and ask whether it is possible to find domatic partitions of various sizes. To do this, the graphs will need to be nicely presented; our approach will be to use tools from computability theory. By "nicely presented" graphs we will mean *computable* graphs and we will ask whether there exists a *computable* domatic partition. The relevant definitions and background in computability theory will be given in Section 2, but essentially *computable* means "there is an algorithm which gives. . . ." For more of a complete background on computability theory, please see [7] and [5]. Since the finite version of this question is NP-hard, it is not surprising that, in general, computable graphs with domatic number $n \geq 3$ might not even have a computable domatic 3-partition (every such graph does have a computable domatic 2-partition—these results can be found in [4]). The general case though allows for graphs in which we might not know how many neighbors a particular vertex has. In this investigation we will thus restrict our focus to *regular* graphs (graphs in which every vertex has the same degree). This restriction would appear to make it *easier* to find domatic partitions—not just because we are restricting the class of graphs we consider, but also because, in attempting to build a domatic partition, we have fewer places where we can make a mistake. However, we will show in Theorem 3.1 that there are computable regular graphs with domatic number n but no computable domatic n-partition.

2010 *Mathematics Subject Classification.* 05C69, 05C63, 05C85, 03D80.

Key words and phrases. domatic partitions, graph algorithms, infinite regular graphs, computability theory.

Computability theory has proved useful in understanding the complexity of a large number of other graph invariants. For example, deciding the chromatic number of a graph is NP-complete, and producing a proper vertex coloring (i.e., an independent partition) of optimal size is computationally difficult. Bean [1] established that there are computable graphs with chromatic number as small as 2 which contain no computable proper vertex coloring of any finite size. However, if one requires that the degree of each vertex is also computable, then it is possible to find computable proper vertex colorings, albeit not of optimal size (the coloring can be as much as one less than twice the chromatic number), a result of Schmerl [6]. Both of these results pertain directly to infinite graphs, but still tell us something about why coloring graphs in general is difficult. The first result (and in particular its proof) shows that, in a strong sense, we must consider the entire graph before being confident that we have not made a mistake in our coloring. The mistakes we might make can be arbitrarily bad, requiring us to use more and more colors. However, knowing how many neighbors a vertex has is indeed useful information, even if it is not enough to give a coloring of smallest possible size.

Mirroring the research for chromatic number, we ask whether there might be computable domatic partitions of size *smaller* than the domatic number (but larger than the trivial 2-partition). In Theorem 4.3 we give a procedure for producing a (computable) domatic n-partition in any computable k-regular graph with domatic number $k + 1 \geq n^2$. Whether this gap is the best-possible is not known. Indeed, for $n \leq 4$ we can do slightly better. Interestingly, the regularity of the graphs seem to both help and hinder our search for the resolution of this question.

We will begin in Section 2 by recalling some graph theoretic definitions and introducing the relevant concepts from computability theory. Then in Section 3, we will prove that it is not always possible to compute a domatic n-partition in computable regular graphs with domatic number n. We will also give an analogous result for *total* domatic partitions, a concept that is defined in that section. We will then turn to the question of whether it is easier to compute *smaller* domatic partitions for these graphs. In Section 4 we show how to compute sub-optimal domatic partitions, and consider what these might look like. In particular, we will prove that they are not *divisible*, that is, they are not the result of taking the union of the sets in a full domatic partition. In Section 5 we conclude with some ideas for future research.

2. Preliminaries

Recall the following definitions from graph theory. A set of vertices D in a graph G is *dominating*, if each vertex of G not in D is adjacent to a vertex from D. It is always possible to partition the vertices of G into (disjoint) dominating sets and such a partition is called *domatic*. Since we will consider issues of computability, we give an equivalent definition. A *domatic n-partition* of a graph G is a function $f : V \to \{1, 2, \ldots, n\}$, such that each $D_i = \{v \in V \mid f(v) = i\}$ (with $1 \leq i \leq n$) is a dominating set in G, and the collection of these D_i partitions V. The largest n for which G admits a domatic n-partition is called the *domatic number of* G, denoted $d(G)$. As is tradition, we will usually say that f assigns "colors" to vertices. If a vertex v is either in D_i or adjacent to a vertex in D_i, we will say that v is *dominated* by color i.

When there exists a domatic n-partition, we wish to know if it is a computable function. Formally, a function $\varphi : \mathbb{N} \to \mathbb{N}$ is *computable* if there is some Turing machine, which, on input n, halts and outputs $\varphi(n)$. By the Church-Turing thesis, this is equivalent to the existence of an informal algorithm which outputs $\varphi(n)$ on input n, and this is the notion of computability that we will generally adopt. Notice that our computable functions have no time or space restrictions, except that both must be finite for each individual input. Of course, there are some algorithms that do not halt on some inputs. Thus in general we consider *partial computable functions.* We still write $\varphi : \mathbb{N} \to \mathbb{N}$, although the domain need not be all of the natural numbers, and for those numbers n in the domain, we write $\varphi(n)\downarrow$ and say that φ *halts* or *converges* on n. Otherwise we write $\varphi(n)\uparrow$ and say that φ *diverges* on n. If a partial computable function converges on all natural numbers, we say it is *total computable* or simply *computable.*

The advantage of considering partial computable functions is that we can effectively list all of them: $\{\varphi_0, \varphi_1, \ldots\}$ (there is a universal Turing machine, or alternatively, think of enumerating all of the countably many algorithms in some fixed programming language). The existence of this effective list of all partial computable functions will allow us to prove that a particular graph does not contain a computable domatic k-partition, by showing that φ_e is not a domatic k-partition for any $e \in \mathbb{N}$—either because $\varphi_e(n)\uparrow$ for some n, or because $\varphi_e(n)\downarrow$ but is "wrong." That is, some vertex is not dominated because of the value of $\varphi_e(n)$. This process is usually called *diagonalizing* against all computable functions, as each partial computable function need only fail on one input (much as happens in Cantor's diagonalization proof).

Intuitively, a graph should be computable if there is an algorithm which tells us when two vertices are adjacent. A little more formally, we take $V = \mathbb{N}$ (although often still refer to vertices as $v_0, v_1, \ldots$) and require that $E \subseteq V \times V$ is computable as a set (that is, its characteristic function is computable).

Note that while this is a natural definition for a graph being computable, many basic properties of a computable graph might not be computable. In particular, there need not be an effective procedure for finding all the neighbors of a given vertex; we might have no way to know when to stop looking for another adjacent vertex. However, all our graphs will be regular, and for these graphs it is possible to compute all the neighbors of a given vertex. For suppose we are promised that each vertex has degree k. Then to find the neighbors of v, we simply ask whether $(v, v_i) \in E$ for each $i \in \mathbb{N}$, one at a time, in order. Since the graph is computable, we will get an answer (the characteristic function will halt) for each i. Eventually, we will get a *positive* answer k times, and at this point we stop and have our list of k neighbors. Regularity is just one way we might happen to have a procedure for finding all neighbors—in general, graphs in which finding the degree of vertices is also computable are called *highly computable.*

In the next section we will build computable graphs in stages, and we will refer to these as constructions. At each stage we will add some vertices to the graph. As soon as we do this, we must say exactly to which of the previously mentioned vertices the new vertices are adjacent. As long as we do this at each stage, the resulting graph will be computable—the algorithm for deciding whether $(v_i, v_j) \in E$ is simply to run through the construction of our graph until a stage at which both

v_i and v_j are mentioned, and check whether we said they were adjacent by that stage.

