ON THE DEFINABILITY OF THE DOUBLE JUMP IN THE COMPUTABLY ENUMERABLE SETS

PETER A. CHOLAK AND LEO A. HARRINGTON

ABSTRACT. We show that the double jump is definable in the computably enumerable sets. Our main result is as follows: Let $\mathcal{C} = \{\mathbf{a} : \mathbf{a}$ is the Turing degree of a Σ^0_3 set $J \geq_T \mathbf{0}''\}$. Let $\mathcal{D} \subseteq \mathcal{C}$ such that $\mathcal{D}$ is upward closed in $\mathcal{C}$. Then there is an $\mathcal{L}(A)$ property $\varphi_{\mathcal{D}}(A)$ such that $F'' \in \mathcal{D}$ iff there is an A where $A \equiv_T F$ and $\varphi_{\mathcal{D}}(A)$. A corollary of this is that, for all $n \geq 2$, the high$_n$ ($\overline{\text{low}_n}$) computably enumerable degrees are invariant in the computably enumerable sets. Our work resolves Martin's Invariance Conjecture.

1. INTRODUCTION

We will work in the computably enumerable sets. The language is just inclusion, $\subseteq$. This structure is called $\mathcal{E}$. Our notation and definitions are standard and follow Soare [16]. As a safety measure all sets will be computably enumerable noncomputable sets and all degrees will be computably enumerable and noncomputable, unless otherwise noted.

We say a class $\mathcal{D}$ of computably enumerable degrees is *invariant* if there is a class $\mathcal{S}$ of computably enumerable sets such that $\mathbf{d} \in \mathcal{D}$ implies there is a W in $\mathcal{S}$ and $\mathbf{d}$, $W \in \mathcal{S}$ implies $\deg(W) \in \mathcal{D}$ and $\mathcal{S}$ is closed under automorphic images.

1.1. **The Martin's Invariance Conjecture.** A coinfinite set $M \in \mathcal{E}$ is maximal if for all $W \in \mathcal{E}$ either $W \subseteq^* M$ or $W \cup M =^* \omega$. A set $A \in \mathcal{E}$ is atomless if A has no maximal superset. Martin [12] showed that the degrees of maximal sets are exactly the high degrees. Lachlan [9] and Shoenfield [14] showed that the degrees of atomless sets are exactly the non-low$_2$ degrees. So the high degrees and the non-low$_2$ degrees are invariant. In addition, $\mathbf{L_0} = \mathbf{0}$ and $\overline{\mathbf{L_0}} = \mathbf{R} - \mathbf{0}$ are invariant; use the computable sets and noncomputable sets, respectively. These results and Martin's work on projective determinacy led Martin to conjecture in the late 1960's:

1991 *Mathematics Subject Classification.* Primary 03D25.

Key words and phrases. definability, computably enumerable, double jump.

This research was partially supported by NSF Grants DMS-96-34565 and DMS 99-88716 (Cholak) and NSF Grant DMS-96-22290 (Harrington). The authors wish to thank Bob Soare for all his interest and his many helpful comments.

Conjecture 1.1 (Martin's Original Invariance Conjecture)**.** *Among all non-trivial degree classes the invariant classes are exactly* $\mathbf{H_{2n-1}}$ *and* $\overline{\mathbf{L_{2n}}}$.

The alternation of every odd $\mathbf{H_n}$ and even $\mathbf{L_n}$ was inspired by the above-mentioned results of Martin and Shoenfield and the behavior of projective determinacy. As more was known the conjecture was modified slightly. Lerman and Soare [10] showed that the d-simple sets form a definable class which splits the low degrees. So at this time the conjecture was restricted to the jump classes $\mathbf{H_n}$ and $\mathbf{L_n}$, for $n > 0$, and their complements.

Conjecture 1.2 (Martin's Invariance Conjecture)**.** *Among jump classes* $\mathbf{H_n}$ *and* $\mathbf{L_n}$*, for* $n > 0$*, and their complements, the invariant classes are exactly* $\mathbf{H_{2n-1}}$ *and* $\overline{\mathbf{L_{2n}}}$.

The conjecture was never published; the only published version can be found in Harrington and Soare [7].

1.2. **Work related to Martin's Conjecture.** The conjecture divides into two types of jump classes: the upward closed jump classes, $\mathbf{H_n}$ and $\overline{\mathbf{L_n}}$, for all n, and the downward closed jump classes, $\mathbf{L_n}$ and $\overline{\mathbf{H_n}}$, for all n. The conjecture implies that no downward closed jump classes are invariant. This was verified earlier. Cholak [2] and, independently, Harrington and Soare [8] showed that every noncomputable set is automorphic to a high set. So the only invariant jump classes must be upward closed.

Harrington, in unpublished work, showed that the property of being creative is elementary definable in $\mathcal{E}$. Hence, $\mathbf{H_0}$ is invariant. Harrington and Soare [8] showed that every prompt set is automorphic to a complete set and hence $\overline{\mathbf{H_0}}$ is non-invariant. Maass et al. [11] showed that there is a definable class of sets which splits all jump classes. Hence no one orbit can positively answer the conjecture for any of the remaining classes. Also, Harrington and Soare [6] have shown that $\overline{\mathbf{L_1}}$ is not invariant [see 7, Corollary 4.4] , further suggesting the Martin's conjecture is correct. They prove this by showing there is a properly low$_2$ degree $\mathbf{d}$ such that if $A \leq_T \mathbf{d}$ then there is a low B such that A and B are automorphic. (Note the similarities between this and Conjecture 1.6.)

1.3. **Our work.** The culmination of our work is the following theorem and corollaries which show that in some sense the double jump can be encoded into $\mathcal{E}$ and as a result resolves the Martin's conjecture; it is not true.

Theorem 8.5. *Let* $\mathcal{C} = \{\mathbf{a} : \mathbf{a}$ *is the Turing degree of a* Σ^0_3 *set* $J \geq_T \mathbf{0}''\}$. *Let* $\mathcal{D} \subseteq \mathcal{C}$ *such that* $\mathcal{D}$ *is upward closed in* $\mathcal{C}$. *Then there is an* $\mathcal{L}(A)$ *property* $\varphi_{\mathcal{D}}(A)$ *such that*

$$(\forall c.e.\ F)[F'' \in \mathcal{D} \iff (\exists A)[\varphi_{\mathcal{D}}(A) \text{ and } A \equiv_T F]].$$

The $\mathcal{L}(A)$ property $\varphi_{\mathcal{D}}(A)$ is defined in Definition 8.4. The theorem gives us some very nice corollaries.

Corollary 1.3. *For all $n \geq 2$, the high$_n$ ($\overline{\text{low}_n}$) computably enumerable degrees are invariant.*

Proof. Let $\mathcal{D} = \{\mathbf{a} : \mathbf{a}$ is the Turing degree of the double jump of some high$_n$ ($\overline{\text{low}_n}$) degree$\}$. So $\varphi_{\mathcal{D}}(A)$ iff A is high$_n$ ($\overline{\text{low}_n}$). □

Corollary 1.4. *Let $\mathcal{F}$ be a class of computably enumerable degrees such that if $\mathbf{a} \in \mathcal{F}$ and $\mathbf{a}'' \leq_T \mathbf{b}''$ then $\mathbf{b} \in \mathcal{F}$. Then $\mathcal{F}$ is invariant.*

Proof. Let $\mathcal{D} = \{\mathbf{a}'' : \mathbf{a} \in \mathcal{F}\}$. □

Recall that we say $A \simeq B$ iff there is an automorphism Φ of $\mathcal{E}$ such that $\Phi(A) = B$.

Corollary 1.5. *If $\mathbf{a}'' > \mathbf{b}''$ then there is an $A \in \mathbf{a}$ such that for all $B \in \mathbf{b}$, $A \not\simeq B$ (in fact, $\mathcal{L}(A) \not\cong \mathcal{L}(B)$).*

Proof. Let $\mathcal{D} = \{\mathbf{d}'' : \mathbf{d} \geq_T \mathbf{a}\}$. Now there is an $A \in \mathbf{a}$ such that $\varphi_{\mathcal{D}}(A)$. If $B \in \mathbf{b}$ then $\neg\varphi_{\mathcal{D}}(B)$. So A and B cannot have isomorphic $\mathcal{L}$'s. □

1.4. **Some questions.** More or less we have shown that the double jump can be encoded into $\mathcal{E}$. But what about the single jump? We conjecture that this cannot be done. The following conjecture is the strongest possible negation of encoding the single jump in the above fashion (see Corollary 1.4) and is a generalization of the result of Harrington and Soare that $\overline{\mathbf{L_1}}$ is noninvariant.

Conjecture 1.6. *Let J be c.e. in and above $\mathbf{0}''$. There are degrees $\mathbf{a}$ and $\mathbf{b}$ such that $\mathbf{a}' \neq \mathbf{b}'$, $\mathbf{a}'' = \mathbf{b}'' = J$ and, for all $A \leq_T \mathbf{a}$, there is a B such that $B \leq_T \mathbf{b}$ and $A \simeq B$.*

The following theorems provide some evidence for the above conjecture.

Theorem 1.7 (Cholak [2])**.** *For all A and for all $\mathbf{H_1}$ degrees $\mathbf{h}$ there is a $B \in \mathbf{h}$ such that $\mathcal{L}^*(A) \cong \mathcal{L}^*(B)$.*

Recently Harrington has announced a generalization of the above theorem:

Theorem 1.8 (Harrington)**.** *For all A and degrees $\mathbf{d}$ if $A' \leq_T \mathbf{d}'$ there is a $B \in \mathbf{d}$ such that $\mathcal{L}^*(A) \cong \mathcal{L}^*(B)$.*

Since the coding is naturally upwardly closed (see the paragraph after Definition 8.4), the above conjecture suggests that Theorem 8.5 is the strongest possible theorem along these lines.

1.5. **The encoding.** Our work relies on a new definable encoding. The actual details of this encoding are slightly complex. These details and some theorems concerning this encoding will appear in the next few sections.

There is another application of this encoding and certainly there will be more. In our other application, we showed that if A and B are automorphic (in $\mathcal{E}$) via Φ then isomorphism between the splits of A modulo the computable subsets of A and the splits of B modulo the computable subsets of B induced by Φ is Δ^0_3. This result will appear in Cholak and Harrington [3]. We will not discuss this result concerning automorphisms further here but direct the reader to Cholak and Harrington [4] for an old announcement of some of our current results which use this new definable encoding.

The encoding relies on the existence of infinitely many of what we call *special $\mathcal{L}$-patterns* $\mathcal{P}_i$ and formulas $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, S)$ (where $\mathcal{L} = \{0, 1, 2, 3\}$, $\vec{U}$ is a 4-tuple and $\vec{B}$ is an n-tuple for some n determined by $\mathcal{P}$). What is important about $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ is that it is a statement in the language $L = \{\subseteq\}$ (in fact it is an $\mathcal{L}(A)$-property), it is Σ^0_3 and it is invariant under automorphisms of $\mathcal{E}$ (so if Φ is an automorphism of $\mathcal{E}$ then $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ iff $\varphi_{\mathcal{P}}(\Phi(A), \Phi(\vec{U}), \Phi(\vec{B}), \Phi(C))$). In Section 3, we will define the patterns $\mathcal{P}_i$, define what $\varphi_{\mathcal{P}_0}(A, \vec{U}, \vec{B}, C)$ means, and provide several examples.

Fix $\vec{U}$. Informally, we say A *realizes* $\mathcal{P}_i$ (A might be $\emptyset$) if for all $\vec{B}$ there is a C such that $\varphi_{\mathcal{P}_i}(A, \vec{U}, \vec{B}, C)$ and C is not computable modulo A (see Section 3.5 for a definition). In other applications of the coding the last condition on C, "C is not computable modulo A", can and will be changed. The condition on C for the application concerning automorphisms is discussed in Cholak and Harrington [4] and, of course, will appear in Cholak and Harrington [3]. In both cases, there is uniformly a universal $\vec{B}$; i.e., a $\vec{B}_e$ such that if $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, C)$ holds for some C then A realizes $\mathcal{P}$ (where e codes A, $\vec{U}$ and $\mathcal{P}_i$). Theorem 4.1 implies that a universal $\vec{B}$ exists for this application of the encoding.

The *key idea* is that it is possible to construct sets (mainly $\vec{U}$) to realize certain chosen $\mathcal{P}_i$'s while avoiding the realization of the other $\mathcal{P}_j$'s. This idea is at the technical heart of our results. In Theorem 5.1 we isolate this idea. We carefully chose our patterns such that this theorem would hold. These theorems, in turn, are used to prove Theorem 6.7 which is a technical theorem at the heart of the coding of the double jump.

We feel that this encoding has great potential. In this paper, we have only mentioned our two recent applications. But surely there will be others. For example, one potential application of the encoding is to use it to show the $\exists\forall\exists$ theory of $\mathcal{E}^*$ is undecidable.

