On the Effectiveness of Symmetry Breaking

Russell Miller[1], Reed Solomon[2], and Rebecca M. Steiner[3]

[1] Queens College and the Graduate Center of the City University of New York
Flushing NY 11367
[2] University of Connecticut
Storrs CT 06269
[3] Vanderbilt University
Nashville TN 37240
r.m.steiner@vanderbilt.edu

Abstract. Symmetry breaking involves coloring the elements of a structure so that the only automorphism which respects the coloring is the identity. We investigate how much information we would need to be able to compute a 2-coloring of a computable finite-branching tree under the predecessor function which eliminates all automorphisms except the trivial one; we also generalize to n-colorings for fixed n and for variable n.

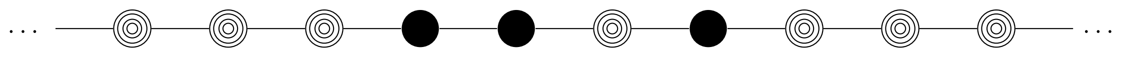
1 Introduction

Symmetry has always been a crucial concept in mathematics. We think of symmetry as a geometric property, but in fact, symmetries appear in many other branches of math as well. The symmetries of a mathematical structure are precisely its automorphisms – the bijections from the structure onto itself which preserve the essential properties of the structure. Some structures have many symmetries, and others have only one (the identity, or trivial automorphism).

Symmetry breaking involves coloring the elements of a structure in such a way that the only automorphism which respects the coloring is the trivial one; "breaking" symmetries can be thought of as "killing off" automorphisms.

Definition 11 An *n-coloring* of a structure is a function from the domain of the structure into a set of size n. It is said to *distinguish* the structure if there are no nontrivial automorphisms of the structure which respect the equivalence relation defined by the coloring. If a distinguishing n-coloring exists, then the structure is said to be *n-distinguishable.*

Definition 12 The *distinguishing number* of a structure is the smallest $n \in \omega$ such that the structure has a distinguishing n-coloring. If it exists, then the structure is *finitely distinguishable.*



As an example, consider a graph that looks like the integers, as shown here. As a graph, it has infinitely many automorphisms. But there is a way to color the elements of this graph with just two colors so that the only automorphism which respects the coloring is the trivial one. A certain three elements are given the “solid” color, as in the figure here, while the rest are given the “striped” color, and the only symmetry of this graph which respects this coloring is the identity. So we would say that this graph has distinguishing number 2.

Symmetry breaking has been studied extensively by combinatorists, with very recent results detailed in [1], [5], and [6]. In fact, the work found in the next section on symmetry breaking from a computability-theoretic perspective was inspired by a result from one of these articles:

Theorem 13 ([1], Theorem 3.1) *The countable random graph has distinguishing number* 2.

This is extremely surprising. The random graph has continuum-many automorphisms, and is ultrahomogeneous: every finite partial automorphism extends to an automorphism of the entire graph. (This says that, in a certain sense, its automorphisms are dense, within the finite partial maps respecting its edge relation.) Moreover, while the result of Theorem 13 was not at all intended to be an effectiveness result, the construction of the distinguishing coloring of the countable random graph in [1] is indeed effective in the edge relation of the graph. In other words, a computable copy of the random graph has a *computable* distinguishing 2-coloring.

Knowing that this holds for the random graph inspired us to investigate the same question for other structures: what kinds of computable structures have *computable* distinguishing n-colorings? It was this question which led to our study of effective symmetry breaking in computable finite-branching predecessor trees, which form a natural first step in the subject, mainly because the automorphisms of such structures are readily understood.

Definition 14 A *tree* is a partial order $\prec$ on a set $\mathcal{T}$ of nodes, with a least element r (the *root*) under $\prec$, such that for every $x \in T$, the set $\{y \in T \ : \ y \prec x\}$ is well-ordered by $\prec$. If every chain under $\prec$ has order type $\leq \omega$ (that is, if the tree has *height* $\leq \omega$), then each $x \in T$ has an immediate predecessor under $\prec$. A *predecessor tree* is a tree of height $\leq \omega$ in a language with equality and one unary function P, the *predecessor function*, for which $P(r) = r$ and $P(x)$ is the immediate predecessor of x whenever $x \neq r$. If $\mathcal{T}$ has domain ω and P is computable, we call $\mathcal{T}$ a *computable predecessor tree*; such $\mathcal{T}$ correspond precisely to computable subtrees of the tree $\omega^{<\omega}$ of finite strings from ω. (The underlying partial order on such a T is computable from P, although not definable by finitary formulas using P.)