While building the graph, we will also be waiting for each φ_e to converge on some fixed finite set of vertices so that we can ensure φ_e is not a domatic partition. If φ_e never converges, we "win" for free (in the sense that φ_e can never be a domatic partition of the graph). If φ_e does converge, then we build the next part of the graph in a way to "defeat" φ_e. But because the graph must be computable, we cannot add any new edges between vertices that were put into the graph previously. We could add a new vertex adjacent to one previously mentioned, except we might run into issues with the regularity requirement. If the plan was to wait for φ_e to converge on v_i, and then add a vertex adjacent to v_i, we run the risk that φ_e will appear never to converge on v_i at all. In that case we would need to enumerate all the neighbors of v_i by some finite stage. After that stage, we might discover that we were wrong, that is, $\varphi_e(v_i)\downarrow$, and we now cannot put any more vertices adjacent to v_i without breaking regularity. For this reason, our constructions will only add vertices adjacent to other recently-enumerated vertices. In particular, the methods we use to diagonalize against all φ_e must account for the fact that a proposed domatic partition on some small subset of vertices can influence how vertices that are arbitrarily far away must be partitioned.

3. Computable Domatic Number

Define the *computable domatic number* $d^c(G)$ to be the largest n for which there is a computable domatic n-partition of G (i.e., the domatic n-partition is a computable function). Given a computable graph G there is no reason to think that $d(G) = d^c(G)$. Indeed, in general, there are computable graphs with $d(G) = n$ but $d^c(G) = 2$ for each $n \geq 3$, as shown in [4]. These graphs are not regular, however. We begin by showing that even for regular graphs, we might have $d(G) > d^c(G)$.

Theorem 3.1. *For each $n \geq 3$, there is an $(n-1)$-regular graph G with $d(G) = n$ and $d^c(G) = n - 1$.*

The proof relies on our ability to generate a *chain* of vertices in which we can guarantee that certain vertices must be colored identically.

Definition 3.1. A K_n^-*-link* is the graph on vertices $\{l, r, v_1, \ldots, v_{n-2}\}$ with edges connecting all pairs of vertices except l and r (in other-words, we have $K_n - (l, r)$). We call l a *left-outer vertex*, r a *right-outer vertex*, and the vertices $\{v_1, \ldots, v_{n-2}\}$ all *inner vertices*. A K_n^-*-chain* consists of a sequence of K_n^--links in which each right-outer vertex of one link is adjacent to the left-outer vertex of the next link. The number of links in a chain is its *length*. For example, see Figure 1.

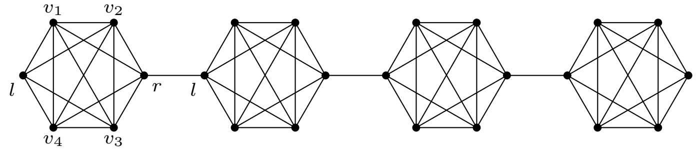

Figure 1. A K_6^--chain of length 4.

Lemma 3.1. *Suppose a graph with domatic number n contains a K_n^--chain. Then in any domatic n-partition, all left-outer vertices in the same chain must be colored identically to each other and distinctly from all the right-outer vertices in the same chain, which themselves must also be colored identically.*

Proof. Notice first, each vertex in the chain has degree $n-1$ (we assume that either the first left-outer vertex and last right-outer vertex are each adjacent to exactly one vertex outside the chain or else that the chain is infinite). Thus in any domatic n-partition, no vertex can be adjacent to two vertices of the same color, and as such each link must contain exactly one vertex of each color. In the first link, say the left-outer vertex is colored red, and the right-outer vertex is colored blue. None of the inner vertices can be colored red or blue, so within the first link, the right-outer vertex is dominated by every color except red. Thus the left-outer vertex of the next link must be red again. Now this vertex is already dominated by red and blue, so the inner vertices of the second link must be the other $n-2$ colors. Each of the inner vertices will be dominated by all colors except blue, forcing the right-outer vertex of the second link to be blue again. We can inductively extend this argument to the entire chain. □

We are now ready to prove Theorem 3.1.

Proof of Theorem 3.1. We build the desired $(n-1)$-regular graph G and a computable domatic $(n-1)$-partition of G, while at the same time diagonalizing against all partial computable functions φ_e, which we interpret as candidates for a computable domatic n-partition of G. The graph will be built in stages; at each stage s we will have a finite graph G_s consisting of some number of K_n^--chains. In fact, G_s will be $\bigcup_{e\leq s} T_{e,s}$, where each $T_{e,s}$ is either one or two finite K_n^--chains which we have built for the purpose of ensuring that φ_e is not a domatic n-partition. Each stage of the construction will consist of increasing the length of each $T_{e,s}$, adding a new $T_{e,s}$, and possibly connecting the two K_n^--chains in a particular $T_{e,s}$ to form one longer K_n^--chain. Our final graph will be $G = \bigcup_{s\in\mathbb{N}} G_s = \bigcup_e T_e$, consisting of infinitely many K_n^--chains each of infinite length (where each $T_e = \bigcup_{s\geq e} T_{e,s}$ is either one or two infinitely long K_n^--chains).

In order to diagonalize against each φ_e, we will focus on four vertices in each $T_{e,s}$. When first initialized as $T_{e,e}$, this graph will consist of two disjoint K_n^--chains each of length 1 (that is a single K_n^--link). Call the left-outer vertices of these two links l_e^1 and l_e^2, and the right-outer vertices of these two links r_e^1 and r_e^2, respectively.

Construction: At stage 0, initialize $G_0 = T_{0,0}$ as two disjoint K_n^--chains each of length 1, and then move on to stage 1. At any stage $s \geq 1$, assume that we have already defined $G_{s-1} = \bigcup_{e<s} T_{e,s-1}$, where each $T_{e,s-1}$ is either one or two K_n^--chains (of some finite length). Proceed as follows:

(1) Initialize $T_{s,s}$ as two disjoint K_n^--chains each of length 1.
(2) For each $e < s$ for which $T_{e,s-1}$ consists of two chains, run φ_e on vertices $\{l_e^1, l_e^2, r_e^1, r_e^2\}$. Do nothing, unless φ_e halts on all four of these inputs, in which case:
    (a) If $\varphi_e(l_e^1) = \varphi_e(l_e^2)$, add a single link to connect the two chains of $T_{e,s-1}$, connecting the right ends of both chains, as in Figure 2.
    (b) If $\varphi_e(l_e^1) \neq \varphi_e(l_e^2)$, add a single link to connect the two chains of $T_{e,s-1}$, connecting the right end of one chain to the left end of the other, as in Figure 3.

(3) For each $e < s$, form $T_{e,s}$ by adding a single link to each end of all (one or two) of the K_n^--chains in $T_{e,s-1}$. Move on to stage $s + 1$.

This completes the construction.

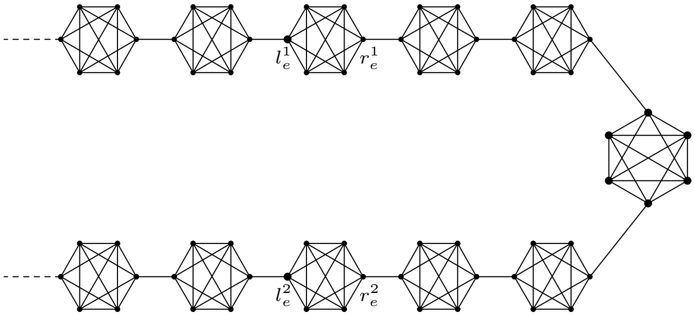

Figure 2. T_e if $\varphi_e(l_e^1) = \varphi_e(l_e^2)$ after φ_e has halted.