We were not the first to develop methods for coding in $\mathcal{E}$. Harrington developed various coding methods in $\mathcal{E}$ to show that the theory of $\mathcal{E}$ is undecidable and has degree $\mathbf{0}^{(\omega)}$ (see [1]). This has been further developed in Harrington and Nies [5]. The culmination of these earlier methods of coding in $\mathcal{E}$ is Harrington's Ideal Definability Theorem which says that every Σ^0_k-ideal of $\mathcal{I}_{\mathcal{R}}(A)$ is uniformly (in k) definable, where k is odd and greater than 1. The fact that the theorem is uniform allows us to quantify over the Σ^0_k-ideals for any k. Informally to use this theorem, one constructs a set A and various definable ideals and then quantifying over the Σ^0_k-ideals allows us to encode the desired structure.

Our current encoding is simpler to decoding than the previous encoding. More or less, it can be decoded at the Σ^A_3 level. Hence we should be able to use it to get sharper results. Our current encoding follows naturally (to those familiar with automorphisms of $\mathcal{E}$) from how balls move and flow in the construction of automorphisms of $\mathcal{E}$.

2. The Players

2.1. **An automorphism construction.** As we claimed above, our encoding follows naturally (to those familiar with automorphisms of $\mathcal{E}$) from how balls move and flow in the construction of automorphisms of $\mathcal{E}$. So we need to start with a very informal description of an automorphism construction. For a complete description of the Δ^0_3-automorphism machinery, in the notation we will be using, we will send the reader to Harrington and Soare [8].

In a random automorphism construction, the goal might be to show two computably enumerable sets, A and B, are automorphic. We think of there being two players, RED and BLUE. All of the computably enumerable sets involved have their enumeration controlled by either RED (we say RED *plays* these sets) or BLUE. RED plays A and depending on the construction may play B.

BLUE's goal is to build an automorphism taking A to B. RED wins if there is no such automorphism. Of course, RED wins iff there is an $\mathcal{L}_{\omega_1,\omega}$ formula $\varphi(X)$ (using just $=$ and $\subset$, the language of $\mathcal{E}$) such that $\varphi(A)$ and $\neg\varphi(B)$. So we say BLUE is the automorphism player and RED is the definability player. We might have called an automorphism result a BLUE result and ask the readers to identify with the BLUE player while reading about this automorphism result. Similarly we would call a result showing two sets have a definable difference a RED result and ask the readers of this result to identify with RED.

In an automorphism construction, balls (elements of ω) start in some state and then may or may not move into many different states and so on.

For a ball x to move into another state means that x has entered one of the sets being played by BLUE or RED. So we think of the movement of balls into states as being controlled by BLUE and RED. (Of course, in the actual construction this action is localized at a node on the tree.)

There are many different types of moves available to BLUE and RED. For example, one move is for RED to empty out a state ν; every ball x that enters ν enters another RED set and hence leaves ν. Of course, there is a similar move for BLUE. All RED moves have a dual BLUE and similarly for BLUE.

2.2. **The players in this construction.** The supposed roles of BLUE and RED will be different in this construction than in previous constructions. As in previous constructions RED and BLUE will control the enumeration of some computably enumerable sets. The sets that BLUE and RED control will be formalized in the patterns. In each pattern BLUE will control some sets and RED some sets. Unlike previous constructions, BLUE moves do not have a RED dual and similarly for RED (this will become clear in Section 3.1.) In addition, not all the computably enumerable sets involved in a random pattern will be controlled by either BLUE or RED.

There is another player who among other things controls the enumeration of some of the computably enumerable sets involved in a pattern. In previous constructions the authors might have asked the readers to identify with either BLUE or RED; here we do not. Here we ask the reader (and the authors) to identify with this third player. We might think of this third player as us — the readers and the authors.

To us the players RED and BLUE are just pawns. We will play the sets which are not played by BLUE or RED. We also have the option of taking over in any pattern the sets played by BLUE or the sets played by RED. That is, we can side with BLUE or RED if desired. In any one pattern if we take over the sets played by BLUE, we cannot take over the sets played by RED in that same pattern and similarly if we take over the RED sets, we cannot also take over the BLUE sets. But we might take over the BLUE sets in one pattern and the RED sets in another pattern. We will be careful to point out the roles of the players in each of our settings.

The issue of the players becomes particularly critical in the middle of the proof of Theorem 5.1. As a result in the middle of the proof of Theorem 5.1 we will take a quick break and reflect upon the roles of our players in this particular situation. So for more see Section 5.1.5.

3. The Patterns

3.1. **An informal example.** We will start with an informal example which we will later make formal and generalize. Throughout this paper, we will

frequently refer to Figure 1. (The reader might wish to sketch a copy on a loose sheet of paper.) As we will see this figure describes the pattern $\mathcal{P}_0$.

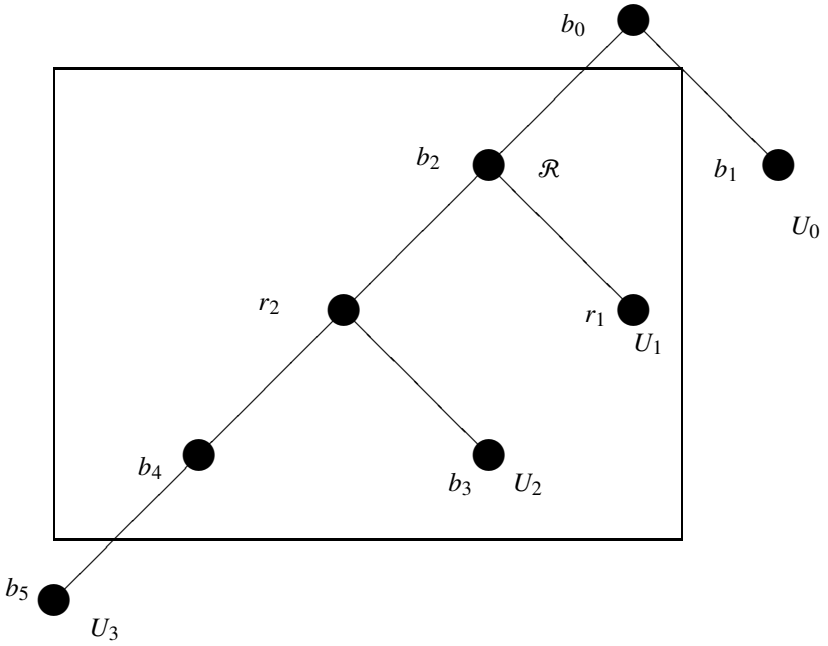

FIGURE 1. The special $\mathcal{L}$-pattern $\mathcal{P}_0$.

Ignoring the box and the labels and considering the solid circles as nodes, Figure 1 describes a tree. We will call this tree $\mathcal{T}$. (This tree is finite and very different from a tree of strategies.) The node labeled b_0 is the top or root. We will use the b_i's and r_i's as labels for the nodes in $\mathcal{T}$.

Corresponding to each node there will be a computably enumerable set D_{b_i} or D_{r_i}. At the nodes labeled by a U_i we can think of there being an additional set U_i. We want to think of the pattern as how an ball entering the set D_{b_0} can enter one of the U_i's. One might think of the U_i's as states.

Consider a ball x. We will only focus on those balls which are in C and not in A; both C and A are external sets. Such a ball x enters the pattern by entering the set D_{b_0} (but x is not in any of the other sets). After entering D_{b_0}, x might enter one of D_{b_1} or D_{b_2} or it might not enter either of these sets. If x enters D_{b_1} then it must enter U_0. If x enters D_{b_2} it might enter either D_{r_1} or D_{r_2}, and so on. If x enters U_0 then x must be in D_{b_1}. Etc.

Now to this we must add the BLUE and RED moves. There is only one RED move in this pattern. If a ball enters the set D_{b_2} it *must* enter one of the sets D_{r_1} or D_{r_2}; x cannot remain just in the set D_{b_2}. So RED controls

the enumeration of elements once in D_{b_2} in the sense that a ball in D_{b_2} must enter D_{r_1} or D_{r_2}. This is like the emptying of a state into a higher state in the automorphism construction. Notice that b_2 is the only node labeled by $\mathcal{R}$.

Other than the above RED move all the remaining possible moves on this pattern are up to BLUE. BLUE decides whether a ball enters any of the D_{b_i}'s. This will be done by external sets, B_{b_i}'s. Generally $x \in D_{b_i}$ iff $x \in B_{b_i}$. Whether a ball x in D_{b_0} enters D_{b_1}, D_{b_2}, or not depending on whether x enters one of the external sets B_{b_1} or B_{b_2}. The external sets for BLUE, $\vec{B}$, might and will change. We can think of each $\vec{B}$ as one possible strategy for BLUE to play. Note that the RED and BLUE moves do not have duals. While BLUE's strategies can change, RED's remain unchanged; put all the balls in D_{b_2} into D_{r_1} or D_{r_2}.

What we have (informally) described is the pattern $\mathcal{P}_0$ and how things would behave if the formula $\varphi_{\mathcal{P}_0}(A, \vec{U}, \vec{B}, C)$ holds (see Definition 3.13). In the next two sections it will become clear that the formula $\varphi_{\mathcal{P}_0}(A, \vec{U}, \vec{B}, C)$ begins with there is a $\vec{D}$ such that some Π^0_2 conditions hold. Normally we are given A and C, and we build $\vec{U}$ and $\vec{D}$ in response to BLUE's strategy $\vec{B}$. One way to view the above action is as the U_i's as states played by us (not RED or BLUE) and the sets D_{b_i}'s, D_{r_i}'s and B_{b_i}'s and the corresponding BLUE and RED moves as how a ball might enter one of these states.

In this paper we will just focus on the special $\mathcal{L}$-patterns $\mathcal{P}_n$ which are the patterns used for our encoding (see Theorem 5.1), although the definitions below are given for any special $\mathcal{L}$-pattern. The special $\mathcal{L}$-patterns $\mathcal{P}_n$ are just iterations of the pattern $\mathcal{P}_0$. For special $\mathcal{L}$-pattern $\mathcal{P}_n$ we just repeat the part of the pattern in the box in $\mathcal{P}_0$ $n+1$ times. We will call this part of the pattern a *block*. So balls can flow from b_4 to another copy of b_2.

3.2. **The definitions.** The goal of this section is to formally define special $\mathcal{L}$-patterns and the formula $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$. The next few pages are rather intense, at least in terms of the notational complexity. The end result of the notational complexity is the definition of a special $\mathcal{L}$-pattern, $\mathcal{P}$, and the formula $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$. The actual definition of $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ appears in Section 3.3. The reader might also like to look above at Section 3.5, where we discuss how one realizes a pattern. Realizing and not realizing a pattern is used for coding in Theorem 5.1 which in turn will be used for our main theorem (Theorem 6.7).

During this subsection, we will use the informal example from Section 3.1 to illustrate our definitions. In Section 3.6, we will provide more

examples of special $\mathcal{L}$-patterns, in particular, the special $\mathcal{L}$-patterns used in Theorem 5.1.

Definition 3.1. For $\mathcal{L}$, a finite set, an *$\mathcal{L}$-interpretation* is a function $\langle U_i : i \in \mathcal{L}\rangle$ from $\mathcal{L}$ to $\mathcal{E}$.

If we let $\mathcal{L} = \{0, 1, 2, 3\}$ then U_1, U_2, U_3 and U_4 is an $\mathcal{L}$-interpretation. If we consider $\mathcal{T}$ from the above subsection as a finite set $\mathcal{L}$ with the b_i's and r_i's as elements then the computably enumerable sets D_{b_i}'s and D_{r_i}'s is an $\mathcal{L}$-interpretation.

Notation 3.2. $\mathcal{T}$ will always denote a finite tree. The ordering on $\mathcal{T}$ will be denoted by $<$. The node b_0 will always be the top or root of $\mathcal{T}$. Our trees will grow downward so $b_0 \geq p$, for all $p \in \mathcal{T}$. For $p \in \mathcal{T}$, if $p \neq b_0$ then p has an immediate predecessor $u(p)$

$$u(p) = q \text{ iff } q > p \text{ and } |q| = |p| - 1$$

(i.e., move *up* in the tree one node). For $q \in \mathcal{T}$, let

$$d(q) = \{p \in \mathcal{T} : u(p) = q\}$$

(i.e., the set of nodes directly *down* from q in $\mathcal{T}$). q is minimal iff $d(q) = \emptyset$.

Fix a computably enumerable set A (A might be the empty set).

Definition 3.3. For $\mathcal{T}$ a finite partial order, a *$\mathcal{T}$-interpretation* $\langle D_p : p \in \mathcal{T}\rangle$ is an $\mathcal{L}$-interpretation (for $\mathcal{L} = \mathcal{T}$) with the additional requirement that if $p \leq q$ then $D_p - A \subseteq D_q$.

So in our informal example, $\vec{D}$ is $\mathcal{T}$-interpretation iff $D_{b_1} - A \subseteq D_{b_0}$, $D_{b_2} - A \subseteq D_{b_0}$, $D_{r_1} - A \subseteq D_{b_2}$, $D_{r_2} - A \subseteq D_{b_2}$, $D_{b_4} - A \subseteq D_{r_2}$, $D_{b_3} - A \subseteq D_{r_2}$, and $D_{b_5} - A \subseteq D_{b_4}$ (and the D's are computably enumerable sets).