A predecessor tree is *finite-branching* if, for every y, there are only finitely many $x \in T$ with $P(x) = y$.

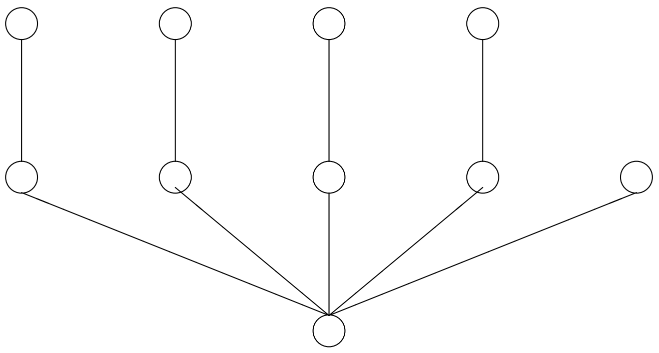
Trees which are computable as partial orders (but for which the predecessor function is not necessarily computable) are considered in a different context in [3, 4], which may provide useful background for readers interested in investigating these questions. In such a tree, it is not generally possible to compute the level of a node, and this makes it substantially more difficult to determine which nodes lie in the same orbit under automorphisms of the tree. It would be natural to attempt to extend the results of this article to computable trees under $\prec$. Some previous effectiveness results about predecessor trees appear in [7], while for more general effectiveness results about symmetries as automorphisms, we refer the reader to [2].

2 The Effectiveness Results

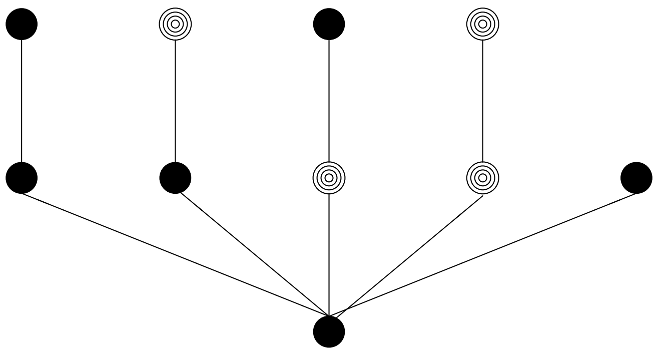
The most natural question to address first is whether a computable finite-branching predecessor tree with distinguishing number 2 must have a *computable* distinguishing 2-coloring.

Theorem 21 *There is a computable finite-branching predecessor tree which is distinguished by a 2-coloring but not by any computable 2-coloring.*

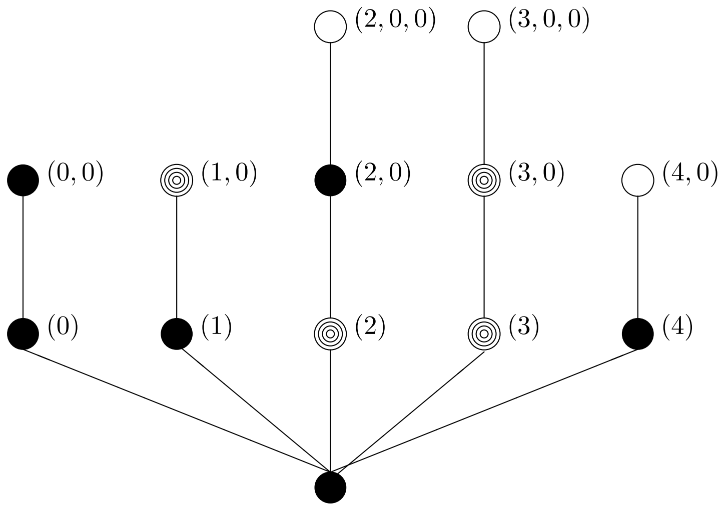
Proof. We will build our tree $\mathcal{T}$ in such a way that no (partial) computable function φ_e can be a distinguishing 2-coloring of the tree. We start by describing the basic module – the strategy which will guarantee that for some fixed e, φ_e is not a distinguishing 2-coloring of the tree. We build a finite tree $\mathcal{T}_e$ beginning with a root, five immediate successors of the root, and one immediate successor each for four of those five immediate successors of the root, as shown here.