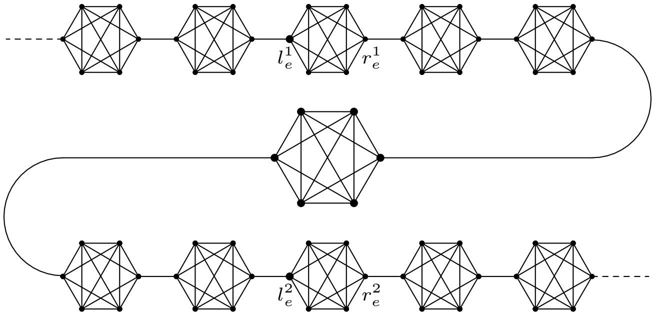

Figure 3. T_e if $\varphi_e(l_e^1) \neq \varphi_e(l_e^2)$ after φ_e has halted.

Verification: First, note that G is indeed $(n-1)$-regular and has domatic number n, since it consists of infinitely many K_n^--chains of infinite length. Also, G is clearly computable, since to decide whether two vertices are adjacent, we simply wait until they are both used in the construction, at which point their adjacency is established forever.

To see that G contains a computable domatic $(n-1)$-partition, notice that if we color all outer vertices with color 1, and the $n-2$ inner vertices of each link with the remaining $n-2$ colors, we will produce a domatic $(n-1)$-partition. This coloring can be done computably for this graph: as soon as a vertex is enumerated, its status as either an inner or outer vertex is determined, so it can be colored accordingly. Connecting two chains in the construction can change a left-outer vertex to a right-outer vertex, but never an outer vertex to an inner vertex.

Finally, we argue that G does not have a computable domatic n-partition. For suppose it did. Then the computable domatic n-partition would be φ_e for some e. There would be some finite stage at which φ_e halts on all four of l_e^1, l_e^2, r_e^1 and r_e^2. So at this stage, the two K_n^--chains for φ_e become a single K_n^--chain. In this single chain, l_e^1 is a left-outer vertex, so by Lemma 3.1, φ_e must eventually color all other left-outer vertices of T_e with the same color as l_e^1 and all right-outer vertices of T_e with a different color than l_e^1. However, φ_e has already colored l_e^2. If l_e^1 and l_e^2 were colored identically, then we connected the right ends of the chains, which makes l_e^2 a right-outer vertex of the single chain. If l_e^1 and l_e^2 were not colored identically, then we connected the chains in such a way that l_e^2 is still a left-outer vertex. In either case, we see that φ_e cannot be a domatic n-partition. □

One way you might try to find a domatic partition is to assign colors to vertices until you get stuck, then backtrack a bit, changing colors to help dominate "problem" vertices. Of course, changing a vertex might create a new problem elsewhere. In fact, you might cause one of the neighbors of the vertex in question to no longer be dominated, and the vertex itself might become undominated. With this as motivation, we consider *total domatic partitions*, in which a vertex can never help dominate itself.

Specifically, a *total dominating set* is a subset of V such that every vertex of the graph is adjacent to a vertex in the set. A *total domatic partition* is a partition of the vertices into total dominating sets, and the size of the largest total domatic partition is called the *total domatic number*, denoted $d_t(G)$. Of course, for computable graphs, we can also ask about the computable total domatic number, $d_t^c(G)$. We get an analogous result to Theorem 3.1.

Theorem 3.2. *For every $n \geq 3$, there is a computable n-regular graph G with $d_t(G) = n$ but $d_t^c(G) = n - 1$.*

The proof of this theorem is similar to that Theorem 3.1. However, in the proof of this theorem, we generate a different sort of chain. Our proof relies on an important combinatorial fact about how these chains must be colored in any total domatic partition.

Definition 3.2. Define a *double K_n-link* (for $n \geq 3$) to be the graph consisting of two complete graphs on vertices $\{v_1, \ldots, v_n\}$ and $\{w_1, \ldots, w_n\}$ respectively with additional edges (v_i, w_i) for $3 \leq i \leq n$. A *double K_n-chain* consists of a sequence of double K_n-links in which the vertex v_2 of each link is adjacent to the v_1 in the next link, and the w_2 of each link is adjacent to the w_1 of the next.

For example, see Figure 4.

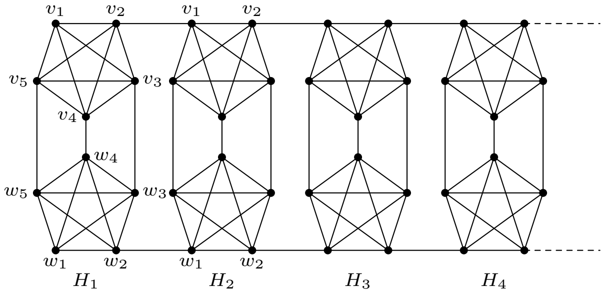

FIGURE 4. Part of a double K_5-chain showing four links.

Following the picture, we will sometimes refer to the *top* and *bottom* K_n's in a link, and refer to v_1, v_2, w_1, and w_2 as *outer* vertices (or sometimes *top-left-outer* for a v_1, for example). For a finite chain, the *ends* refer to the vertices v_1 and w_1 of the "first" link and v_2 and w_2 of the "last" link. To keep track of which link we are discussing, we will enumerate links as H_i and then refer to their vertices as $\{v_1^i, \dots, v_n^i, w_1^i, \dots, w_n^i\}$.

Lemma 3.2. *Suppose a graph G with total domatic number $n \geq 3$ contains a double K_n-chain (either infinite or with ends adjacent to exactly one other vertex outside of the chain). Assume p is a total domatic n-partition of G for which $p(v_1) \neq p(w_1)$ in any link in the chain. Then for* every *link in the chain, $p(v_1) \neq p(w_1)$ and $p(v_2) \neq p(w_2)$. Further, for consecutive links H_i and H_j in the chain, we have $p(v_1^i) = p(w_2^i) = p(w_1^j) = p(v_2^j)$ and similarly $p(w_1^i) = p(v_2^i) = p(v_1^j) = p(w_2^j)$.*

Proof. Let p be a total domatic n-partition of G. Note that since each vertex in the double K_n-chain has degree n, all the neighbors of a given vertex must be colored distinctly so that the vertex is dominated by all n colors. This implies that the n vertices of each K_n must be colored distinctly, for if two vertices were colored identically, then a third vertex of the K_n would have identically colored neighbors.

Every vertex v in the double K_n-chain is adjacent to $n-1$ vertices in its complete graph, plus one vertex w outside of its complete graph. Since v must be dominated by $p(v)$, and v is colored distinctly from all the other vertices in its complete graph, we must have $p(w) = p(v)$.

Now suppose that $p(v_1^1) \neq p(w_1^1)$ in the link H_1, which is adjacent to H_2 in the chain. Since $p(v_i^1) = p(w_i^1)$ for all i such that $3 \leq i \leq n$ (because v_i^1 is adjacent to w_i^1), we must have $p(w_2^1) = p(v_1^1)$, since this is the only color not accounted for among $\{w_1^1, \dots, w_n^1\}$. Similarly, $p(v_2^1) = p(w_1^1)$. Thus we have $p(v_2^1) \neq p(w_2^1)$. To extend this to the next link, note that v_2^1 is adjacent to v_1^2, so $p(v_2^1) = p(v_1^2)$ and similarly $p(w_2^1) = p(w_1^2)$. Now the same argument used for H_1 can be applied to H_2 to get $p(w_1^2) = p(v_2^2)$ and $p(v_1^2) = p(w_2^2)$. We can then inductively extend this to all links in the chain. □

This lemma allows us to ensure that the top-left-outer vertex of every other link in a double K_n-chain must be colored identically, as long as we know that one of the corresponding bottom-left-outer vertices is colored differently. To ensure this happens, we will build a graph T consisting of four double K_n-chains, connected to a central double K_n-link in the following way. The central link H_0 will be connected to links H_1, H_2, H_3, H_4, each the first link in their own double K_n-chain, by edges

$$(v_1^0, w_1^1),\ (v_2^0, w_1^2),\ (w_1^0, v_1^3),\ (w_2^0, v_1^4)$$

Additionally, there are edges (v_1^1, v_1^2) and (w_1^3, w_1^4). See Figure 5. To keep track of the links in each of the four chains, we enumerate the links such that H_i is followed by H_{i+4} for all $i \geq 1$. We will call the number of links in each chain the *size* of T.