If $p \leq q$ then consider "$D_p - A \subseteq D_q$" as a *condition* of the computably enumerable sets D_p, D_q and A. Since $\mathcal{F}$ is finite, whether $\langle D_p : p \in \mathcal{T}\rangle$ is an $\mathcal{L}$-interpretation $\mathcal{T}$ corresponds to a finite conjunction of conditions. Each condition is a Π^0_2 condition:

$$\begin{gathered} D_p - A \subseteq D_q \text{ iff} \\ (\forall x)(\forall s)(\exists t)[x \in D_{p,s} \implies (x \in A_t \vee x \in D_{q,t})]. \end{gathered} \tag{3.1}$$

Hence whether an $\mathcal{L}$-interpretation (for $\mathcal{L} = \mathcal{T}$) is a $\mathcal{T}$-interpretation is Π^0_2 in the parameters $\vec{D}$ and A.

Definition 3.4. $(\mathcal{T}, \mathcal{R})$ is a *pure pattern* if $\mathcal{T}$ is a finite tree, $\mathcal{R} \subseteq \mathcal{T}$ and $\mathcal{R}$ does not contain any minimal elements of $\mathcal{T}$.

In our figures we consider the nodes labeled by $\mathcal{R}$ as the set $\mathcal{R}$. So in Figure 1, $\mathcal{R} = \{b_2\}$ and clearly $(\mathcal{T}, \mathcal{R})$ is a pure pattern. Informally, if a node is in $\mathcal{R}$ and φ holds then RED must move all balls which arrive at that node further down the tree. This is a RED forced move. Formally:

Definition 3.5. $\langle D_p : p \in \mathcal{T} \rangle$ is an *interpretation* of the pure pattern $(\mathcal{T}, \mathcal{R})$ if, in addition to being an interpretation of $\mathcal{T}$, for $q \in \mathcal{R}$, $D_q - A = \bigcup\{D_p : p \in d(q)\} - A$.

Assume $\vec{D}$ is an interpretation of the pure pattern $(\mathcal{T}, \mathcal{R})$. The idea is that if $q \in \mathcal{R}$ then RED moves all balls which enter the set, D_q, into either A or some D_p, where $p \in d(q)$. For example, if $\vec{D}$ is an interpretation of the pure pattern defined in Figure 1 then $D_{b_2} - A = (D_{r_1} \cup D_{r_2}) - A$; all the balls in D_{b_2} enter one of D_{r_1}, D_{r_2} or A. RED does not get to determine which of the D_p, where $p \in d(q)$, balls move into, only that they must move. The RED moves are determined by the nodes that are in the set $\mathcal{R}$.

The condition "$D_q - A = \bigcup\{D_p : p \in d(q)\} - A$" is Π^0_2 in the parameters D_q, D_p and A (the proof is similar to that presented in Equation 3.1). Since $\mathcal{R}$ is finite, a $\mathcal{T}$-interpretation is an interpretation of the pure pattern $(\mathcal{T}, \mathcal{R})$ iff a finite conjunction of Π^0_2 conditions holds. Hence whether a $\mathcal{L}$-interpretation is an interpretation of the pure pattern $(\mathcal{T}, \mathcal{R})$ is Π^0_2 in the parameters $\vec{D}$ and A.

Definition 3.6. $(\mathcal{T}, \mathcal{R}, \mathcal{B})$ is a *special pattern* iff $(\mathcal{T}, \mathcal{R})$ is a pure pattern, $\mathcal{B} \subseteq \mathcal{T}$, and for all $q \in \mathcal{R}$, $d(q) \cap \mathcal{B} = \emptyset$.

Informally if a node q is in $\mathcal{B}$ then BLUE controls the enumeration of D_q via the set B_q. If $q \in \mathcal{B}$ then BLUE has free choice to move balls out of the set D_q. BLUE's choice is given by the set B_q.

In Figure 1, the set $\mathcal{B}$ is all the nodes which are labeled by a b_i. Given that $\mathcal{R} = \{b_2\}$ and $d(b_2) = \{r_1, r_2\}$ this is the maximal set of nodes which can be in $\mathcal{B}$. For the patterns that we consider in this paper, it will always be the case that the set of nodes in $\mathcal{B}$ is the maximal set of nodes with respect to the property that for all $q \in \mathcal{R}$, $d(q) \cap \mathcal{B} = \emptyset$. Hence, in the patterns we consider here all movement is controlled by BLUE or RED. This need not always be the case.

Definition 3.7. A special pattern $(\mathcal{T}, \mathcal{R}, \mathcal{B})$ is *maximal* if $\mathcal{B}$ is the largest possible set w.r.t. the condition that for all $q \in \mathcal{R}$, $d(q) \cap \mathcal{B} = \emptyset$.

Definition 3.8. Let $\vec{B} = \langle B_p : p \in \mathcal{B} \rangle$ be a $\mathcal{B}$-interpretation, $\mathcal{P}$ be a special pattern (so $\mathcal{P} = (\mathcal{T}, \mathcal{R}, \mathcal{B})$) and $\vec{D} = \langle D_p : p \in \mathcal{T} \rangle$ be an interpretation of the pure pattern $(\mathcal{T}, \mathcal{R})$. Then $\vec{D}$ *is a* $\vec{B}$*-interpretation of* $\mathcal{P}$ *inside of* C iff for all $p \in \mathcal{B}$, $(D_p \cap C) - A = (B_p \cap D_{u(p)} \cap C) - A$ and $D_{u(b_0)} = C$.

Consider a ball x which does not enter A. Assume x is in $D_{u(p)} \cap C$. Then x enters B_p (a BLUE move) iff x enters D_p. Note that the BLUE moves are determined by the external sets $\vec{B}$; the BLUE choice is made by the sets $\vec{B}$. We think of the sets $\vec{B}$ as BLUE's strategy.

Like above (see Equation 3.1), $\vec{D}$ is a $\vec{B}$-interpretation of $\mathcal{P}$ inside of C is equivalent to a finite conjunction of Π^0_2 conditions and hence Π^0_2 in the parameters $\vec{D}$, $\vec{B}$, C and A.

For example, in the case of the pattern $\mathcal{P}_0$ defined in the last subsection, if $\vec{D}$ is a $\vec{B}$-interpretation of $\mathcal{P}_0$ inside of C then $(D_{b_0} \cap C) - A = (B_{b_0} \cap C) - A$, $(D_{b_1} \cap C) - A = (B_{b_1} \cap D_{b_0} \cap C) - A$, etc.

Remark 3.9. Let $\mathcal{P} = (\mathcal{T}, \mathcal{R}, \mathcal{B})$ be a maximal special pattern. Let $\vec{D}$ be such that $\vec{D}$ is an interpretation of the pure pattern $(\mathcal{T}, \mathcal{R})$ and $\vec{D}$ is a $\vec{B}$-interpretation of $\mathcal{P}$ inside of C. Then all movement into the D_p is determined by BLUE's strategy $\vec{B}$ and the RED forced moves. If we are given $\vec{B}$ and building $\vec{D}$ the only choice we have is if $p \in \mathcal{R}$ and x is in D_p then we can choose the D_q that x enters from the $q \in d(p)$. If we are building $\vec{B}$ and given $\vec{D}$ then if $p \in \mathcal{B}$ and x is in $D_{u(p)}$ we can decide if x enters D_p by adding x to B_p. In this case we can force x into D_p, where p is minimal. These situations will come up in the proof of Theorem 5.1.

Definition 3.10. Let $\mathcal{L}$ be a finite set. $(\mathcal{T}, \mathcal{R}, l)$ is a *pure $\mathcal{L}$-pattern* iff $(\mathcal{T}, \mathcal{R})$ is a pure pattern and l maps elements of $\mathcal{L}$ to subsets of $\mathcal{T}$ such that for all $i \in \mathcal{L}$, $l(i)$ is closed downward in $\mathcal{T}$.

If we let $\mathcal{L} = \{0, 1, 2, 3\}$ then in Figure 1 we can consider $l(i)$ as the set of nodes labeled by U_i. All such nodes are minimal so $l(i)$ is downward closed. In the patterns that we use in this paper, sets $l(i)$ will always be sets of minimal nodes and hence downwardly closed. In our other examples (see Section 3.6), $l(i)$ will not be a singleton.

Informally, if a ball x is in D_p and $p \in l(i)$ then x should also be in U_i. The whole idea of a pattern is set up to consider how balls might enter one of the U_i's.

Definition 3.11. Let $\vec{U} = \langle U_i : i \in \mathcal{L} \rangle$ be an $\mathcal{L}$-interpretation and $\vec{D} = \langle D_p : p \in \mathcal{T} \rangle$ be an interpretation of the pure pattern $(\mathcal{T}, \mathcal{R})$ and $\mathcal{P} = (\mathcal{T}, \mathcal{R}, l)$ be a pure $\mathcal{L}$-pattern. Then $\vec{D}$ *is an interpretation of $\mathcal{P}$ over* $\vec{U}$ iff for all $i \in \mathcal{L}$, all $p \in l(i)$, $D_p - A \subseteq U_i$ and for all $i \in \mathcal{L}$, for all $q \in \mathcal{T}$, $(D_q \cap U_i) - A = \bigcup \{D_p : p \leq q \ \& \ p \in l(i)\} - A$.

Assuming that $\vec{D}$ is an interpretation of $\mathcal{P}$ over $\vec{U}$ then if a ball is in D_p and $p \in l(i)$ then that ball must enter U_i or A, and if a ball is in D_q and U_i then that ball must move into A or some D_p, where $p \leq q$ and

$p \in l(i)$. What this means for the example in the last subsection is if $\vec{D}$ is an interpretation of $\mathcal{P}_0$ over $\vec{U}$ then $U_0 - A = D_{b_1} - A$, $U_1 - A = D_{r_1} - A$, etc. Again we observe that whether $\vec{D}$ is an interpretation of $\mathcal{P}$ over $\vec{U}$ is equivalent to a finite conjunction of Π^0_2 conditions, in the parameters $\vec{D}$, $\vec{U}$ and A, like those in Equation 3.1.

The U_i's are not played by RED or BLUE but by us. But RED and BLUE, as in Remark 3.9, can force balls into minimal nodes and, as such, if $\vec{D}$ is an interpretation of $\mathcal{P}$ over $\vec{U}$, then into some of the U_i's. Since we control the U_i's we will be able to control whether $\vec{D}$ is an interpretation of $\mathcal{P}$ over $\vec{U}$ or not. As we will later see this control translates into the realization and nonrealization of patterns; for more see Section 3.5.

Definition 3.12. Let $\mathcal{L}$ be a finite set. $\mathcal{P} = (\mathcal{T}, \mathcal{R}, \mathcal{B}, l)$ is a *special* $\mathcal{L}$-*pattern* iff $(\mathcal{T}, \mathcal{R}, l)$ is a pure $\mathcal{L}$-pattern and $(\mathcal{T}, \mathcal{R}, \mathcal{B})$ is a special pattern.

This is a combination of Definitions 3.4, 3.6 and 3.10. In Section 3.1, we gave an informal definition of the special $\mathcal{L}$-pattern $\mathcal{P}_o$ and in Section 3.6, we provide the formal definition of the patterns $\mathcal{P}_i$, for all i.

3.3. **The formula φ.** We use the definitions from Section 3.2 to define the following:

Definition 3.13. Fix a finite set $\mathcal{L}$, $\vec{U}$ an $\mathcal{L}$-interpretation, $\mathcal{P} = (\mathcal{T}, \mathcal{R}, \mathcal{B}, l)$ a special $\mathcal{L}$-pattern and computably enumerable sets A and C. Let $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ be the property: there is a $\vec{D}$ such that $\vec{D}$ is a $(\mathcal{T}, \mathcal{R})$-interpretation, $\vec{D}$ is a $\vec{B}$-interpretation of $\mathcal{P}$ inside of C and $\vec{D}$ is an interpretation of $\mathcal{P}$ over $\vec{U}$ (this is a combination of Definitions 3.5, 3.8 and 3.11). Let $\varphi_{\mathcal{P}}(\vec{U}, \vec{B}, C)$ be $\varphi_{\mathcal{P}}(\emptyset, \vec{U}, \vec{B}, C)$.

Remark 3.14. As we have shown with Definitions 3.5, 3.8 and 3.11, the statement "$\vec{D}$ is a $(\mathcal{T}, \mathcal{R})$-interpretation, $\vec{D}$ is a $\vec{B}$-interpretation of $\mathcal{P}$ inside of C and $\vec{D}$ is an interpretation of $\mathcal{P}$ over $\vec{U}$" (this statement is a conjunction of those three statements in Definitions 3.5, 3.8 and 3.11, respectively) is equivalent to a finite conjunction of Π^0_2-conditions (all like the one displayed in Equation 3.1) in the parameters $\vec{D}$, $\vec{B}$, $\vec{U}$, C and A and hence this statement is Π^0_2. Let $\Lambda(A, \vec{U}, \vec{B}, C, \vec{D})$ be this Π^0_2 statement. Furthermore, since all of the Π^0_2 conditions (like the one displayed in Equation 3.1) are of the form

$$(\forall x)(\forall s)(\exists t)[\Lambda'(x, s, t, A, \vec{U}, \vec{B}, C, \vec{D})],$$

where Λ' is quantifier-free, Λ can be written in that same form.