We wait for φ_e to converge on each of these ten inputs. If φ_e doesn't converge on all the inputs or converges outside the set $\{0,1\}$, then we do nothing further, because φ_e is not a 2-coloring of $\mathcal{T}_e$. If φ_e converges on all ten nodes to values in $\{0,1\}$ in such a way that there is already a nontrivial automorphism of $\mathcal{T}_e$ which preserves the coloring, then we do nothing further, because φ_e is not a distinguishing coloring of $\mathcal{T}_e$. So, without loss of generality, suppose φ_e converges on all ten nodes to values in $\{0,1\}$ and colors them as shown.



We do not want φ_e to be a distinguishing coloring of $\mathcal{T}_e$, so we respond by adding three more nodes to $\mathcal{T}_e$, as seen below. (For convenience, nodes are now identified as in $\omega^{<\omega}$.)



We have now made it impossible for φ_e to be a distinguishing 2-coloring of $\mathcal{T}_e$. φ_e must color the new node $(4, 0)$ at level 2, and whichever color it chooses, there will be a nontrivial automorphism of $\mathcal{T}_e$ which respects that coloring. However, there does exist another 2-coloring f_e which distinguishes $\mathcal{T}_e$: change (4) to a striped node, while keeping the other colors and coloring the remaining three nodes with either color. Under this coloring the tree is rigid.

We put these basic modules together to build one big tree $\mathcal{T}$ as follows. Start with a *spine*: a single path $d_0 < d_1 < d_2 < \cdots$ of nodes, among which d_0 will be the root of $\mathcal{T}$. Above each d_{2e+1}, in addition to d_{2e+2}, we place a node r_e. Then we build a copy of the tree $\mathcal{T}_e$ with r_e as its root. Thus no φ_e is a distinguishing 2-coloring of $\mathcal{T}$, but $\mathcal{T}$ is distinguishable by combining the 2-colorings f_e for each T_e into a single f and coloring every node on the spine striped. The spine is fixed by every automorphism of T, as is each r_e, and this (noncomputable) f then ensures that no automorphism except the identity can respect f. □

In the tree constructed in Theorem 21, the branching function is not computable: it was not decidable which of the $(4, 0)$ nodes (in all the different finite subtrees $\mathcal{T}_e$) have successors and which do not. So one naturally asks whether a computable finite-branching tree with distinguishing number 2 and with computable branching function would necessarily have a computable distinguishing 2-coloring. However, the answer is still no: below, in Theorem 23, we construct such a tree with no computable distinguishing 2-coloring.

We start by describing the *balanced* 2*-coloring* of the complete binary tree $2^{<\omega}$. Two nodes in this tree are called *siblings* if they are of the form $\sigma\hat{}0$ and $\sigma\hat{}1$, that is, if they have the same immediate predecessor σ. Thus every node except the root has exactly one sibling. A 2-coloring is *balanced* if it colors each node differently from its sibling: for instance, every node $\sigma\hat{}0$ is solid and every node $\sigma\hat{}1$ is striped. This example is isomorphic to every other balanced 2-coloring of $2^{<\omega}$, except for the color of the root. We will therefore speak of the *balanced* 2*-coloring with striped root* or the *balanced* 2*-coloring with solid root.* It is clear, by induction on the lengths of nodes, that each balanced 2-coloring distinguishes $2^{<\omega}$, i.e., no automorphism of $2^{<\omega}$ except the identity respects this coloring. Likewise, we speak of balanced 2-colorings of finite binary trees 2^n.