Lemma 3.3. *Let T be a graph as described above, and p be a total domatic n-partition of T. Then $p(v_1^1) \neq p(w_1^1)$ or $p(v_1^2) \neq p(w_1^2)$ (or both); also $p(v_1^3) \neq p(w_1^3)$ or $p(v_1^4) \neq p(w_1^4)$ (or both).*

Proof. Assume the negation of the lemma for the sake of contradiction. Without loss of generality, suppose both $p(v_1^1) = p(w_1^1)$ and $p(v_1^2) = p(w_1^2)$. Since v_1^1 and v_1^2 are adjacent but in different complete graphs, we have $p(v_1^1) = p(v_1^2)$. Thus we

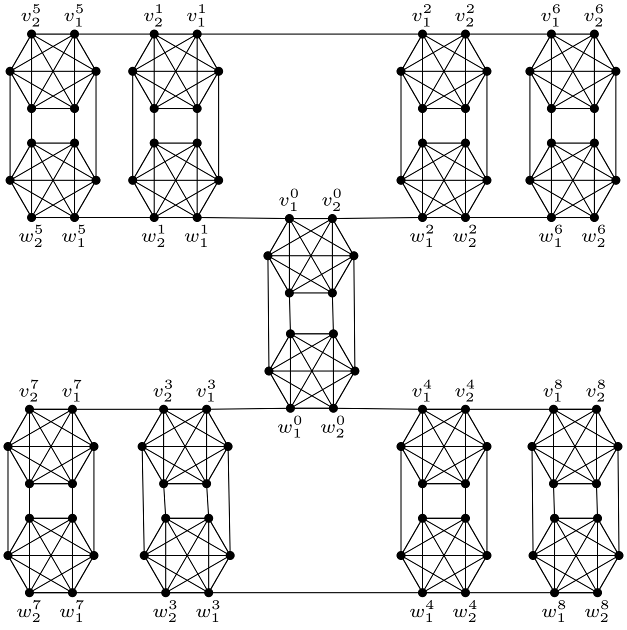

FIGURE 5. The graph T of size 2 for $n = 6$.

have $p(w_1^1) = p(w_1^2)$ as well. But w_1^1 is adjacent to v_1^0, and these are in different copies of K_n, so $p(w_1^1) = p(v_1^0)$, and similarly $p(w_1^2) = p(v_2^0)$. This implies that $p(v_1^0) = p(v_2^0)$, a contradiction, since these are two vertices in the same copy of a K_n in a double K_n-link. □

We are now ready to prove Theorem 3.2.

Proof of Theorem 3.2. The proof follows the same format as that of Theorem 3.1. As before, we build a graph for each φ_e, except this time the graph T_e will be a copy of the graph T described above consisting of a central double K_n-link connected to four double K_n-chains. As we continue to extend each of the four chains, adding a new link to each chain at each stage, we wait for φ_e to converge on all $10n$ vertices of $H_0, \dots, H_4$ in T_e. If φ_e never halts on these vertices, or does halt but not in a way that could be extended to a total domatic n-partition of all of T_e, we will simply continue to extend the four chains forever. On the other hand, if φ_e does halt on these vertices at some stage and appears to be a total domatic n-partition, we act as follows.

By Lemma 3.3, there will be $i \in \{1, 2\}$ and $j \in \{3, 4\}$ such that $\varphi_e(v_1^i) \neq \varphi_e(w_1^i)$ and $\varphi_e(v_1^j) \neq \varphi_e(w_1^j)$, otherwise φ_e could not be extended to a total domatic n-partition. We will connect the two chains starting with H_i and H_j into one chain by adding in either one or two links, depending on how φ_e colors the vertices $v_1^i, v_1^j, w_1^i, w_1^j$. Note that since this is happening at stage s, the last links in these chains are called H_{4s+i} and H_{4s+j}. There are two cases.

Case 1: If $\varphi_e(v_1^i) = \varphi_e(v_1^j)$, then add a link $H' = \{v_1', \dots, w_n'\}$ with edges

$$(v_2^{4s+i}, v_1'),\ (v_2^{4s+j}, v_2'),\ (w_2^{4s+i}, w_1'),\ (w_2^{4s+j}, w_2').$$

Case 2: Otherwise, add two links H' and H'' with edges

$$(v_2^{4s+i}, v_1'),\ (v_2^{4s+j}, v_1''),\ (w_2^{4s+i}, w_1'),\ (w_2^{4s+j}, w_1''),\ (v_2', v_2''),\ (w_2', w_2'').$$

From this point on, continue to extend the other two chains of T_e forever.

We claim that in either case, φ_e can no longer be extended to a total domatic n-partition. Suppose it could be extended to all of T_e. If we are in Case 1 above, then by Lemma 3.2 we would then have either

$$\varphi_e(v_1^i) = \varphi_e(v_1^{4s+i}) = \varphi_e(v_1^{4s+j}) = \varphi_e(v_1^j)$$

if s is even, or

$$\varphi_e(v_1^i) = \varphi_e(w_1^{4s+i}) = \varphi_e(w_1^{4s+j}) = \varphi_e(v_1^j)$$

if s is odd. But it is then impossible to color H' properly, since we would need v_1' and v_2' or w_1' and w_2' to be colored identically, which cannot happen in a total domatic n-partition. In Case 2, again by Lemma 3.2, we will either have $\varphi_e(v_1^i) = \varphi_e(v_2')$ and $\varphi_e(v_1^j) = \varphi_e(v_2'')$ or else $\varphi_e(v_1^i) = \varphi_e(w_2')$ and $\varphi_e(v_1^j) = \varphi_e(w_2'')$ (depending on the parity of s). But both of these are impossible because v_2' and v_2'' must be colored identically since they are adjacent but in different complete graphs, and similarly for w_2' and w_2''. Therefore φ_e cannot be a total domatic n-partition, and as such, the computable total domatic number is not n.

However, the graph does have a computable total domatic $(n-1)$-partition: simply color all the outer vertices of all chains with color 1, and all inner vertices with the remaining $n-2$ colors. This is clearly computable and it is easy to check that every vertex is totally dominated.

Finally, we must argue that the graph does have a (non-computable) domatic n-partition. For any T_e that gets extended forever as four chains, this is clear. For a T_e which has two chains connected, say without loss of generality the chains starting with H_2 and H_4, we can color v_1^2, w_1^2, v_1^4, and w_1^4 identically (the non-connected chains will have initial vertices satisfying Lemma 3.3). It is easy to check that this can be extended to a total domatic n-partition of all of T_e. Since the collection $\{T_e\}$ is pairwise disjoint, the separate total domatic n-partitions of each T_e can be combined into a total domatic n-partition of G. □

Here we note that, while we proved Theorem 3.2 for $n \geq 3$, the theorem is indeed still true for $n = 2$. In fact, the proof of the theorem when $n = 2$ is much simpler. The idea of the proof is that T_e will start out by consisting of a pair of two-way paths. Since $n = 2$, there are only two colors, and therefore there is only one way for φ_e to color each path in order for the coloring to be a total domatic 2-partition: the coloring would have to be of the form $\ldots, 1, 1, 2, 2, 1, 1, 2, 2, \ldots$ (where the colors are 1 and 2, without loss).