Remark 3.15. $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ is Σ^0_3; it is "$(\exists \vec{D})(\Lambda(A, \vec{U}, \vec{B}, C, \vec{D})$". Later we speak of $\vec{D}$ realizing or witnessing that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds which means that $\Lambda(A, \vec{U}, \vec{B}, C, \vec{D})$ holds.

$\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ can be written as a statement in the language $L = \{\subseteq\}$ (since the same is true for each one of the Π^0_2 conditions in Definitions 3.5, 3.8 and 3.11). Also whether $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds is a property of $\mathcal{L}(A)$; we only worry about balls on the outside of A. That is, if $\mathcal{L}(A)$ is isomorphic to $\mathcal{L}(F)$ via Φ then $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds iff $\varphi_{\mathcal{P}}(F, \Phi(\vec{U}), \Phi(\vec{B}), \Phi(C))$ holds. (Currently this is not the case for $\mathcal{L}^*(A)$. We could, if desired, replace the $=$'s in the above definitions with $=^*$'s to get a property for $\mathcal{L}^*(A)$.)

Notice if $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds and $F \subseteq C$ then $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, F)$ holds, and if $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, F)$ holds, $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C \cup F)$ holds.

Since we now need a computably-theoretic definition, we delay until Section 3.5 a discussion of how a pattern $\mathcal{P}$ is realized. For more on φ we direct the reader to Section 3.5 and to the proof of Theorem 5.1.

3.4. **Computable modulo A.** We need the following computably-theoretic definition:

Definition 3.16. X is *computable modulo* A iff there is a Y such that $X \cap Y \subseteq A$ and $X \cup Y \cup A = \omega$ iff there is a computable R such that $R \subseteq X$ and $X \subseteq R \cup A$. (The first "iff" is the definition. The second "iff" follows so easily that we will use it as the definition when needed.)

To say X is computable modulo A is Σ^0_3. In addition, "X is computable modulo A" can be written as a statement in the language $L = \{\subseteq\}$. X is computable modulo A is a property of $\mathcal{L}(A)$. That is, if $\mathcal{L}(A)$ is isomorphic to $\mathcal{L}(F)$ via an isomorphism Φ then X is computable modulo A iff $\Phi(X)$ is computable modulo F. Here the same holds for $\mathcal{L}^*(A)$.

If A is computable, then X is computable modulo A iff X is computable. X is not computable modulo A iff $A \cup (\omega - X)$ is not a computably enumerable set. In this case by using the Owings Splitting Theorem we can split X into two sets which are not computable modulo A. If S is a split of W and S is not computable modulo A then W is not computable modulo A. If $W \searrow A$ is finite then W is computable modulo A.

Recall that S is a split of W if there is a computably enumerable set Y such that $X \cap Y = \emptyset$ and $X \cup Y = W$; this is sometimes written $X \sqcup Y = W$.

3.5. **Realizing patterns.** The main use of φ is in the following definition:

Definition 3.17. Fix $\mathcal{L}$, $\vec{U}$ an $\mathcal{L}$-interpretation and computably enumerable sets A and W. We say $\vec{U}$, A and W *realize* $\mathcal{P}$ if for all $\mathcal{B}$-interpretations, $\vec{B}$, there is a split C of W such that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds and C is not computable modulo A.

Now $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ is Σ^0_3, being not computable modulo A is Π^0_3 (see Section 3.4) and whether S is a split of W is Σ^0_3. Hence on the surface it appears that whether $\vec{U}$, A and W realize $\mathcal{P}$ is Π^0_5. Corollary 4.2 allows us to improve this to Σ^0_4. Corollary 8.3 further improves this count to Σ^A_3.

Realizing a pattern can be considered as a game. That is, for all possible BLUE strategies, given by $\vec{B}$, and fighting the RED moves, given by $\mathcal{R}$, there is a split C of W such that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ and C is not computable modulo A. Normally when we try to realize a pattern we get to build the $\vec{U}$ and the $\vec{D}$ to witness $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ in response to A, C and $\vec{B}$. The split C is built in response to $\vec{B}$ using A and W and standard computably-theoretic arguments, like the Owings Splitting Theorem. More about realizing patterns can be found in Sections 4 and 5.

We will also be interested in when $\vec{U}$, A and W do not realize $\mathcal{P}$. So it is worth decoding the negation of realizing. $\vec{U}$, A and W do not realize $\mathcal{P}$ iff there is $\vec{B}$ such that for all splits S of W, S is computable modulo A or for all $\vec{D}$, $\vec{D}$ does not witness $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, \tilde{S})$. Furthermore we can ask that each component of $\vec{B}$ is a subset of W. We call such a $\vec{B}$ a witness to the nonrealization of $\mathcal{P}$. More about the nonrealization of patterns can be found in the proof of Theorem 5.1.

Our main goal (see Theorem 5.1) is to build A and $\vec{U}$ which realize certain patterns without realizing other patterns. This creates a conflict since $\vec{U}$ is shared between all patterns. We, not RED or BLUE, control the enumeration of the U_i's. This control is enough to allow us to balance needs of realization and nonrealization and do both. For more see Sections 5.1.5 and 5.1.6.

The phrase "C is not computable modulo A" is there to ensure that there are infinitely many balls which are in C but not in A. For each possible $\vec{B}$ we want infinitely many balls moving through the pattern $\mathcal{P}$ and witnessing that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, C)$ holds. As we mentioned earlier in other applications of the our patterns, this phrase can and will be changed.

The following fact will prove useful in Section 6.5.2: If, for all l, $V_l \cap W = U_l \cap W$ then $\vec{U}$, A and W realize $\mathcal{P}$ iff $\vec{V}$, A and W do.

3.6. **Our special $\mathcal{L}$-patterns.** Fix $\mathcal{L} = \{0, 1, 2, 3\}$. To be complete we will first repeat the formal definition of the special $\mathcal{L}$-pattern $\mathcal{P}_0$ (this was done informally in Section 3.1).

Again we will refer to Figure 1. This figure describes the pattern $\mathcal{P}_0$. Ignoring the box and the labels and considering the solid circles as nodes, this figure describes a tree. We will call this tree $\mathcal{T}$. The node labeled b_0 is the top or root. We will use the b_i's and r_i's as labels for the nodes in $\mathcal{T}$. The nodes labeled by $\mathcal{R}$ are in the set $\mathcal{R}$. So $\mathcal{R} = \{b_2\}$. Hence $(\mathcal{T}, \mathcal{R})$ is a

pure pattern. Let $l(i)$ be the sets of nodes with the label U_i. Let $\mathcal{B}$ be those nodes labeled with b_i, for some i. $(\mathcal{T}, \mathcal{R}, \mathcal{B})$ is a maximal special pattern and $(\mathcal{T}, \mathcal{R}, \mathcal{B}, l)$ is a special $\mathcal{L}$-pattern.

For special $\mathcal{L}$-pattern $\mathcal{P}_n$ we just repeat the part of the pattern in the box in $\mathcal{P}_0$ $n+1$ times. We will call this part of the pattern a *block*. So balls can flow from b_4 to another copy of b_2.

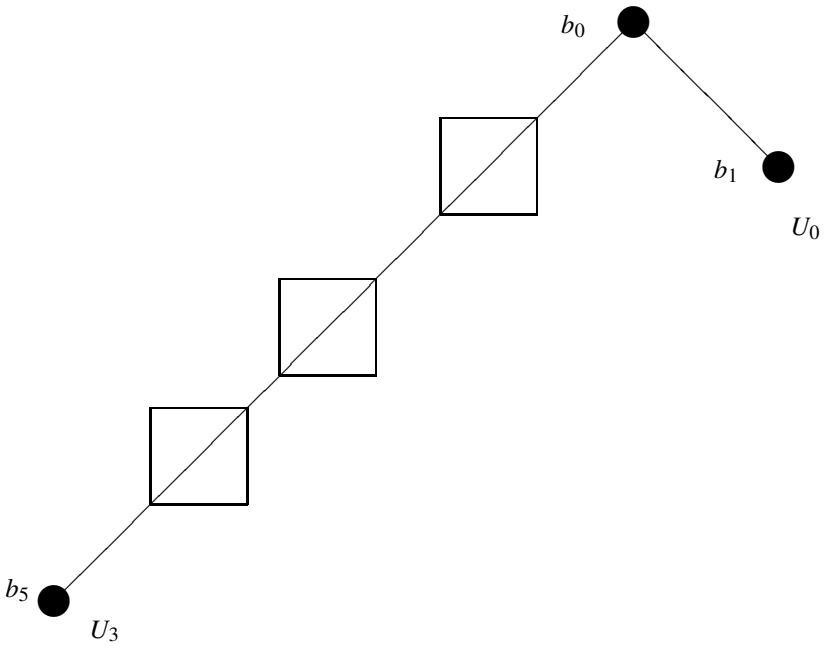

FIGURE 2. The special $\mathcal{L}$-pattern $\mathcal{P}_3$ (The boxes refer to the block in Figure 1).

Figure 2 shows a picture of $\mathcal{P}_3$ where the boxes refer to the copies of the block in Figure 1. The labeling is done in exactly the same way except in the ith copy of the block, the nodes b_2, r_1, r_2, b_3, b_4 will be labeled $b_2^i, r_1^i, r_2^i, b_3^i, b_4^i$, respectfully. We will refer to $b_2^0, r_1^0, r_2^0, b_3^0, b_4^0$ as b_2, r_1, r_2, b_3, b_4, respectfully. In $\mathcal{P}_j$, if $l < j$, the nodes b_4^l and b_2^{l+1} are the same node. So $(\mathcal{T}_i, \mathcal{R}_i, \mathcal{B}_i)$ is a maximal special pattern and $(\mathcal{T}_i, \mathcal{R}_i, \mathcal{B}_i, l_i)$ is a special $\mathcal{L}$-pattern.

Note the repeated part of the patterns has both a RED forced move and some possible BLUE moves. We should point out that there is nothing "unique" about our presented patterns. What is important about them is that using them we can prove Theorem 5.1.

4. A universal $\mathcal{B}$-interpretation

Fix $\mathcal{L}$, $\mathcal{P}$, $\vec{U}$ an $\mathcal{L}$-interpretation and two computably enumerable sets A and W. This next theorem says that there is a universal $\mathcal{B}$-interpretation. That is, there is a $\vec{B}_e$ (where $\vec{U}$, $\mathcal{P}$, A and W are coded or indexed somehow by e) such that there is a split S of W_e where S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$ iff $\vec{U}$, A and W realize $\mathcal{P}$.

Theorem 4.1. *Fix $\mathcal{L}$, $\vec{U}$ an $\mathcal{L}$-interpretation, $\mathcal{P} = (\mathcal{T}, \mathcal{R}, \mathcal{B}, l)$ a special $\mathcal{L}$-pattern and two computably enumerable sets A and W. Uniformly in $\vec{U}$, $\mathcal{P}$, A and W (think of all of these items as indexed by e), there is a $\mathcal{B}$-interpretation, $\vec{B}_e$ such that the following are equivalent:*

(1) *There is a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$.*
(2) *$\vec{U}$, A and W realize $\mathcal{P}$.*

The proof will follow shortly. If S is a split of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$, where e codes $\vec{U}$, $\mathcal{P}$, A and W, then we say S *witnesses* that $\vec{U}$, A and W *realize* $\mathcal{P}$. In addition, since $\vec{B}_e$ can be found uniformly we get the following corollary which will prove very useful. It removes one quantifier from the definition of realizing.

Corollary 4.2. *Whether $\vec{U}$, A and W realize $\mathcal{P}$ is Σ^0_4.*

Proof of Corollary 4.2. By Theorem 4.1, $\vec{U}$, A and W realize $\mathcal{P}$ iff there is a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$. Being not computable modulo A is Π^0_3 (see Section 3.4) and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$ is Σ^0_3 (see Section 3.3). □

Corollary 4.2 removes one quantifier from the definition of realizing (see Section 3.5). Corollary 8.3 further improves this count to Σ^A_3.

As we discussed above each $\vec{B}$ corresponds to a play by BLUE (see Definition 3.8). Definition 3.5 can be read for all BLUE plays something happens (there is a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$). Corollary 4.2 says that to realize a pattern it is enough that something happens for the one BLUE play $\vec{B}_e$. Because of this and the way $\vec{B}_e$ is constructed, we will consider $\vec{B}_e$ as the universal $\mathcal{B}$-interpretation, the universal BLUE play.

4.1. **Proof of Theorem 4.1.** By Definition 3.5, $\vec{U}$, A and W realize $\mathcal{P}$ iff for all $\mathcal{B}$-interpretations, $\vec{B}$, there is a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, S)$. Hence (2) implies (1) is clear. The rest of the proof focuses on the other direction. We can safely assume, by the Recursion Theorem, that we have an index for $\vec{B}_e$.