However, it is also possible for an *unbalanced* 2-coloring to distinguish $2^{<\omega}$. As an example, color the nodes so that every node 1^n is striped, and also every node 1^n0 is striped, with all other nodes colored according to the balanced coloring with striped root. The two siblings 0 and 1 at level 1 are both striped, but the two successors of 0 are two different colors, while those of 1 are both striped. Therefore no automorphism respecting the coloring can interchange 0 with 1, and one then uses this same argument to go upwards through all levels of the tree and see that each level is fixed pointwise by every automorphism respecting the coloring.

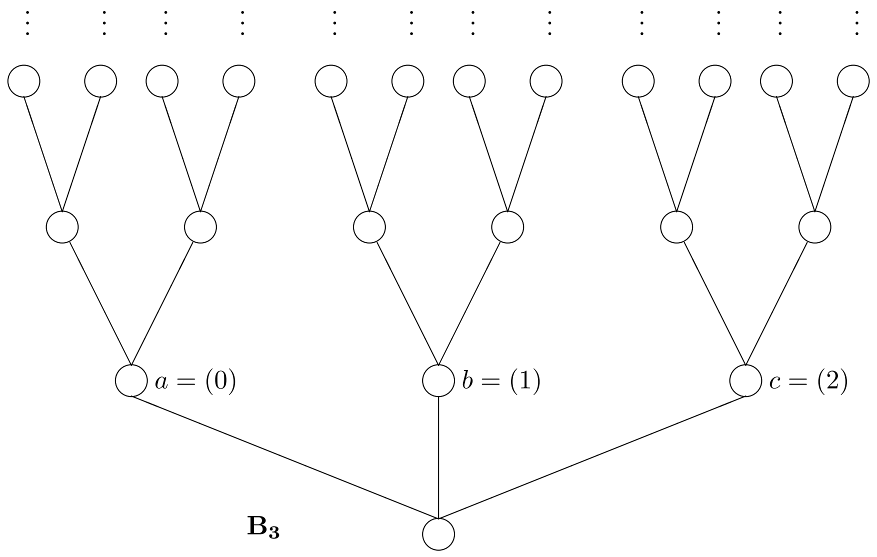
There are in fact many of these unbalanced colorings. However, once an imbalance has been introduced, it perpetuates itself.

Lemma 1. *In a distinguishing 2-coloring of $2^{<\omega}$, if some siblings $\sigma\hat{}0$ and $\sigma\hat{}1$ share a color, then either some two siblings extending $\sigma\hat{}0$ share a color as well, or else some two siblings extending $\sigma\hat{}1$ share a color.*

Proof. If not, then the coloring would restrict to the balanced coloring on the tree above $\sigma\hat{}0$, and also on the tree above $\sigma\hat{}1$, with the same colored root in both. Therefore, there would be an automorphism interchanging $\sigma\hat{}0$ with $\sigma\hat{}1$ and respecting the coloring. □

Corollary 22 *For every finite binary tree $2^{<n}$, no unbalanced 2-coloring is distinguishing.*

Proof. The reasoning is the same as in the lemma: any imbalance forces there to be another imbalance above itself. However, now this yields a pair of siblings with the same color at the very top level of the tree, and the automorphism which interchanges this pair and fixes all other nodes respects the unbalanced coloring. □



With these unbalanced colorings, we see that the tree B_3 in the figure above is 2-distinguishable:

$$B_3 = \left\{\sigma \in \omega^{<\omega} : \sigma(0) < 3 \;\&\; \sigma(n) < 2 \text{ for } 0 < |\sigma| \leq n\right\}.$$

Indeed, B_3 has a computable distinguishing 2-coloring: just give the balanced coloring with solid root on the binary tree above a, the balanced coloring with striped root on the binary tree above b, and any unbalanced coloring (say with striped root) on the binary tree above c. We also have a distinguishing 2-coloring of each tree $B_{3,n}$:

$$B_{3,n} = B_3 - \{\sigma \in B_3 : \sigma(0) \neq 0 \ \& \ |\sigma| \geq n\}.$$

This $B_{3,n}$ is the tree gotten by chopping off (at level n) two of the three binary trees in B_3. To get a distinguishing 2-coloring of $B_{3,n}$, however, one is forced by the corollary above to use balanced colorings, with roots of different colors, on each of the finite trees above b and c. The binary tree above a is still complete, however: one can color it using the balanced 2-coloring with either color for the root, or using any unbalanced distinguishing 2-coloring.