In the construction of T_e, we begin by growing both ends of each two-way path at each stage. As soon as φ_e decides on a particular coloring of each of the two paths in T_e, we simply add new vertices and edges to connect the two paths (to form a single two-way path) in such a way that it is impossible for φ_e to be a total domatic 2-partition of T_e. For instance, if φ_e colors one path $1, 1, 2, 2$ and the other path $1, 1, 2, 2$, then we could add a single new vertex to T_e and connect it to the end vertices that have color 2. Then at subsequent stages we continue to grow the other ends of T_e so that T_e will become a single infinite two-way path. Of course, because of this fact, there is certainly a total domatic 2-partition of T_e (it just is not φ_e). Also, there is certainly a computable total domatic 1-partition—color everything with color 1. If φ_e never yields a total domatic 2-partition, then T_e will be a pair of two-way infinite paths, and therefore similarly has a total domatic 2-partition, as well as a computable total domatic 1-partition.

4. Smaller domatic partitions

A graph with domatic number n necessarily has domatic partitions of all sizes less than n as well: we can identify two or more of the n colors resulting in fewer than n disjoint dominating sets. The smaller domatic partitions created in this fashion will be *divisible* in that a domatic n-partition can be formed by dividing single dominating sets into smaller disjoint dominating sets. Of course not all smaller domatic partitions need be divisible; it could be that the only way to get an additional dominating set would be for two or more of the dominating sets to contribute vertices to form the new dominating set.

The existence of divisible domatic partitions suggests a potential strategy for building larger domatic partitions. For instance, to construct a domatic 5-partition in a graph, we might first build a divisible domatic 4-partition, and then divide a divisible dominating set. Maybe we start by building a divisible domatic 2-partition, in which one set could further be divided into two disjoint dominating sets, the other into three. Or perhaps we start by finding a minimal dominating set, one whose complement can be divided into 4 dominating sets.

We will show that all these strategies are fruitless. Not only are there regular graphs which possess no computable divisible domatic partitions of a particular type, even if such a divisible domatic partition were computable, there is no guarantee that we would be able to carry out the division effectively.

Definition 4.1. A *divisible domatic k-partition* is a domatic k-partition in which at least one of the sets in the partition is the disjoint union of two or more dominating sets.

Definition 4.2. Given a graph with domatic number n, a domatic k-partition is called *fully divisible* if each set in the k-partition is the union of one or more sets from some fixed domatic n-partition.

Theorem 4.1. *For each $n \geq 3$, there is a computable $(n-1)$-regular graph G with $d(G) = n$ containing no computable fully divisible domatic 2-partition.*

Proof. The proof is similar to that of Theorem 3.1. The main difference here is what we take for the subgraphs T_e. Each T_e starts as three copies of the star graph S_{n-1}. During the construction, we build K_n^--chains starting with each leaf of each star. We wait for φ_e to color its three copies of S_{n-1}, at which point we select two chains to connect if φ_e looks like it might be a fully divisible domatic 2-partition. All other chains are extended forever.

Notice that in a K_n^--chain, any fully divisible domatic k-partition must color all left-outer vertices identically, as well as all right-outer vertices identically, since any full division requires that these be so colored. It could be that all outer vertices (both left and right) could be the same color though. In terms of the stars S_{n-1}, a fully divisible domatic k-partition with $k \geq 2$ must color the center vertex differently than at least one of the leaves, otherwise no full division would dominate the center vertex. The center of each star must be colored as if it were a right-outer vertex of each chain it is attached to.

When φ_e converges on its three stars, we only act if φ_e looks like it could be a fully divisible domatic 2-partition. Among the three stars, there will necessarily be two central vertices colored identically. At least one of the leaves from each of these two stars must be assigned a different color, and the first such pair of chains gets connected. This results in a chain with left-outer vertices not colored identically, as in Figure 6.

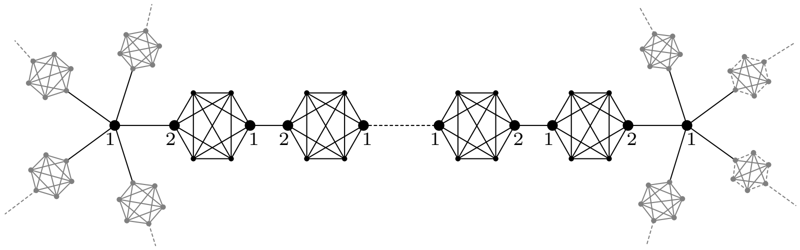

Figure 6. For two stars with identically colored central vertex (in this case 1) connect chains with initial vertex *not* colored 1 (in this case, 2).

Finally, we argue that the graph so built does indeed have domatic number n. Each T_e is either the disjoint union of three stars each with infinite K_n^--chains attached to each of their leaves, or the same only with two of the stars attached by a finite K_n^--chain. In the first case, it is easy to produce a domatic n-partition: color the center of each star with color 1 and all the leaves with color 2, which are all left-outer vertices of the first link in their chains. The inner vertices of all links get colors 3 through n, the right-outer vertices get color 1, and the left-outer vertices color 2. For the second case, we can use this same coloring for two of the stars, but for one of the stars connected by the K_n^--chain, we simply swap all instances of color 1 and color 2. Note that while we can do this, φ_e cannot, because it does not know which pair of stars will be connected when it colors the vertices of the stars. □

Corollary 4.1. *For each $n \geq 3$ and k such that $2 \leq k < n$, there is a computable $(n-1)$-regular graph G with $d(G) = n$ containing no computable fully divisible domatic k-partition.*

Proof. Consider a graph G as guaranteed by Theorem 4.1. For any k such that $2 \leq k < n$, G does not have a computable fully divisible domatic k-partition either. For if it did, we could identify colors 1 through $k-1$ to get a fully divisible domatic 2-partition. Identifying colors can be done computably (in the algorithm for the k-partition, just replace the colors 1 through $k-1$ with color 1), so this would result in a *computable* fully divisible domatic 2 partition. □

Even if we could find a fully divisible domatic partition, we might still not be able to use this to build up to a full domatic partition of the graph.

Theorem 4.2. *For any $n \geq 3$ and $k < n$, there is a computable $(n-1)$-regular graph G with $d(G) = n$ which has a computable fully divisible domatic k-partition but no computable domatic n-partition.*

Proof. The graph constructed in Theorem 3.1 works, as the computable domatic $(n-1)$-partition constructed there is in fact fully divisible. This shows that the theorem holds for $k = n - 1$. For smaller k, we get a computable fully divisible domatic k-partition by identifying colors. □

Although there is no hope of finding these "nice" divisible domatic partitions of smaller size, there still might be computable domatic partitions of sub-optimal size. Certainly any computable graph with $d(G) = n$, for $n \geq 2$, has a computable domatic 2-partition. For computable regular graphs, we can do better.

Theorem 4.3. *Suppose G is a computable k-regular graph with domatic number $k+1$. Then G has a computable domatic n-partition for all n satisfying $n^2 \leq k+1$.*

Proof. We show the theorem for $k \geq 8$. (The theorem clearly holds for $k < 8$, because every computable graph without isolated vertices already has a computable domatic 2-partition, as the current authors showed in [4].) We also fix n to be maximal (such that $n^2 \leq k+1$), because the theorem will then clearly follow for smaller values of n. Also, assume G is connected. At the end of the proof, we will describe how to modify the procedure in case we do not know that G is connected.