4.1.1. *A first approximation.* The following is a good first approximation (it is only a module and must be made effective later). Assume we know there is a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$.

Now we can use the Owings Splitting Theorem [see 16, Theorem X.2.5]. By the Owings Splitting Theorem, we can effectively split S into S_j, for $j \in \omega$, such that S_j is not computable modulo A. Now on S_j, we can replicate the jth $\mathcal{B}$-interpretation, $\vec{B}_j$; that is, since we are building $\vec{B}_e$, we can let $B_{e,p} \cap S_j = B_{j,p} \cap S_j$, for all $p \in \mathcal{T}$.

So for the jth $\mathcal{B}$-interpretation, $\vec{B}_j$, we have S_j is not computable modulo A and if $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$ holds, then $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S_j)$ holds and hence $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_j, S_j)$ holds (since $B_{e,p} \cap S_j = B_{j,p} \cap S_j$, for all $p \in \mathcal{T}$).

The above argument is not effective; S is not given to us effectively and therefore the index for $\vec{B}_e$ cannot be found effectively. Hence it is not correct; to use the Recursion Theorem the argument must be effective. But it is a good first approximation.

4.1.2. *The tree construction.* We will use the tree method to make the argument effective. Again, we can safely assume, by the recursion theory, that we have an index for $\vec{B}_e$.

We can construct a sequence of sets $(\widehat{S}_i, \tilde{S}_i, \vec{D}_i)$. Now asking if $\tilde{S}_i$ witnesses $\widehat{S}_i$ is a split of W and $\vec{D}_i$ witnesses that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, \widehat{S}_i)$ is Π^0_2. We need to code this information on a Π^0_2 tree.

We define the true path as follows: Let $\alpha \in 2^{<\omega}$ be of length i such that $\alpha \subset f$. We will let $\alpha\hat{}0 \subset f$ iff $\tilde{S}_i$ witnesses $\widehat{S}_i$ is a split of W and $\vec{D}_i$ witnesses that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, \widehat{S}_i)$.

Now we need to define the approximation to the true path, f_s. Let's assume that $\tilde{S}_i$ witnesses $\widehat{S}_i$ is a split of W and $\vec{D}_i$ witnesses $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, \widehat{S}_i)$ iff $(\forall x)(\exists s)[\Theta(x, s)]$, where Θ is computable (Θ can be found uniformly in the given parameters). Let $l_\alpha(s)$ be the greatest x such that for all $y < x$ there is an s_y such that $\Theta(y, s_y)$. Assume that $\alpha \subseteq f_s$ and $|\alpha| < s$. Then let $\alpha\hat{}0 \subseteq f_s$ iff $l_\alpha(s) > l_\alpha(t)$, where t is the last stage when $\alpha\hat{}0 \subseteq f_t$ (if such a stage does not exist let $t = 0$). It is straightforward to show that $f = \liminf_s f_s$.

We need to discuss the needed action at $\beta = \alpha\hat{}0$. At β we will effectively build a subset S_β of $\widehat{S}_i \cap W$. To build S_β, β will use a number d_β and a function $u_\beta(s)$. At the first stage when $\beta \subseteq f_s$ or the first such stage after being initialized we will set d_β to be large (just some number not yet occurring in the construction), $u_\beta(s)$ to be even larger and discontinue this stage of the construction. If $\beta \subseteq f_s$, we will set $u_\beta(s)$ to be large; otherwise

the value of $u_\beta(s)$ is $u_\beta(s-1)$. Like always, the number d_β and the function $u_\beta(s)$ will be initialized if $f_t <_L \beta$, for some later stage t.

Consider a ball x. If there is a stage s such that x is in the interval $[d_\beta, u_\beta(s)]$, x is not in any $S_{\gamma,s}$, x is in $\widehat{S}_{i,s} \cap W_s$ and, for all $\gamma\hat{}0 \subset \beta$, $x \in \tilde{S}_j$, where $j = |\gamma|$, then we will add x to S_β at stage s.

On S_β we will use the module from Section 4.1.1. Using the Owings Splitting Theorem, we will effectively split S_β into $S_{\beta,j}$, for $j \in \omega$. On $S_{\beta,j}$, we can replicate the jth $\mathcal{B}$-interpretation, $\vec{B}_j$; that is, we can let $B_{e,p} \cap S_{\beta,j} = B_{j,p} \cap S_{\beta,j}$, for all $p \in \mathcal{T}$.

4.1.3. *The verification.* The construction of S_β is uniformly effective in β. The S_β's are constructed to be pairwise disjoint. Therefore the construction of $\vec{B}_e$ is uniform in $\vec{U}$, $\mathcal{P}$, A and W and hence it is legal to use the Recursion Theorem as we did.

Assume there is a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, S)$ (condition (1) from the theorem). Let $(\widehat{S}_i, \tilde{S}_i, \vec{D}_i)$ be the least triple such that $\widehat{S}_i$ is not computable modulo A, $\tilde{S}_i$ witnesses $\widehat{S}_i$ is a split of W and $\vec{D}_i$ witnesses $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_e, \widehat{S}_i)$.

Let $\alpha \subset \beta \subset f$ such that $|\alpha| = i$ and $\beta = \alpha\hat{}0$. If $\gamma\hat{}0 \subset \beta \subset f$, then $\widehat{S}_j$ is computable modulo A split of W, where $j = |\gamma|$. So $\bigcup\{\widehat{S}_{|\gamma|} : \gamma\hat{}0 \subset \beta\}$ is computable modulo A split of W. Let

$$\widehat{S} = \widehat{S}_i \cap \bigcap\{\tilde{S}_{|\gamma|} : \gamma\hat{}0 \subset \beta\} = \widehat{S}_i - \bigcup\{\widehat{S}_{|\gamma|} : \gamma\hat{}0 \subset \beta\}.$$

Therefore $\widehat{S}$ is a noncomputable modulo A split of W.

We will define a computable subset, R, of W as follows: First assume that β is never initialized after stage t. Given $x \geq t$ wait for a stage $s \geq t$ such that x is the interval $[d_\beta, u_\beta(s)]$. If x is in any S_γ (a subset of W) at stage s then put x into R; otherwise x is not in R.

By the construction of S_β, $S_\beta =^* \widehat{S} \cap \overline{R}$. Since $\widehat{S}$ is a split of W and R is a subset of W, $\widehat{S} \cap R$ is computable. If Y witnesses that S_β is computable modulo A then $Y - (\widehat{S} \cap R)$ witnesses that $\widehat{S}$ is computable modulo A. Hence S_β is a split of W and S_β is not computable modulo A.

Then, by the Owings Splitting Theorem, $S_{\beta,j}$ is also a noncomputable modulo A split of W. Therefore $S_{\beta,i}$ is a noncomputable modulo A split of W and $\vec{D}_i$ witnesses $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_j, S_{\beta,j})$. Hence, by Definition 3.5, $\vec{U}$, A and W realize $\mathcal{P}$ (condition (2) from Theorem 4.1) □

5. Realizing and not realizing patterns

The following theorem is the key idea behind our patterns. The patterns were chosen such that the following theorem holds. Informally the theorem says that we can realize one pattern without realizing any others.

Theorem 5.1. *Effectively in i, W, a $\mathcal{B}$-interpretation $\vec{B}_S$ and S a subset of W, there exists $\vec{U}$ and $\vec{D}$ such that $\vec{D}$ witnesses that $\varphi_{\mathcal{P}_i}(\emptyset, \vec{U}, \vec{B}_S, S)$ holds (and all the components of $\vec{U}$ and $\vec{D}$ are contained in S). Furthermore, for all $j \neq i$, and for all $C \subseteq W$, $\vec{U}$, $A = \emptyset$, and C do not realize $\mathcal{P}_j$. Moreover, the witness, $\vec{B}_{j,C}$, which shows that $\vec{U}$, $A = \emptyset$, and C do not realize $\mathcal{P}_j$ can be found effectively from j and C (and such that all components are contained in C).*

Recall, from Theorem 4.1, for $\vec{U}$, $A = \emptyset$ and W to realize $\mathcal{P}_i$ there must be a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}_i}(\emptyset, \vec{U}, \vec{B}_e, S)$, where e codes $\vec{U}$, $\mathcal{P}_i$, A and W. So if we apply the above theorem to a split S of W which is not computable modulo A and to $\vec{B}_e$, where e codes $\vec{U}$, $\mathcal{P}_i$, A and W and is arrived at by an application of the Recursion Theorem (the above theorem builds $\vec{U}$ which must be included in the code) then we have that $\vec{U}$, $A = \emptyset$ and W realize $\mathcal{P}_i$ without realizing any other $\mathcal{P}_j$. For more see Corollary 6.5.

5.1. **Proof of Theorem 5.1.**

5.1.1. *The requirement $\mathcal{Q}$.* Our highest priority requirement is to meet the following requirement:

($\mathcal{Q}$) build $\vec{U}$ and $\vec{D}$ such that $\vec{D}$ witnesses that $\varphi_{\mathcal{P}_i}(\emptyset, \vec{U}, \vec{B}_S, S)$ holds.

Since we are building $\vec{U}$ and $\vec{D}$, we can just use the strategy "playing with $\vec{B}_S$" which is outlined below.

We are given A and S and are forced to construct $U_i \cap S$, for $i \in \mathcal{L}$, and $\vec{D}$ in response to $\vec{B}_S$ such that $\varphi_{\mathcal{P}_i}(\emptyset, \vec{U}, \vec{B}_S, S)$ holds. This is not a difficult process since we are building $\vec{U}$ and $\vec{D}$. Assume that x is a ball and that there is a node $p \in \mathcal{T}$. If p is in $\mathcal{B}_i$ (hence $u(p) \notin \mathcal{R}_i$) and $x \in D_{u(p)} \cap S$ (just S if $p = b_0$) then x enters D_p iff x enters B_p. This is a BLUE choice. If p is in $\mathcal{R}$ and x is in D_p then x must enter one of the D_q where $q \in d(p)$. It is a RED choice that x must leave D_p but RED does not choose the q. This is our choice and we will use it to our benefit in this proof. If p is in some $l(i)$ and x is in D_p then we must add x to U_i. We call this strategy *playing with $\vec{B}_S$*.

We have some choice under a ball's movement on the pattern $\mathcal{P}_i$. If a ball x gets into $D_{b_2^l}$ then x must enter $D_{r_1^l}$ or $D_{r_2^l}$. If we move x rightward into $D_{r_2^l}$ then x must enter U_2. But if we always move x leftward into $D_{r_2^l}$ then we can always keep x out of U_2.

Note that we have no need to add a ball which is not in S into any component of $\vec{U}$ or $\vec{D}$.

5.1.2. *The negative requirements.* To ensure we do not realize any other pattern it is enough to meet the following negative requirements:

$(\mathcal{N}_{j,C})$ build a $\mathcal{B}$-interpretation, $\vec{B}_{j,C}$, such that for all $\tilde{S}$ and $\vec{X}$,

$(\mathcal{N}_{j,C,\tilde{S},\vec{X}})$ $\vec{X}$ does not witness $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$ or, $\tilde{S}$ is computable,

for all $C \subseteq W$, for all $j \neq i$ and for all splits $\tilde{S}$ of C. Before we proceed we will to fix need some priority ordering on the requirements $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$.

Consider the eth such requirement $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. Between now and Section 5.1.11, we will just focus on this one requirement. Furthermore we will focus on only the first clause. The second clause will only come into play in Section 5.1.12. By meeting $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we mean showing the first clause holds. To make life notationally easier we will drop the indexes j and C from $\vec{B}_{j,C}$ up until Section 5.1.11.

5.1.3. *Meeting the* e*th requirement* $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. Here we are given $\vec{X}$ and get to build $\vec{B}$ and $\vec{U}$ such that $\vec{X}$ does not witness $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$.

By Remarks 3.14 and 3.15, $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$ is $(\exists \vec{X})\Lambda(\emptyset, \vec{U}, \vec{B}, \tilde{S}, \vec{X})$, where Λ is Π^0_2 and of the form $(\forall x)(\forall s)(\exists t)[\Lambda'(x, s, t, \emptyset, \vec{U}, \vec{B}, \tilde{S}, \vec{X})]$, where Λ' is quantifier-free. So we are looking for a ball x that shows that $\vec{X}$ does not witness $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$: a ball x such that it is not the case "(for all stages s)(there is a stage t)$[\Lambda'(x, s, t, \emptyset, \vec{U}, \vec{B}, \tilde{S}, \vec{X})]$".

With x there is implicitly a stage s where things start to fail. This stage might change during the construction but if x is the witness we are looking for at some point this stage will stop changing. Hence it is, at worst, Π^0_1 to tell if x and its corresponding stage s show that $\vec{X}$ does not witness $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$.

The first issue is getting balls for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. We process the balls x as they enter C until we have an apparent win as suggested above. The fact that $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ gets assigned balls when they first enter C (not $\tilde{S}$) will be very useful in showing the negative requirements interact without any difficulties; see Section 5.1.11.