Theorem 23 *There is a computable finite-branching predecessor tree* with computable branching function *which is distinguished by a 2-coloring but not by any computable 2-coloring.*

Proof. With the above observations, we can produce a tree $\mathcal{T}_e$ with distinguishing number 2 for which a given partial computable function φ_e is not a distinguishing 2-coloring. Moreover, $\mathcal{T}_e$ will have computable branching (uniformly in e). This will form the basic module of the construction below. To build $\mathcal{T}_e$, start with three distinct immediate successors a, b, c of the root node r, and begin building a copy of $2^{<\omega}$ above each, exactly as in the tree B_3. When we add level s to these binary trees, we check to see whether $\varphi_{e,s}$ has converged yet on all three nodes at level 1. If it never does so, or if it gives values $\notin \{0,1\}$ for any of them, then we simply keep building a copy of B_3. However, if it does output values in $\{0,1\}$ for all three, then we change our strategy. Without loss of generality, say that $\varphi_e(b) = \varphi_e(c)$. Once we see this, we end the construction of the binary trees above b and c: they are complete up to level s, but contain no nodes at all above level s. (Above a, we continue to build the complete binary tree, although in fact putting a single node at level $n+1$ above a would suffice.)

The point is that, having committed to the same color for both b and c, φ_e is now trapped into giving a non-distinguishing 2-coloring of $\mathcal{T}_e$. By the Corollary, the only way to give a distinguishing 2-coloring above b is to give the balanced 2-coloring, up to level n; and the same above c. However, then there will be an automorphism of $\mathcal{T}_e$ interchanging b with c and respecting this coloring, so φ_e failed to distinguish $\mathcal{T}_e$ by its coloring.

On the other hand, $\mathcal{T}_e$ is 2-distinguishable, exactly as above: just color b and c different colors, and then use the balanced coloring above each of them, while coloring the complete binary tree above a with any distinguishing 2-coloring of $2^{<\omega}$. (Clearly no automorphism of this $\mathcal{T}_e$ can avoid fixing a, so the choice between balanced and unbalanced above a is irrelevant.)

Finally, we wish to combine these basic modules to build a single computable tree $\mathcal{T}$, with computable finite branching, which has distinguishing number 2 but has no computable distinguishing 2-coloring. This is straightforward. Start with a spine $d_0 < d_1 < d_2 < \cdots$, among which d_0 will be the root of $\mathcal{T}$. Above each d_{2e+1}, in addition to d_{2e+2}, we place a node r_e. Then we build a copy of the tree $\mathcal{T}_e$ with r_e as its root, diagonalizing against the possible coloring φ_e exactly as above in Theorem 21, using the three successors a_e, b_e, and c_e of r_e. No φ_e can be a distinguishing 2-coloring of the entire tree $\mathcal{T}$, because, assuming φ_e is total, there will be some nontrivial automorphism of $\mathcal{T}_e$ which respects the coloring φ_e, and this automorphism extends to an automorphism of all of $\mathcal{T}$ just by fixing the rest of $\mathcal{T}$ pointwise. $\square$

The branching function of a computable finite-branching predecessor tree is always $\mathbf{0}'$-computable. However, it turns out that even with a $\mathbf{0}'$-oracle, we could not necessarily compute a distinguishing 2-coloring of such a tree with distinguishing number 2.

Theorem 24 *There is a computable finite-branching predecessor tree which is distinguished by a 2-coloring but not by any $\mathbf{0}'$-computable 2-coloring.*

Proof. This proof uses a simple modification to the trees $\mathcal{T}_e$ used to build $\mathcal{T}$ in Theorem 23. Now that the branching is allowed to be noncomputable, we may temporarily stop building the tree $\mathcal{T}_e$ at levels $> n$ above b and c (when φ_e has given the same color to b and c), and then resume building the complete binary tree above b and c when/if φ_e "changes its mind" about its coloring of b and c. This makes the branching (above these nodes at level n) noncomputable, since we do not know whether we will ever add nodes at level $n+1$. However, it enables us to satisfy the following requirement.

$$\mathcal{R}_e : \lim_t \varphi_e(x,t) \text{ is not a distinguishing 2-coloring of } T_e.$$

These requirements together will show that $\mathcal{T}$ has no $\mathbf{0}'$-computable distinguishing 2-coloring, where $\mathcal{T}$ is built from the trees $\mathcal{T}_e$ exactly as before.