As a point of terminology, we say that a set S of vertices is *dominated* by a set of colors $\{c_1, \ldots, c_l\}$ if all of the vertices of S are colored with $\{c_1, \ldots, c_l\}$ and each vertex of S that has all of its neighbors in S is dominated under this coloring. Note that any finite set S of vertices of G can be dominated by $\{1, \ldots, k+1\}$. Indeed, since G has a domatic $(k+1)$-partition by hypothesis, we will eventually find one that works on S by searching through all possible partitions of S (because each such partition is itself finite). We will use this fact throughout the procedure.

Procedure: Define the *disk centered at vertex v of radius r* to be the set $D_v(r) = \{u \in V \mid d(u,v) \leq r\}$, where $d(u,v)$ is the length of the shortest path from u to v. For brevity, we write $D_{v_0}(r)$ as $D(r)$. Let $A_0 = D(2)$ and $B_0 = D(4) \setminus A_0$. Color $A_0 \cup B_0$ so that it is dominated by colors $\{1, \ldots, k+1\}$, ensuring that vertex v_0 is colored with color 1. We declare v_0 to be permanently colored (with color 1), while the other vertices have only been temporarily colored. Assume inductively that at the previous stage s of the procedure, $A_s = D(r_s)$ for some radius r_s and $B_s = D(r_s+2) \setminus A_s$. Also assume inductively that the vertices of $(A_s \setminus D(r_s-2)) \cup B_s$ were temporarily colored with colors $\{1, \ldots, k+1\}$ at stage s. We color vertices at stage $s+1$ as follows (some vertices will be permanently colored, while others will only be temporarily colored).

Define $A^0_{s+1} = A_s$, where the superscript 0 indicates an initial definition of A_{s+1}, as we will be altering the set at the end of this stage. Define B^0_{s+1} to be $D(r_s+6) \setminus A_s$. Color the set $D(r_s+6) \setminus D(r_s-2)$ so that it is dominated by colors $\{1, \ldots, k+1\}$. Note, we have *doubly colored* some of the vertices (i.e., the 2 layers of A^0_{s+1} closest to B^0_{s+1} and the 2 layers of B^0_{s+1} closest to A^0_{s+1}, constituting a 4-wide band of doubly-colored vertices): they have their original coloring (the temporary coloring from the previous stage), which we will say is *preferred* by A^0_{s+1}, and the coloring just given, *preferred* by B^0_{s+1}. Denote the double-color of each such vertex by $x|y$, where x is the color preferred by A^0_{s+1} and y is the color preferred by B^0_{s+1}.

Resolve the color of each doubly-colored vertex v by choosing one color for v as follows. First, to change from $k+1$ down to n colors, we will convert to congruent numbers modulo n. Specifically, define $f : \{1, \ldots, k+1\} \to \{1, \ldots, n\}$ by $f(c) = (c - 1 \mod n) + 1$. (This way, colors 1 through n stay the same, color $n+1$ goes to 1, $n+2$ to 2, $n+3$ to 3, and so on.) Use the following rules to resolve v, which is currently colored $x|y$.

- If $x, y \leq n$, color it $f(x)$ if $v \in A^0_{s+1}$ and $f(y)$ if $v \in B^0_{s+1}$.
- If $x, y > n$, color it $f(x)$.
- Otherwise, color it $f(x)$ if $x \leq n$ and $f(y)$ if $y \leq n$.

After resolving a doubly-colored vertex, declare it permanently colored. Next, let $r_{s+1} = r_s + 4$. Enlarge A^0_{s+1}, defining A_{s+1} now to be $D(r_{s+1})$, and restrict B^0_{s+1} accordingly, defining $B_{s+1} = D(r_{s+1}+2) \setminus A_{s+1}$. Note that A_{s+1} includes all the vertices that are permanently colored, as well as 2 layers previously just (temporarily) colored by B^0_{s+1}. For the next stage, informally speaking, we consider all the coloring done by B to be the one preferred by A, as B will be coloring a new set of vertices, including all the non-permanently (i.e., temporarily) colored vertices in A.

Verification: Let $v \in V$, and fix s such that $s+1$ is the earliest stage of the procedure after which v and all of its neighbors have been permanently colored. We show that v is dominated by the end of stage $s+1$ and therefore will remain dominated. Consider the two possible cases that could occur just before the resolution of the doubly-colored vertices: (1) some vertices among v and its neighbors are *singly colored* (i.e., colored, but not doubly colored), and (2) none of these vertices are singly colored. In each case, fix $c \in \{1, \ldots, n\}$. We show that v or one of its neighbors gets permanently colored c.

Case 1: At least one vertex (among v and its neighbors) is singly colored. Note that because of the width of the doubly-colored region, v and its neighbors are either all in A^0_{s+1} or all in B^0_{s+1}. If they are all in B^0_{s+1}, then one of them would be colored c by B^0_{s+1} and would thus get resolved to c by rule 1. Also note that since we have assumed that G is connected, it cannot be the case that they are all in B^0_{s+1}. However, this can happen in the general case, as we will see when describing that procedure below, so it was important for us to consider that possibility.

Now assume they are all in A^0_{s+1}, and assume that no singly-colored vertex (among v and its neighbors) is colored c. Again because of the width of the doubly-colored region in each stage, all of the singly-colored vertices must have been doubly colored (and thus received their permanent color) during the previous stage (i.e., stage s) and were in the set B^0_s. So, since none of them resolved to the color c, the preference of B^0_s for each of those vertices during stage s could not have been c. Hence B^0_s must have preferred a different vertex w (among v and its neighbors) to be colored c (in order to dominate v, since v was also in B^0_s at the time), namely one of the now doubly-colored vertices (in stage $s+1$). Since A^0_{s+1} and B^0_s color vertices in the same way and $w \in A^0_{s+1}$, w will resolve to c.

Case 2: All vertices (among v and its neighbors) are doubly colored. Then for every j, with $0 \leq j < n$, there is a vertex among v and its neighbors that is colored $(c+jn)|y_j$ for n distinct colors y_j. Now if any $y_j > n$, then $(c+jn)|y$ resolves to $f(c+jn) = c$ by the second or third rule for resolving doubly-colored vertices. On the other hand, if $y_j \leq n$ for all $0 \leq j < n$, then one of these n vertices (with color of the form $(c+jn)|y_j$) has $y_j = c$ and hence gets resolved to c by the first or third rule. Thus at least one of the vertices resolves to c as desired.

Not Necessarily Connected Graphs: If we are not promised that G is connected, we perform the same procedure as in the connected case, only now we enumerate G and decide if the current vertex v of G should be part of the existing sets A and

B, or if it should be put in its own pair of sets A and B, disjoint from the original pair. This decision will depend on the distance of v from A and B.

As the procedure progresses, there will be disjoint collections of vertices with their own sets A and B. We will call each such collection of vertices a *system*. These disjoint systems will get resolved (via the double-coloring routine) simultaneously. As they are increasing in size, however, they might eventually collide, at which point the part of the procedure that enlarges B will now enlarge it to a size that fully encompasses all colliding systems. This will result in all systems that were involved in the collision to combine into one common system. Note that some vertices in this enlarged B might not be a part of the double-coloring and will thus still have colors outside of $\{1, \ldots, n\}$. So we will change the colors of the vertices of B, except those in its outer 4 layers, according to the following rule (which will narrow down from $k + 1$ to n colors and will give these vertices their permanent color).