The processing will be done by priority. So if two requirements are interested in x, the highest priority one gets x. If a ball x (and its corresponding stage s) for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ witnesses a win, we will suspend the collection of balls for this requirement. If that apparent win disappears we will continue with the collection. All the balls used for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ will be labeled with e, the priority of $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$.

If $\tilde{S}$ is a noncomputable split of C then the requirement $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ will be able to get enough balls as they just enter C. However we will have to do a little computability-theoretic work to ensure this; see Section 5.1.12. Lets assume that $\tilde{S}$ is a noncomputable split of C and we can get enough balls as they enter C for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ to work with.

We get to build the sets $\vec{B}$ and $\vec{U}$ in order to show $\vec{X}$ does not witness $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$. Lets first see how we might find a ball x which shows this without making any enumerations into any of these sets. If $\vec{X}$ witnesses $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$, then $\vec{X}$ is a $(\mathcal{T}_j, \mathcal{R}_j)$-interpretation (Definition 3.5) and $\vec{X}$ is a $\vec{B}$-interpretation of $\mathcal{P}_j$ inside of C (Definition 3.8).

These definitions say something about the sets X_p, for $p \in \mathcal{T}$, and hence the movement of the balls in the pattern $\mathcal{P}_j$. For example, if x is X_p and $q \geq p$ then x must be in X_q and if x is $X_{u(p)} \cap B_p$ then x must also be in X_p. So we might find an x and a stage s such that $x \in X_{u(p),s} \cap B_{p,s}$ but for all stages t, $x \notin X_{p,s}$. If this happens then x witnesses that $\vec{X}$ is not a $\vec{B}$-interpretation of $\mathcal{P}$ inside of C. Notice that we must hang onto x indefinitely in this case.

There are many other similar ways that x can witness that $\vec{X}$ is not a $(\mathcal{T}, \mathcal{R})$-interpretation and is not a $\vec{B}$-interpretation of $\mathcal{P}$ inside of C. Each time we find such a witness we do not need to make any enumeration into $\vec{U}$ or $\vec{B}$. So this is the easy way to find the needed witness. Let's assume that our witness to this requirement does not take this form. We will assume (at least on the finitely many balls we process for this requirement) that $\vec{X}$ is a $(\mathcal{T}_j, \mathcal{R}_j)$-interpretation and is a $\vec{B}$-interpretation of $\mathcal{P}_j$ inside of C.

Under this assumption and the fact that we are building $\vec{B}$ we can control where balls are in the pattern $\mathcal{P}_j$. If a ball x under consideration is in $X_{u(p)}$ then we can move x into X_p by adding x to B_p. Since $\mathcal{P}_j$ is maximal (see Definition 3.7), we can force any ball x into a minimal node of $\mathcal{T}_i$ (see Remark 3.9). Now in $\mathcal{P}_j$, all the minimal nodes of $\mathcal{P}_j$ are in some $l(k)$. If we do not allow x to enter U_k then $\vec{X}$ is not an interpretation of $\mathcal{P}_j$ over $\vec{U}$ and x is our desired witness.

5.1.4. *Overview of the strategy used to meet* $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. Let's review how we can meet the requirement $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ in isolation. First we will assume that $\vec{X}$ is a $(\mathcal{T}_j, \mathcal{R}_j)$-interpretation and is a $\vec{B}$-interpretation of $\mathcal{P}_j$ inside of C. Otherwise no work is needed to meet this requirement. Then using this assumption and enumerating balls in $\tilde{S}$ into components of $\vec{B}$ we can force a ball into some minimal node p. Now $p \in l(k)$, for some k. Now to meet this requirement we do not allow x to enter U_k.

So in short the strategy for this requirement is to move balls downward in $\mathcal{P}_j$ and restrain them from entering all U_k until we meet this requirement. While we move the balls downward we are presented with choices; at some nodes, since we can play $\vec{B}$, we can move balls left or right. This choice will prove to be important; see Section 5.1.6.

All it takes is one ball in a minimal node for us to meet this requirement. Hence the requirement is finitary. If we get one ball in $\tilde{S}$ which we can hang onto indefinitely then we will be able to meet this requirement. Of course, more balls make the action for this requirement go faster.

5.1.5. *The roles of the players, revisited.* It might be a good time to revisit the roles of the three players, RED, BLUE and the player we are identifying with, the unnamed third player. It is the goal of this third player to show that $\vec{U}, A, S$ realize pattern $\mathcal{P}_i$ without realizing any other patterns. Actually this is an overstatement; the goal of this third player at this point is to prove Theorem 5.1. But as we will later see, Theorem 5.1 is almost enough to achieve our loftier goals (see Section 6). To further simplify things we will not worry about A or S at this point.

For $\vec{U}, A, S$ to realize $\mathcal{P}_i$ this player wants to meet requirement $\mathcal{Q}$ (more or less). To this end our third player builds $\vec{U}$ and $\vec{D}$ against BLUE's universal $\mathcal{B}$-interpretation, $\vec{B}_S$. Since $\mathcal{P}_i$ is maximal (see Definition 3.7), every move through the pattern is determined by RED or BLUE. The only choice available to this third player is, if $p \in \mathcal{R}$ and a ball x is in D_p, which of the D_q, where $q \in d(p)$, x will enter, making sure RED's promise to empty that state is realized. So movement through this pattern determines when balls enter $\vec{U}$.

On the other hand, this third player also does not want $\vec{U}, A, S$ to realize $\mathcal{P}_j$, for $j \neq i$. To do this our third player wants to meet the requirements $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. Here the movement through $\mathcal{P}_j$ is determined by $\vec{X}$, sets out of this third player's control. But this third player gets to build $\vec{U}$ and it controls BLUE's moves through the pattern via building $\vec{B}_{j,C}$. Basically the strategy used here is to move the balls downward in $\mathcal{P}_j$ via enumeration into $\vec{B}_{j,C}$ (see the above section and Remark 3.9). Then to meet the requirement it is enough to restrain the ball(s) from entering the correct $\vec{U}$.

5.1.6. *The interaction between $\mathcal{Q}$ and $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$.* So the only conflict between strategies is with the sets $\vec{U}$. For $\mathcal{Q}$ we might have a ball in some node p in $\mathcal{T}_i$ where $p \in l_i(k)$ and this same ball might be in a similar node in $\mathcal{P}_j$. So to meet $\mathcal{Q}$ we must add x to U_k and to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we would like to restrain x from entering U_k. This is a conflict and we will ensure that this situation does not occur.

Remark 5.2. Assume that $S = \emptyset$. Then this situation does not occur; there are no balls moving in the pattern $\mathcal{P}_i$. In this case we have a very easy way to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$; simply force every ball in C downward and leftward by letting $U_0 = U_1 = U_2 = U_3 = \emptyset$ and $B_{b_0} = B_{b_2^l} = B_{b_4^j} = B_{b_5} = C$, for all $l \leq j$.

But in general we cannot stop a ball from moving through the pattern $\mathcal{P}_i$. Every ball in S must flow through the pattern $\mathcal{P}_i$. So if we are lucky enough to find a ball in $\tilde{S}$ which is not in S we will not have a conflict. But S is a computably enumerable set so a ball which we think at some stage is not in S might actually belong to S at some later stage. Hence we must act as if every ball moving in the pattern $\mathcal{P}_j$ is also moving through $\mathcal{P}_i$.

The trick is to use the choice we have available in the movement of balls through the patterns $\mathcal{P}_i$ and $\mathcal{P}_j$. In $\mathcal{P}_i$ if x is in $D_{b_2^l}$ (b_2^l is in $\mathcal{R}_i$) then we can either choose to add x to $D_{r_2^l}$ (move downward and leftward), choose to add x to $D_{r_2^l}$ (move downward and rightward). This is the only possible choice that is allowed under the strategy being used to meet $\mathcal{Q}$; see Section 5.1.2. All movement on $\mathcal{P}_j$ is controlled by the external sets $\vec{X}$. But if $x \in X_{u(p)}$ (and in $\mathcal{P}_j$) we can choose to move x into X_p by adding x to B_p. In both cases the default action is to move downward and leftward unless we have a win for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ that does not conflict with $\mathcal{Q}$.

5.1.7. *An analysis by cases.* The following in the next three sections (Sections 5.1.8, 5.1.9 and 5.1.10) is an analysis of all the possible cases. Interlaced with the analysis is an argument that we have actually covered all the possible cases; this argument concludes in Section 5.1.10. The end result, of course, is that it is possible to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ without interfering with the strategy for $\mathcal{Q}$. During these sections it would be very helpful to refer back to Figure 1. In addition it will be very important to understand the notation from Section 3.6. This is the only place where we need the actual details of our patterns.

5.1.8. *Movement on $\mathcal{P}_i$.* First lets focus on the choices in $\mathcal{P}_i$. So assume that x is in $D_{b_2^l}$, for some $l \leq i$, and we must decide to move x left or right. If we move x right then x will enter U_1. If we always move x leftward when given the choice then x will never enter U_1. The choice we make depends on where x is on $\mathcal{P}_j$. Let p be the least node in $\mathcal{P}_j$ such that $x \in X_p$ (p is the node where x is in pattern $\mathcal{P}_j$).

If $p = b_1$ (a minimal node), b_5 (the last node in $\mathcal{T}_j$), b_4^j (the second to last node in $\mathcal{P}_j$) or $b_3^{l'}$ (a minimal node), for some l', then to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ it is enough to restrict x from entering any of U_0, U_2 or U_3. In this case, we

can move x to the right in $\mathcal{P}_i$ and into U_1 for a win. If p is $r_1^{l'}$ (a minimal node) then to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we must keep x out of U_1. In this case, we always agree to move x leftward in $\mathcal{P}_i$ when we have a choice and hence keep x out of U_1.

Assume that $p = r_2^{l'}$ ($p = b_0$). Then we can move x right in $\mathcal{P}_j$ by adding x to $B_{b_3^{l'}}$ (B_{b_1}) or move left in $\mathcal{P}_j$ by adding x to $B_{b_2^{l'+1}}$, if $l' < j$, or $B_{b_4^j}$, if $l' = j$ (B_{b_2}).

If we have made a move in $\mathcal{P}_j$ (in this case the default leftward move) and $l' < j$ then x is in $B_{b_2^{l'+1}}$ and hence should at some later stage appear in $X_{b_2^{l'+1}}$. If we have made a move in $\mathcal{P}_j$ and $l' = j$ we have a win by moving x right in $\mathcal{P}_i$ since to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we like to restrain x from U_3.

If we have not yet made this move in $\mathcal{P}_j$, we have a win for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. We can move x right in both patterns. Moving right in $\mathcal{P}_i$ asks us to add x to U_1. Moving right in $\mathcal{P}_j$ asks us to restrain x from entering U_2 (U_0).

Remark 5.3. So, in conclusion, when x is in $D_{b_2^l}$ we have a winning strategy to met $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ and $\mathcal{Q}$ unless x is either in $X_{b_2^{l'}}$ (so $p = b_2^{l'}$) or $B_{b_2^{l'}}$ (and hence targeted for $X_{b_2^{l'}}$), for some $l' \leq j$.

5.1.9. *Movement on $\mathcal{P}_j$.* Now we will focus on the choice that we have in $\mathcal{P}_j$. Again the default action to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ is to move downward and leftward in $\mathcal{P}_j$. But at several places we have the choice to move left or right. Again the choice that we make will depend on where x is on the pattern $\mathcal{P}_i$. Let p be the least node in $\mathcal{P}_i$ such that $x \in D_p$.

If x is in X_{b_0} we have the choice of moving right in $\mathcal{P}_j$ into B_{b_1} or moving left in $\mathcal{P}_j$ into $B_{b_2^1}$. If we move right we wish to restrain x from entering U_0 to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. Otherwise, to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we will never need to restrain x from entering U_0. So if $p = b_1$ take the default move for a win. If $b \neq b_0, b_1$, we can move right for another win.

Remark 5.4. So if x starts moving down $\mathcal{P}_i$ before it starts down $\mathcal{P}_j$ we have a winning play for both $\mathcal{Q}$ and $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$.

If x is in $X_{r_2^l}$ we have the choice of moving right into $B_{b_3^l}$ or moving left into $B_{b_4^l}$, which equals $B_{b_2^{l+1}}$ if $l < j$. If we move right we wish to restrain x from entering U_2 to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. If we always move left when given the choice to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we will never need to restrain x from entering U_2. If $p = b_1, r_1^{l'}, b_4^i$ or b_5, we move x rightward in $\mathcal{P}_j$. In this case, to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we just restrain x from entering U_2 and the strategy we are playing to meet $\mathcal{Q}$ will never ask to enumerate x into U_2. So we have a

win. If $p = b_3^{l'}$ then we always move left in $\mathcal{P}_j$ when given the choice for another win. If $p = b_2^{l'}$ then we are in the situation discussed just before Remark 5.3; we move right in both patterns. If $p = b_0$ then we move right in $\mathcal{P}_j$ and if x ever enters $D_{b_2^1}$ we move x rightward in $\mathcal{P}_i$ like in the above case.