The alteration to the construction of $\mathcal{T}_e$ is simple. As before, wait for $\varphi_{e,s}(a,0)$, $\varphi_{e,s}(b,0)$, and $\varphi_{e,s}(c,0)$ to halt with values in $\{0,1\}$. Pick two of them which have the same value, and stop building the binary trees above those two nodes (while continuing to build a binary tree above the third node). Meanwhile, wait for $\varphi_{e,s}(a,1)$, $\varphi_{e,s}(b,1)$, and $\varphi_{e,s}(c,1)$ to halt with values in $\{0,1\}$. When and if this happens, these values supersede those from before: for example, if previously we had halted construction above b and c (as in the original description of $\mathcal{T}_e$), but now $\varphi_e(a,1) = \varphi_e(b,1) = 0 \neq \varphi_e(c,1)$, then we build up the trees above b and c until all three have the same height, then continue building the tree above c but stop building the ones above a and b. On the other hand, if $\varphi_{e,s}(a,1) = \varphi_e(a,0)$, $\varphi_{e,s}(b,1) = \varphi_e(b,0)$, and $\varphi_{e,s}(c,1) = \varphi_e(c,0)$, then φ_e has not changed its mind, and we do not resume construction above the two nodes above which it was stopped. We then continue on to consider $\varphi_e(a,2)$, etc., using the same program relative to the values $\varphi_e(a,1)$, etc., and so on for all t.

If $\lim_t \varphi_e(x,t)$ exists for all three values $x \in \{a,b,c\}$, then we wind up in the same situation as in the previous proof, showing that this limit cannot be a distinguishing 2-coloring of $\mathcal{T}_e$, so that $\mathcal{R}_e$ holds. On the other hand, if the limit fails to exist (but φ_e is total with range $\subseteq \{0,1\}$), then we simply built B_3 above r_e, and B_3 is indeed 2-distinguishable (although not by $\lim_t \varphi_e(x,t)$). Finally, if φ_e is not total or assumes values > 1, then $\mathcal{R}_e$ will hold, and the $\mathcal{T}_e$ we build is either a copy of B_3 or a copy of some $B_{3,n}$, both of which are 2-distinguishable. (Technically, even if range(φ_e) contains some values > 1, the limit could still have values 0 and 1 only. However, if this holds, then some other $\varphi_{e'}$ would have the same limit and would have range $\subseteq \{0,1\}$, so that $\mathcal{R}_{e'}$ would have taken care of showing that $\lim_t \varphi_e$ was not a distinguishing 2-coloring.) □

So how much information would we need to compute a distinguishing 2-coloring of a computable finite-branching predecessor tree with distinguishing number 2? We answer this below in Theorem 25, but we begin by defining an *extendible* node of a tree to be a node which lies on an infinite path.

Lemma 2. *If all nodes of a computable predecessor tree T are extendible, then T is 2-distinguishable.*

Proof. This is true even if the tree is infinite-branching, so we prove it for this more general case. Suppose we have a computable predecessor tree with every node extendible. Label the nodes at level 1 of the tree $x_1, x_2, \ldots$ in order of their enumeration into the tree. Color x_1 striped. Color x_2 solid, and color every node at level 2 above x_2 striped. Color x_3 solid, color every node at level 2 above x_3 solid, and color every node at level 3 above x_3 striped, and so on for $x_4, x_5, \ldots$. This procedure distinguishes each node at level 1 from every other node at level 1.

Fix an $n > 0$, and consider the immediate (level-2) successors $y_1, y_2, \ldots$ of x_n. These may have already been colored by the previous instructions; in fact, their successors up to level n will already be colored. Color the level-$(n+1)$ successors of y_1 striped, the level-$(n+1)$ successors of y_2 solid and its level-$(n+2)$ successors striped, then the same above y_3 with solid-solid-striped, and so on. When we do this for every n, each node at level 2 is distinguished from all its siblings at level 2.