If the vertex has color x, change it to $f(x)$.

Eventually the procedure will have been successfully performed on each component of G. □

We observe the following version of Theorem 4.3, which uses a more efficient procedure for small values of k. However, it quickly becomes less efficient, requiring an exponential gap between the domatic numbers. Indeed, our original procedure only requires a polynomial (in fact, quadratic) gap.

Theorem 4.4. *Suppose G is a computable k-regular graph with domatic number $k + 1$. Then G has a computable domatic n-partition for all n satisfying $2^n - 1 \leq k + 1$.*

Although we do not present the complete proof of the above observation, we show as an example the case where $k = 6$, which we call the 7-3 problem (because the graph will have domatic number $k+1 = 7$ and computable domatic number 3). Notice that for the procedure given in Theorem 4.3, the corresponding problem is the 9-3 problem, which gives a larger gap between domatic and computable domatic number. However, Theorem 4.3 quickly becomes more efficient, providing smaller gaps for $k \geq 24$.

Proof of Theorem 4.4 when $k = 6$. Consider a computable graph that is 6-regular and has domatic number 7. Also assume that the graph is connected. As before, the procedure can be modified to deal with graphs that are not necessarily connected. We perform a similar procedure as in Theorem 4.3, but we use a different set of rules to resolve the doubly-colored vertices. Also, due to a counting argument needed in the verification, we make a slight change to the placement of the doubly-colored region of vertices in each stage. Note that in the original procedure, the doubly-colored regions in consecutive stages were adjacent to each other. This time around, we must keep them apart by a distance of two vertices.

Procedure: Let $A_0 = D(4)$ and $B_0 = D(6) \setminus A_0$. Now perform the coloring just as we did in the original procedure with one additional condition. Whenever coloring a set S of vertices, ensure that any vertex v outside of S but adjacent to at least two vertices in S is not adjacent to two vertices of the same color, according to the coloring just performed. This can be ensured by coloring the union of S and all such v, then suppressing the coloring of those v. Also as before, at the beginning of stage $s+1$, let $A^0_{s+1} = A_s$, but now let $B^0_{s+1} = D(r_s+8) \setminus A_s$. Color the set $D(r_s+8) \setminus D(r_s-2)$ similarly to the original procedure (noting that this creates two layers in A^0_{s+1} of singly-colored vertices between the doubly-colored regions of stage s and stage $s+1$).

Now use the new set of rules, shown below, for resolving the doubly-colored vertices and the aforementioned singly-colored vertices (i.e., the two layers between the doubly-colored regions of the current and previous stage). These rules will give the corresponding vertices their permanent color. As a point of terminology, we say that a color c is *present* if the vertex in question is either singly colored and has color c or is doubly colored and has color of the form $c|x$ or $x|c$ for some x.

- If 1 is present, color it 1.
- If 2 or 3 is present (with 1 not), color it 2.
- Otherwise, color it 3.

Finally, let $r_{s+1} = r_s + 6$, $A_{s+1} = D(r_{s+1})$, and $B_{s+1} = D(r_{s+1}+2) \setminus A_{s+1}$.

Verification: We show that each $v \in V$ is eventually dominated. For any $v \in V$, wait for the stage $s+1$ after which v and all of its neighbors have been permanently colored. First, we claim that there exist two sets (either the pair A^0_s, B^0_s or the pair A^0_{s+1}, B^0_{s+1}) such that each vertex among v and its neighbors received its permanent color at the time it was being either singly colored by one of the two sets or doubly colored by one of the two sets. Indeed, if none of the vertices is doubly colored during the current stage, then A^0_s, B^0_s is such a pair of sets. On the other hand, if at least one of the vertices is doubly colored during the current stage, then we can choose A^0_{s+1}, B^0_{s+1} to be the pair of sets.

We now fix two sets A and B from the claim above, and we notice two more facts. First, neither A nor B colors any two vertices (among v and its neighbors) the same, because we ensured that whenever a set of vertices was being dominated, no vertex adjacent to at least two vertices in the set could have two neighbors colored the same. Second, note that either A colors all seven vertices or B colors all seven vertices (or both sets color all of them).

Therefore, for the sake of this verification, we can consider all seven vertices to be "doubly colored." Specifically, each vertex has color of the form $x|y$, where A prefers x and B prefers y, and we use a blank color when a set has not actually colored the vertex. Also, because of our second observation, we know that either A colors none of the vertices blank or B colors none of them blank. (Notice that we are abusing the notation for doubly-colored vertices here in the sense that some of the vertices may have been given their permanent color in the previous stage, while others are being given it in the current stage. This is the case if $A = A^0_s$ and $B = B^0_s$. The original notation was being used only for vertices being doubly colored in the current stage.)

We can now show that for each color $c \in \{1,2,3\}$, at least one of the vertices among v and its neighbors gets permanently colored c. For brevity, we will say "there is a c," when making this claim. First, it is clear that there is a 1, because by the discussion above there is a vertex for which 1 is present (i.e., colored $1|x$ or $x|1$ for some color x). So it gets permanently colored 1 by rule 1. Next, to show that there is a 2, note that there are at most two vertices (among v and its neighbors) for which 1 is present. So there is at least one vertex for which 2 or 3 is present without 1 being present. Thus, by rule 2, at least one vertex gets resolved to 2, so there is a 2. Finally, to show there is a 3, note that on top of the at most two vertices for which 1 is present, there are at most four more vertices for which 2 or 3 is present. So there is still at least one remaining vertex (of the seven) having no $1, 2$, or 3 present at all. Thus at lest one vertex gets resolved to 3, so there is a 3. □

Finally, to give one more detail of the proof of Theorem 4.4, we present the sets of rules for larger values of k. The rules shown above work for $6 \leq k \leq 13$. (These correspond to the 7-3 problem, the 8-3 problem, and so on, up to the 14-3 problem). We now give the rules for $14 \leq k \leq 29$ (i.e., the $(k+1)$-4 problems), $30 \leq k \leq 61$ (i.e., the $(k+1)$-5 problems), and arbitrary k.

$14 \leq k \leq 29$:

- If 1 is present, color it 1. As shorthand, we write $1 \to 1$.
- If 2 or 3 is present (with 1 not), color it 2 (i.e., $2, 3 \to 2$ for short).
- If $4, 5, 6$, or 7 is present (with $1, 2$, and 3 not), color it 3 (i.e., $4, 5, 6, 7 \to 3$).
- Otherwise, color it 4.

$30 \leq k \leq 61$:

- $1 \to 1$
- $2, 3 \to 2$
- $4, 5, 6, 7 \to 3$
- $8, 9, 10, 11, 12, 13, 14, 15 \to 4$
- Otherwise, color it 5.

The pattern is that each rule uses one more color than the sum of the number of colors used in all of the previous rules. This is what forces us to have an exponential number of colors. Indeed, for arbitrary k, letting n be the computable domatic number of the graph, we have:

- $1 \to 1$
- $2, 3 \to 2$
- $4, 5, 6, 7 \to 3$
- $8, 9, 10, 11, 12, 13, 14, 15 \to 4$

$\vdots$

- $2^{n-1}, \ldots, 2^n - 1 \to n - 1$
- Otherwise, color it n.

5. Conclusion and Open Questions

As it stands right now, we know that there are computable regular graphs with domatic number n but computable domatic number $n-1$, and that every computable regular graph with domatic number n has a computable domatic partition of size at least $\lfloor \sqrt{n} \rfloor$. It would be nice to shrink this gap. We suspect that Theorem 3.1 could be improved to build a computable regular graph with domatic number n but no computable domatic $n-1$ partition. However, notice that the method we used to diagonalize against all computable functions relied on our ability to force all domatic n-partitions to color specific vertices identically. We can be sure that this particular approach will not help in separating $d(G)$ and $d^c(G)$ further.