Remark 5.5. So if x is in $X_{r_2^l}$ then we have a winning strategy unless x is in $D_{r_2^{l'}}$, for some $l' \leq i$.

5.1.10. *Putting our choices together.* Let's assume that we were unable to find a choice above that allows us to meet both $\mathcal{Q}$ and $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. We can force the ball downward and leftward in $\mathcal{P}_j$. Since we cannot make a choice which provides us with a win for both requirements, the ball x must also move downward and leftward in $\mathcal{P}_i$. The only question is what is the rate of movement through both patterns.

By Remark 5.4, the stage when x enters $D_{b_2^1}$ on $\mathcal{P}_i$ is after the stage when x enters $B_{b_2^1}$ on $\mathcal{P}_j$ and by Remark 5.3, before the stage when x enters $X_{r_2^1}$. By Remark 5.5, the stage when x enters $X_{r_2^1}$ must be after the stage when x enters $D_{r_2^1}$. Now by Remark 5.3, the stage x enters $D_{b_4^1} = D_{b_2^2}$ on $\mathcal{P}_i$ is after the stage when x enters $B_{b_4^1} = B_{b_2^2}$ on $\mathcal{P}_j$. By induction, this repeats until x enters $D_{b_4^{\hat{\imath}}}$, where $\hat{\imath} = \min\{i, j\}$, which it enters after it has entered $B_{b_4^{\hat{\imath}}}$.

If $i > j$ then $D_{b_4^{\hat{\imath}}} = D_{b_2^{\hat{\imath}+1}}$ and $B_{b_4^{\hat{\imath}}} = B_{b_4^j}$ and we have a winning move (see Remark 5.3). Otherwise $D_{b_4^{\hat{\imath}}} = D_{b_4^i}$ and $B_{b_4^{\hat{\imath}}} = B_{b_2^{\hat{\imath}+1}}$. In this case, the strategy we are using to meet $\mathcal{Q}$ will not want to add x into the sets U_0, U_1 or U_2. If x moves downward in $\mathcal{P}_i$ one more node then this strategy will want to add x to U_3. Now on $\mathcal{P}_j$, $b_2^{\hat{\imath}+1} \in \mathcal{R}_j$ so x must move downward in $\mathcal{P}_j$. If x moves into $X_{r_1^{\hat{\imath}+1}}$ (right) we have a win for both strategies. If x moves left into $X_{r_2^{\hat{\imath}+1}}$ then we can move x rightward into $B_{b_3^{\hat{\imath}+1}}$ for a win for both strategies.

5.1.11. *Interaction between all the negative requirements.* As we have seen to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ we just need at least a ball in $\tilde{S}$ but we might have to keep this ball indefinitely. That ball will have to move through the pattern $\mathcal{P}_j$ with its course being determined by $\vec{X}$ and $\vec{B}_{j,C}$. In addition this ball might have to follow the pattern $\mathcal{P}_i$. We have seen above how a ball can flow through the two patterns $\mathcal{P}_j$ and $\mathcal{P}_i$. Decisions on which way to move through the patterns get more difficult and sometimes even impossible if a ball must flow through more than two patterns.

The easiest way to avoid this difficulty is to assign each requirement $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ its own collection of balls in $\tilde{S}$. $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ will use the balls in its collection to meet itself. The goal will be for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ to have a ball in its collection indefinitely. If this is the case we can meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$.

$\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ will be assigned an interval of balls dynamically by a tree construction; see Section 5.1.12. If a ball in that interval then enters C and $\tilde{S}$, that ball will enter $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection. If $\tilde{S}$ is a noncomputable split of C this method will provide a finite number of balls in $\tilde{S}$ for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ to work with; again see Section 5.1.12.

Since the balls in $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection are determined before they enter C, the balls in $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection cannot enter $\mathcal{N}_{j,C,\tilde{S}',\vec{X}'}$'s collection, for any $\tilde{S}' \neq \tilde{S}$ and $\vec{X}' \neq \vec{X}$. Hence once a ball starts flowing through $\mathcal{P}_j$ for the requirement $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ it will not be moving through the same pattern with the same $\vec{B}_{j,C}$ for a different requirement.

However the balls in $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection might enter $\mathcal{N}_{j',C',\tilde{S}',\vec{X}'}$'s collection but only if $\mathcal{N}_{j',C',\tilde{S}',\vec{X}'}$ has higher priority than $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. Since there are only finitely many (active) higher priority requirements and each requirement only wants one ball indefinitely, this will only impact finitely many of the balls which enter $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection.

When a ball x leaves $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection it will start moving through some possibly different pattern $\mathcal{P}_{j'}$ with some possibly different $\vec{X}'$ and a definitely different $\vec{B}_{j',\vec{X}'}$. Hence we get to make, as above, all the decisions for x moving through $\mathcal{P}_{j'}$ with $\vec{X}'$ and $\vec{B}_{j',\vec{X}'}$. If x is already in some U_k then x must be already in the pattern $\mathcal{P}_i$ and hence, by Remark 5.4, we know how to meet $\mathcal{N}_{j',C',\tilde{S}',\vec{X}'}$ and $\mathcal{Q}$.

Fix j and C. Each requirement of the form $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ effectively enumerates balls from its disjoint (among requirements with the same j and C) collection of balls into components of $\vec{B}_{j,C}$. To determine $\vec{B}_{j,C}$ just look at all such collections. Hence $\vec{B}_{j,C}$ is built effectively from j and C. Furthermore, since $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection is a subset of C, each component of $\vec{B}_{j,C}$ is a subset of C.

5.1.12. *The tree construction.* So the only remaining question at this point is if $\tilde{S}$ is a noncomputable split of C then how do we get a ball indefinitely into $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection? To answer this we must put the requirements on a tree. The tree construction will be similar to the tree construction found in Section 4.1.2.

Fix a listing of all tuples $(j, C, \tilde{S}, \widehat{S}, \vec{X})$, where $j \neq i$ and $C \subseteq W$. We will build a tree which encodes whether $\widehat{S}$ witnesses that $\tilde{S}$ is a split of C.

This is a Π^0_2-piece of information and hence can be easily encoded into a tree. We will use the nodes of even length for this purpose. Let $(j, C, \tilde{S}, \widehat{S}, \vec{X})$ be the eth tuple. Let $\alpha \in 2^{<\omega}$ be of length $2e$. $\alpha\hat{}0$ will denote that $\widehat{S}$ witnesses that $\tilde{S} \subseteq S$ is a split of C and $\alpha\hat{}1$ will denote that this is not the case. Let $l(\alpha, s)$ be the greatest y such that for all $x < y$, if $x \in C_s$ then either $x \in \tilde{S}_s$ or $x \in \widehat{S}_s$. The $\liminf_s l(\alpha, s) = \infty$ iff $\widehat{S}$ witnesses that $\tilde{S} \subseteq S$ is a split of C.

We will use this function to determine the approximation to the true path at stage s, f_s. Assume that $\alpha \subseteq f_s$. Let $t < s$ be the last stage when $\alpha\hat{}0 \subseteq f_t$. Then $\alpha\hat{}0 \subseteq f_s$ iff $l(\alpha, t) < l(\alpha, s)$ (or t does not exist); i.e., s is α-expansionary. Let f be the true path; $f = \liminf_s f_s$ (under the standard ordering). Assume that $\alpha \subset f$. Then we can show that $\alpha\hat{}0 \subseteq f$ iff $\widehat{S}$ witnesses that $\tilde{S} \subseteq S$ is a split of C.

At the node $\beta = \alpha\hat{}0$ we will try to get a permanent ball for $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$'s collection. To help keep track of things, we will relabel $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ as $\mathcal{N}_\beta$ and denote β's collection as the collection for the requirement $\mathcal{N}_\beta$. Note that $|\beta|$ is of odd length.

At stages s when $\beta \subseteq f_s$ and we do not have a ball in β's collection we must take some action to get some balls into this collection. This might not be possible, in which case we must show that $\tilde{S}_\beta$ is a computable split of C_β.

Whether we get a ball permanently into β's collection is one more piece of information that we must encode in the tree. How we encode this information depends on how we get balls for this collection.

The node β will be given a number d_β and a function $u_\beta(s)$. At the first stage when $\beta \subseteq f_s$ or first such stage after being initialized we will set d_β to be large (just some number not yet occurring in the construction), $u_\beta(s)$ to be even larger and discontinue this stage of the construction. If $\beta \subseteq f_s$ and there is not a ball in β's collection, we will set $u_\beta(s)$ to be large; otherwise the value of $u_\beta(s)$ is $u_\beta(s-1)$. As always, the number d_β and the function $u_\beta(s)$ will be initialized if $f_t <_L \beta$, for some later stage t. As we will see, initialization allows nodes of higher priority to take balls for β's collection.

If $\beta \subseteq f_s$ and we increase $u_\beta(s)$ at this stage then we let $\beta\hat{}0 \subseteq f_s$; otherwise $\beta\hat{}1 \subseteq f_s$ (this is for all odd β). Since $u_\beta(s)$ is a nondecreasing function, if $\beta \subset f$ then $\beta\hat{}0 \subset f$ iff $\lim u_\beta(s) = \infty$ iff we do not have a permanent ball in β's collection. So if $\beta\hat{}0 \subset f$ we must show $\tilde{S}_\beta$ is computable; it is already a split of C.

Let $G_\beta = \{\gamma : |\gamma| = 2k+1 \wedge \gamma^-\hat{}0 = \gamma \subset \beta \wedge C_\gamma = C_\beta\}$. If $G_\beta = \emptyset$ then let $W_\beta = C_\beta$. Otherwise let $W_\beta = C_\beta \cap (\bigcap\{\widehat{S}_\gamma : \gamma \in G_\beta\})$. Consider a ball x. Let s be the least stage such that x is in the interval $[d_\beta, u_\beta(s)]$. Let t be the least stage such that $x \in W_{\beta,t}$, if t exists. If $t \geq s$ and x enters

$\tilde{S}_\beta$, then we will add x to β's collection. Any such ball is fair game, even balls in another node's collection.

Now assume that $\beta\hat{}0$ is on the true path and inductively assume that for all $\gamma \in G_\beta$ that $\tilde{S}_\gamma$ is computable. We must show $\tilde{S}_\beta$ is computable. By our inductive hypothesis, $\bigcup\{\tilde{S}_\gamma : \gamma \in G_\beta\}$ is a computable split of C_β and $W_\beta = C_\beta - \bigcup\{\tilde{S}_\gamma : \gamma \in G_\beta\}$. So it is enough to show $W_\beta \cap \tilde{S}_\beta$ is computable.

Assume that d_β is never reset after stage t and for all δ if either $\delta <_L \beta$ or $\delta \subset \beta$ and δ's collection has a permanent member then it will have one by stage t. So after stage t, no δ will remove a ball from β's collection. Let $x \geq d_\beta$. Wait for a least stage $s \geq t$ such that $x \leq u_\beta(s)$. If x later enters W_β it must enter $\widehat{S}_\beta$; otherwise we will take x for β's collection. So $x \in W_\beta \cap \tilde{S}_\beta$ iff $x \in W_\beta \cap \tilde{S}_\beta$ at stage s and hence $W_\beta \cap \tilde{S}_\beta$ is computable. □

6. Variations on Theorem 5.1

In the next several sections we will present various modifications to Theorem 5.1. We will do this in a piecemeal fashion.

Let F be a computably enumerable set and J be a Σ_3^F set. Our goal is to use this coding to code J into A. For more details see Section 6.5.

6.1. **Revisiting the negative requirements.** The negative requirements are the following:

$(\mathcal{N}_{j,C})$ build a $\mathcal{B}$-interpretation, $\vec{B}_{j,C}$, such that for all $\tilde{S}$ and $\vec{X}$,

$(\mathcal{N}_{j,C,\tilde{S},\vec{X}})$ $\vec{X}$ does not witness $\varphi_{\mathcal{P}_j}(\emptyset, \vec{U}, \vec{B}, \tilde{S})$ or, $\tilde{S}$ is computable.

In the proof of Theorem 5.1, we met all these requirements for all $C \subseteq W$, for all $j \neq i$ and all splits $\tilde{S}$ of C. In fact, for all j and C, we uniformly found $\vec{B}_{j,C}$.

This uniformity will be important in other applications (see Cholak and Harrington [3]) but we do not need it here. We will be willing to drop the uniformity if we could meet $\mathcal{N}_{j,C}$ for all C, not just the C which are subsets of W.