We continue in this vein to distinguish each node at level k from every other node at level k, for every k.

Here is the algorithm for determining the color of an arbitrary node on such a tree if we've used the above coloring:

```
    Choose a node on the tree. Call it $n$. Call the root $r$.
($\star$) Use the predecessor function to determine the level of $n$ above $r$. Call this level $l$.
Label the immediate successors of $r$ with $x_1, x_2, \ldots$.
If $n$ sits above $x_l$, then $n$ is red.
Else, if $n$ sits above $x_i$ for some $i > l$, then $n$ is blue.
Else, $n$ sits above $x_i$ for some $i < l$. Let this $x_i$ be the new $r$, and go back to ($\star$).
```
□

Theorem 25 *If a computable finite-branching predecessor tree has distinguishing number* 2*, then it has a* $\mathbf{0}''$*-computable distinguishing* 2*-coloring.*

Proof. Because the tree is finite-branching, with a $\mathbf{0}''$-oracle we can determine, for each immediate successor y of a given node x, whether y is extendible or not: König's Lemma states that a non-extendible node must have only finitely many nodes extending it. Above the extendible ones we use the process illustrated above in Lemma 2. For each extendible x, consider the (finite) subtree containing x, the non-extendible immediate successors of x and all of their successors. There must be a way to distinguish this subtree with a 2-coloring, since the tree has a distinguishing 2-coloring. Each non-extendible node has only finitely many nodes above it. With the $\mathbf{0}''$-oracle we can find them all, and we can try out all the possible colorings until we find one which admits no non-trivial automorphism. Having done all of this, we know that every automorphism of the tree which respects our coloring and which fixes x must also fix each immediate successor y (and all the nodes above each non-extendible y). By induction on levels, this means that an automorphism which respects the coloring must fix each single node, i.e. must be the identity. □

Thus the existence of a distinguishing 2-coloring is equivalent to the existence of such a coloring computable in $\mathbf{0}''$. This significantly reduces the complexity of the property (for computable finite-branching trees $\mathcal{T}$ under predecessor) of being 2-distinguishable. On its face, that property was Σ^1_2: it said that there exists a function (the coloring) such that every automorphism of $\mathcal{T}$ either fixes all nodes or disrespects the coloring. Of course, this complexity can quickly be reduced, since the complexity of an orbit in a finite-branching computable predecessor tree is at most Π^0_2. Nevertheless, 2-distinguishability still could have been Σ^1_1-hard for these trees, up until we established Theorem 25, which showed that we need only quantify over $\mathbf{0}''$-computable functions, not over all functions, to define 2-distinguishability.

We now investigate how much further we can lower the complexity of the property of 2-distinguishability, and whether the same complexity level holds for n-distinguishability. Theorem

26 answers these questions. Subsequently we will consider distinguishability by finite colorings, i.e., colorings with finitely many colors, but with no fixed bound on the number of colors.

Theorem 26 *For each fixed n, the property of having a distinguishing n-coloring is Π^0_2-complete within the class of finite-branching predecessor trees.*

Proof. We will show that having a distinguishing 2-coloring is Π^0_2-complete, and we will explain how the argument extends to an n-coloring for fixed n.

A finite-branching predecessor tree has **no** distinguishing 2-coloring if and only if

$$\exists \sigma_1, \ldots, \sigma_k \begin{bmatrix} \sigma_1, \ldots, \sigma_k \text{ are not extendible and have a common predecessor } \tau \;\& \\ \text{the tree } \{\tau\} \cup \bigcup_{i=1}^{k} \{\delta : \delta \supseteq \sigma_i\} \text{ has no distinguishing 2-coloring} \end{bmatrix}.$$

Non-extendibility is a Σ^0_2 property, and the other two conjuncts inside the brackets are each computable. So, **not** having a distinguishing 2-coloring is Σ^0_2. Thus, having a distinguishing 2-coloring is Π^0_2. Notice that if "2-coloring" in the above argument were replaced with "n-coloring," the result would still hold; so, for fixed n, having a distinguishing n-coloring is Π^0_2.