Proposition 5.1. *If a graph G has domatic number $n \geq 3$, then for any vertices u and v of G, there is a domatic $(n-1)$-partition in which u and v are colored identically, and a domatic $(n-1)$-partition in which u and v are colored differently.*

Proof. Consider the domatic n-partitions of G. If there exists partitions in which u and v are colored differently, then we can get a domatic $(n-1)$-partition in which u and v are colored differently by eliminating another color, and a domatic $(n-1)$-partition in which u and v are colored identically by changing the color of u (for all similarly colored vertices) to the color of v. On the other hand, if all domatic n-partitions put u and v into the same color, the we can get a domatic $(n-1)$-partition in which u and v are identically colored by eliminating a different color. To get a domatic $(n-1)$-partition in which u and v are differently colored, take a color other than that of u and v and merge it with the color of u and v. Then v and all of its neighbors will be dominated by v's color at least *twice*: once by v and once by the vertices whose color we just changed. Thus we can recolor v without losing domination. □

Similar arguments can show that there is in fact very little we can force all domatic $(n-1)$-partitions to do when the graph has domatic number n. There are two obvious exceptions to this general principle. First, by the pigeonhole principle, given n vertices, we know that every domatic $(n-1)$-partition will put at least two of these into the same class. Second, since it is possible to force n vertices to belong to n different classes in a domatic n-partition, we can force that among n vertices, every domatic $(n-1)$-partitions will put exactly two of them into the same class. To summarize:

Proposition 5.2. *There is a graph G with domatic number n containing a set of n vertices $\{v_0, v_1, \ldots, v_{n-1}\}$ such that every domatic $(n-1)$-partition of G will put exactly two of these n vertices into the same class.*

Whether this observation can be leveraged to further separate $d(G)$ and $d^c(G)$ is not clear. Thus we as:

Question 1. Is there a computable regular graph G with $d(G) = n$ but $d^c(G) = n-2$? In general, for a fixed n, what is the largest $d(G) - d^c(G)$ can be?

Also not clear is what role the minimal degree of our graphs play. The graph in Theorem 3.1 is domatically full in that the domatic number is as large as possible based on the minimal degree of the graph, so we have that the minimal degree of G is $n-1$, equal to the computable domatic number. Can we force the computable domatic number to be *less* than the minimal degree? The answer is yes if the graphs need not be regular - we can build a graph in which the minimal degree is arbitrarily larger than the domatic number, using a generalization of Zelinka's argument in [8] and independently build a graph with large domatic number but computable domatic number 2. If we can further separate $d(G)$ and $d^c(G)$ for regular computable graphs, then a similar argument can separate the minimal degree and the computable domatic number. But perhaps answering the following question would be easier.

Question 2. Is there a computable n-regular graph G with $d(G) = n$ and $d^c(G) = n-1$.

Along similar lines, we would like to know what happens when we relax our insistence that our graphs be regular, but still require that there was a procedure to find all neighbors of a given vertex. In particular, regularity is required for the proof of Theorem 4.3, but perhaps a different approach would allow us to find a small non-trivial domatic partition.

Question 3. Suppose G is a highly computable graph (i.e., there is an algorithm which produces all the neighbors of any given vertex) with domatic number $n \geq 4$. For which such n (if any) must there be a computable domatic k-partition with $k \geq 3$?

Finally, another direction to take this line of inquiry is to consider other kinds of domatic partitions. We have already seen that considering total domatic partitions gives at least some separation between classical and computable domatic number, but there are other notions of domatic partition which might be interesting to consider. For example, the *independent domatic number*, denoted $d_i(G)$ is defined in [3] to be the size of the largest partition of vertices into sets which are both domatic and independent. Another: the *paired domatic number*, denoted $d_{pr}(G)$ is the size of the largest partition into dominating sets, for which the induced subgraph of each contains a perfect matching (this notion is also defined in [3]). We can, of course, consider the computable analogues of these invariants, $d_i^c(G)$ and $d_{pr}^c(G)$, and ask how they can differ from the classical invariants.

Quite by accident, the graph constructed in Theorem 3.1 has $d_i(G) = n$ but $d_i^c(G) = n - 1$: the domatic n partition of G happens to be an independent domatic partition, and the computable domatic $(n-1)$-partition is a computable independent domatic $(n-1)$-partition. It also just so happens that the total domatic n-partition and computable total domatic $(n-1)$-partition in the graph built for Theorem 3.2 are also paired domatic partitions, so we have an example of a computable graph with $d_{pr}(G) = n$ but $d_{pr}^c(G) = n - 1$. We wonder if this is a coincidence.

Classically we have $d(G) \geq d_i(G)$ and $d(G) \geq d_t(G) \geq d_{pr}(G)$. It would be interesting to see what happens to these inequalities when we consider the computable analogues. For example, is there a computable graph G for which $d_t(G) = d_{pr}(G)$ but $d_t^c(G) > d_{pr}^c(G)$? Could we have $d_t(G) > d_{pr}(G)$ but $d_t^c(G) = d_{pr}^c(G)$? How extreme could the inequalities be? We summarize these questions as follows.

Question 4. Let G be a computable graph. Let d_x and d_y be two comparable notions of domatic number. Then what, if anything, does $d_x(G) - d_y(G)$ imply about $d_x^c(G) - d_y^c(G)$?

References

[1] Dwight R. Bean, *Effective coloration*, J. Symbolic Logic **41** (1976), no. 2, 469–480. MR0416889 (54 #4952)

[2] Michael R. Garey and David S. Johnson, *Computers and intractability*, W. H. Freeman and Co., San Francisco, Calif., 1979. A guide to the theory of NP-completeness, A Series of Books in the Mathematical Sciences. MR519066 (80g:68056)

[3] Teresa W. Haynes, Stephen T. Hedetniemi, and Peter J. Slater, *Fundamentals of domination in graphs*, Monographs and Textbooks in Pure and Applied Mathematics, vol. 208, Marcel Dekker, Inc., New York, 1998. MR1605684 (2001a:05112)

[4] Matthew Jura, Oscar Levin, and Tyler Markkanen, *Domatic partitions of computable graphs*, Arch. Math. Logic **53** (2014), no. 1-2, 137–155. MR3151402

[5] Hartley Rogers Jr., *Theory of recursive functions and effective computability*, Second, MIT Press, Cambridge, MA, 1987. MR886890 (88b:03059)

[6] James H. Schmerl, *Recursive colorings of graphs*, Canad. J. Math. **32** (1980), no. 4, 821–830. MR590647 (81m:03054)

[7] Robert I. Soare, *Recursively enumerable sets and degrees*, Perspectives in Mathematical Logic, Springer-Verlag, Berlin, 1987. A study of computable functions and computably generated sets. MR882921 (88m:03003)

[8] Bohdan Zelinka, *Domatic number and degrees of vertices of a graph*, Math. Slovaca **33** (1983), no. 2, 145–147. MR699083 (85b:05146)

Department of Mathematics, Manhattan College, 4513 Manhattan College Parkway, Riverdale, NY 10471, USA
E-mail address: matthew.jura@manhattan.edu

School of Mathematical Sciences, University of Northern Colorado, Greeley, CO 80639, USA
E-mail address: oscar.levin@unco.edu

Department of Mathematics, Physics, and Computer Science, Springfield College, 263 Alden Street, Springfield, MA 01109, USA
E-mail address: tmarkkanen@springfieldcollege.edu