To show completeness, we start by building a copy of the tree B_3 as follows. Begin with the root and the three nodes at level 1. Then, whenever a new element is enumerated into the e-th c.e. set W_e, we add the whole next level of B_3 to our tree. If W_e turns out to be finite, our copy of B_3 will only have finitely many levels, and thus will not be 2-distinguishable. If W_e turns out to be infinite, then our B_3 will likewise be infinite, and thus 2-distinguishable. So the tree has a distinguishing 2-coloring just if W_e is infinite.

If we define B_{n+1} to be the tree whose root has exactly $(n+1)$ immediate successors and every other node has exactly n immediate successors, then if we replace B_3 with B_{n+1} in the immediately preceding paragraph, we show that having a distinguishing n-coloring for some fixed n is Π^0_2-complete as well. $\square$

It remains to consider "having a distinguishing n-coloring for some (arbitrary) n." This is the property of being *finitely distinguishable*. Expressing this property takes an extra $\exists$ quantifier, so it is plausible that having a finite distinguishing coloring is Σ^0_3-complete.

Theorem 27 *The property of having a distinguishing finite coloring is Σ^0_3-complete within the class of finite-branching predecessor trees.*

Proof. Define S_∞ to be the following tree: There is an infinite "spine," and the node α_n on the spine at level n has exactly n additional immediate successors, all of which are terminal. We will show, using the tree S_∞, that being finitely distinguishable is Σ^0_3-complete by giving a 1-reduction from the set of indices for finitely distinguishable trees to the set Cof.

We start by building the tree S_∞. Let A_k be the subtree consisting of α_k and all its non-extendible immediate successors. We wait for elements to be enumerated into W_e. At stage s, suppose $m \in W_{e,s} - W_{e,s-1}$. Then we make A_m rigid by adding paths of distinct finite lengths above the immediate successors of α_m.

We claim that the resulting tree is finitely distinguishable if and only if W_e is cofinite. Suppose $\overline{W_e}$ is finite and nonempty. (If $\overline{W_e}$ is empty, then the tree is rigid, i.e., 1-distinguishable.) Then every tree A_k with $k > \max(\overline{W_e})$ is rigid. Thus the whole tree is $\max(\overline{W_e})$-distinguishable. Now suppose $\overline{W_e}$ is infinite. Then, given any n, there is A_k with $k > n$ for which A_k is k-distinguishable but not n-distinguishable. Thus, for each n, the whole tree is not n-distinguishable. So the tree has a finite distinguishing coloring just if W_e is cofinite. $\square$

3 Further Questions

The most natural next question is:

Question 1. What are the analogous results for computable infinite-branching predecessor trees?

We have already begun answering this question, and results will be forthcoming in an expansion of this abstract.

In [7], we showed that many computability-theoretic results which were true of finite-branching predecessor trees were also true of finite-valence pointed graphs, and so we also ask:

Question 2. Do any of these results carry over to computable finite-valence pointed graphs?

This is our expected next step after considering the infinite-branching predecessor trees.

References

1. W. Imrich, S. Klavzar, & V. Trofimov; Distinguishing infinite graphs, *Electronic Journal of Combinatorics*, **14** (2007), #R36.
2. V. Harizanov, R. Miller, & A. Morozov; Simple structures with complex symmetry, *Algebra and Logic* **49** (2010), 51-67.
3. S. Lempp, C. McCoy, R.G. Miller, & R. Solomon; Computable categoricity of trees of finite height, *Journal of Symbolic Logic* **70** (2005), 151–215.
4. R.G. Miller; The computable dimension of trees of infinite height, *Journal of Symbolic Logic* **70** (2005), 111–141.
5. S. M. Smith & M. E. Watkins; Bounding the distinguishing number of infinite graphs, submitted for publication.
6. S. M. Smith, T. W. Tucker, & M. E. Watkins; Distinguishability of infinite groups and graphs, *Electronic Journal of Combinatorics* **19** (2012), #R27.
7. R. M. Steiner; Effective algebraicity, *Archive for Mathematical Logic* **52** (2013), 91-112.