Fix j. Lets consider any computably enumerable set C. We would like to meet $\mathcal{N}_{j,C}$ and $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$, for all splits $\tilde{S}$ of C and all $\vec{X}$. Remark 5.2 tells us how to do this part of this. Let $B_{b_0} = B_{b_2^l} = B_{b_4^j} = B_{b_5} = C$, for all $l \leq j$ (and let $U_i = \emptyset$). Without regard to a ball x's position in $\mathcal{P}_i$ just move all balls x in C downward and leftward in $\mathcal{P}_j$. Now if $\tilde{S}$ is a split of C such that $\tilde{S} - S$ is nonempty we will be able to meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ since for all

l, $U_l \cap (\tilde{S} - S)$ is empty. However if $\tilde{S} \subset S$ we may not have met $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$. (For more about these items, see the beginning of Section 5.1.6.) But then we can meet $\mathcal{N}_{j,C,\tilde{S},\vec{X}}$ exactly as we did in the proof of Theorem 5.1. So to meet $\mathcal{N}_{j,C}$ for all C, we build two $\vec{B}_{j,C}$, one as above and one as in the proof of Theorem 5.1. Note we do not lose any uniformity in the construction of $\vec{U}$ or $\vec{D}$ (this will be important; see Sections 6.4 and 6.5). Hence we have the following:

Corollary 6.1. *Effectively in i, W, a $\mathcal{B}$-interpretation $\vec{B}_S$ and S a subset of W, there is a $\vec{U}$ and $\vec{D}$ such that $\vec{D}$ witnesses that $\varphi_{\mathcal{P}_i}(\emptyset, \vec{U}, \vec{B}_S, S)$ holds (and all the components of $\vec{U}$ and $\vec{D}$ are contained in S). Furthermore, for all $j \neq i$, and for all C, $\vec{U}$, $A = \emptyset$, and C do not realize $\mathcal{P}_j$.*

6.2. **Building A.** We want to mix Corollary 6.1 with the construction of A. The key to proving Theorem 5.1 is to ensure that each β has a permanent member in its collection or $\tilde{S}_\beta$ is computable. However if $A \neq \emptyset$ then when a ball enters A we will remove it from all collections. The formula φ only involves balls outside of A. To prove a theorem like Corollary 6.1 with $A \neq \emptyset$ it would be enough if each β either has a permanent member (outside of A) or $\tilde{S}$ is computable modulo A. (Whether A is empty or not has no impact on the strategy outlined in Section 6.1.) This is possible if we are building A inside W via finitary positive requirements and we can restrict balls in β's collection from entering A with priority β.

All the above details are the same except for the last paragraph of the proof of Theorem 5.1. We change the last two sentences to read: "If x later enters W_β it must enter $\widehat{S}_\beta$ *or* A; otherwise we will take x for β's collection. So $x \in (W_\beta \cap \tilde{S}_\beta) - A$ iff $x \in (W_{\beta,s} \cap \tilde{S}_{\beta,s}) - A_s$ and hence $W_\beta \cap \tilde{S}_\beta$ is computable modulo A."

We also would like to expand the construction of A so it need not be completely within W. For this we will introduce a computably enumerable set $\tilde{W}$. We will build A completely within $\tilde{W}$. W will be a split of $\tilde{W}$ and inside W we will build A as assumed above. We do not care how A is being constructed in $\tilde{W} - W$ but since W is a split of $\tilde{W}$ this construction does not spill over into W.

Corollary 6.2. *Let $\tilde{W}$ be a computably enumerable set and W a computably enumerable split of $\tilde{W}$. Let A be a computably enumerable set being constructed completely inside $\tilde{W}$ such that, inside W, A is being constructed via finitary positive requirements and we can restrict balls in β's collection from entering A with priority β.*

Effectively in i, W, a $\mathcal{B}$-interpretation $\vec{B}_S$ and S a subset of W, there is a $\vec{U}$ and $\vec{D}$ such that $\vec{D}$ witnesses that $\varphi_{\mathcal{P}_i}(A, \vec{U}, \vec{B}_S, S)$ holds (and all the

components of $\vec{U}$ and $\vec{D}$ are contained in S). Furthermore, for all $j \neq i$, and for all C, $\vec{U}$, A, and C do not realize $\mathcal{P}_j$.

6.3. **Building S.** We would like to build a computably enumerable set S such that S is a split of W and S is noncomputable modulo A. To do this we are going to need to make some assumptions about $\tilde{W}$ and W.

Since W is a split of $\tilde{W}$, S will be a split of $\tilde{W}$. Therefore if S is going to be noncomputable modulo A, $\tilde{W}$ must be noncomputable modulo A. Otherwise all computably enumerable splits of $\tilde{W}$ are computable modulo A. Hence $\tilde{W}$ must be noncomputable modulo A. Since $A \subseteq \tilde{W}$, if $\tilde{W}$ is simple then $\tilde{W}$ is not computable modulo A. Lets assume that $\tilde{W}$ is simple.

To see what kind of assumptions we need on W and how we build S the following lemma will prove very useful:

Lemma 6.3. *Assume that $A \subset \tilde{W}$ and that $\tilde{W}$ is noncomputable modulo A. Let $Y \subseteq \tilde{W}$ be noncomputable modulo A. If we build $X \subseteq Y$ to meet the following requirements*

$(\mathcal{R}_e)$ *if $W_e \searrow Y$ is infinite then $(W_e \cap X) - A \neq \emptyset$.*

then X is noncomputable modulo A.

Proof. Z is noncomputable modulo A iff for all W_e, either $W_e \cup Z \cup A \neq \omega$ or $W_e \cap Z \not\subseteq A$. If $W_e \cup Y \cup A \neq \omega$ then no matter how we build $X \subseteq Y$, $W_e \cup X \cup A \neq \omega$. If $W_e \cap Y \not\subseteq A$ then $W_e \searrow Y$ is infinite; otherwise $(W_e \backslash Y) - (W_e \searrow Y)$ is a computably enumerable set witnessing that Y is computable modulo A. So to make X noncomputable modulo A it is enough to build X such that if $W_e \searrow W$ is infinite then $W_e \cap X \not\subseteq A$. □

Now let us mix this with Corollary 6.2. So in addition to the assumption that $\tilde{W}$ is noncomputable modulo A, let's assume A is a computably enumerable set being constructed inside W via finitary positive requirements and we can restrict balls in β's collection with priority β.

W will be externally built as a Friedberg split of $\tilde{W}$. Recall if X is a Friedberg split of Y then $W_e \searrow Y$ is infinite implies that $W_e \cap X \neq A$ and, in fact, $W_e \searrow X$ is infinite [see 16, Theorem X.2.1]. So if we can restrain at least one of the balls in $W_e \searrow W$ from entering A with priority e, W will be noncomputable modulo A. If $|\beta| = 2e + 1$ then in addition to β's normal duties we will let β restrain a ball x in $W_e \cap W$ from entering A. When we do this we will not add x to β's collection but allow x to enter any collection as before. We provide some more details in Section 6.5. So W is noncomputable modulo A.

We can build S to be a Friedberg split of W [again see 16, Theorem X.2.1]. Now if we let β also restrain a ball in $W_e \searrow S$ from entering A,

S will be noncomputable modulo A. The construction for the Friedberg Splitting Theorem is uniform. Hence we have the following corollary:

Corollary 6.4. *Let $\tilde{W}$ be a simple set and W be a Friedberg split of $\tilde{W}$. Let A be a computably enumerable set being constructed completely inside $\tilde{W}$ such that, inside W, A is being constructed via finitary positive requirements and we can restrict balls (in β's collection and possibly others) from entering A with priority β.*

Effectively in i, W, and a $\mathcal{B}$-interpretation $\vec{B}_S$ there is an S, $\vec{U}$ and $\vec{D}$ such that S is a Friedberg split of W, S is noncomputable modulo A, and $\vec{D}$ witnesses that $\varphi_{\mathcal{P}_i}(A, \vec{U}, \vec{B}_S, S)$ holds (and all the components of $\vec{U}$ and $\vec{D}$ are contained in S). Furthermore, for all $j \neq i$, and for all C, $\vec{U}$, A, and C do not realize $\mathcal{P}_j$.

6.4. **Realizing $\mathcal{P}_i$.** Recall, from Theorem 4.1, for $\vec{U}$, A and W to realize $\mathcal{P}_i$ there must be a split S of W such that S is not computable modulo A and $\varphi_{\mathcal{P}_i}(A, \vec{U}, \vec{B}_e, S)$, where e codes $\vec{U}$, $\mathcal{P}_i$, A and W. So if we apply the above corollary to $\vec{B}_e$, where e codes $\vec{U}$, $\mathcal{P}_i$, A and W and is arrived at by an application of the Recursion Theorem to the above corollary (the above corollary builds $\vec{U}$ which must be included in the code and it is here that we care that $\vec{U}$ is constructed uniformly) then we have the following corollary:

Corollary 6.5. *Let $\tilde{W}$ be a simple set and W be a Friedberg split of $\tilde{W}$. Let A be a computably enumerable set being constructed completely inside $\tilde{W}$ such that, inside W, A is being constructed via finitary positive requirements and we can restrict balls (in β's collection and possibly others) from entering A with priority β.*

Effectively in i, there is a $\vec{U}$ such that $\vec{U}$, A, and W realize $\mathcal{P}_i$ (and all the components of $\vec{U}$ are contained in W). Furthermore, for all $j \neq i$, and for all C, $\vec{U}$, A, and C do not realize $\mathcal{P}_j$.

6.5. **Actually building A.** Now it is time to actually build A. We start with a computably enumerable set F and a set J which is Σ_3^F. We want to construct A and $\vec{U}$ such that A and F have the same Turing degree and $i \in J$ iff there is a Y such that $\vec{U}$, A and Y realize $\mathcal{P}_i$. We want to code J into A and $\vec{U}$.

Whether $i \in J$ is not an effective piece of information for all i but it is Σ_3^F. The question of "is Ψ_n^F total" is Π_2^F-complete. So we can construct a uniform sequence of functionals $\Psi_{\langle i,n\rangle}^F$ such that $i \in J$ iff there is n, $\Psi_{\langle i,n\rangle}^F$ is total.

Let $\tilde{W}$ be a simple set of the same Turing degree as F (every computably enumerable degree contains a simple set). Using the Friedberg Splitting

Theorem we will break $\tilde{W}$ uniformly into infinitely many Friedberg splits, W_{-1} and $W_{i,n}$, for all i and n.

We apply Corollary 6.5 to $W_{i,n}$ to code i and construct $\vec{U}_{i,n}$. Lets assume that everything in the above corollaries is indexed by i and n. We will use $\Psi^F_{\langle i,n\rangle}$ to construct A inside $W_{i,n}$. We hope to satisfy the hypothesis of Corollary 6.5.

6.5.1. *Building inside $W_{i,n}$.* Assume that $|\beta| = 2e+1$. At stages s that $\Psi^{F_s}_s(e)$ converges we will allow β to add balls to its collection and restrain them out of A with priority β and if some x in W_e enters $W_{i,n}$ or the needed $S_{i,n}$ at stage s (and we are not restraining such balls already with priority β) we will restrain x out of A with priority β. If there is a later stage t such that $F_t \upharpoonright \psi_s(e) \neq F_s \upharpoonright \psi_s(e)$ then we will dump all balls restrained with priority β into $A \cap W_{i,n}$. If x enters $W_{i,n}$ and is not immediately restrained we will dump x into A.

6.5.2. *The realization and nonrealization of patterns.* Before we continue we should recall the following fact from Section 3.5: If, for all l, $V_l \cap W = U_l \cap W$ then $\vec{U}$, A and W realize $\mathcal{P}$ iff $\vec{V}$, A and W do.

Since Corollary 6.5 is uniform and all $W_{i,n}$ are disjoint, we can let $U_l = \bigcup\{U_{i,n,l} : i,n \in \omega\}$ for $l = 0,1,2,3$. If $\Psi^F_{\langle i,n\rangle}$ is total then the construction of A inside $W_{i,n}$ meets the hypothesis of Corollary 6.5 and we have that $\vec{U}$, A, and $W_{i,n}$ realize $\mathcal{P}_i$. So if $i \in J$ then there is a $W_{i,n}$ such that $\vec{U}$, A and $W_{i,n}$ realize $\mathcal{P}_i$.

Let $j \notin J$ then for all n, $\Psi^F_{\langle j,n\rangle}$ is not total. If $\Psi^F_{\langle j,n\rangle}$ is not total then $A \cap W_{j,n} =^* W_{j,n}$ and therefore no split of $W_{j,n}$ is noncomputable modulo A. Hence for all i and n, $\vec{U}_{i,n}$, A, and $W_{i,n}$ do not realize $\mathcal{P}_i$.

Let's quickly revisit the negative requirements to make sure for all Y, $\vec{U}$, A, and Y do not realize $\mathcal{P}_i$. The key is to meet $\mathcal{N}_{i,Y,\tilde{S},\vec{X}}$ for the above $\vec{U}$, where $\tilde{S}$ is a split of Y. Now if $\tilde{S} \searrow \tilde{W}$ is infinite then for all i and n, $\tilde{S} \searrow W_{i,n}$ is infinite and hence inside $W_{i,n}$ there is a witness to $\mathcal{N}_{i,Y,\tilde{S},\vec{X}}$. If $\tilde{S} \searrow W_{i,n}$ is finite then, by the construction of A, $\tilde{S} \cap W_{i,n} \subseteq^* A$. So if $\tilde{S} \searrow \tilde{W}$ is finite and $\tilde{S} \not\subseteq^* A$ then $\tilde{S} - A \subseteq^* W_{-1}$. But then we can realize $\mathcal{N}_{i,Y,\tilde{S},\vec{X}}$ exactly as we discussed in Section 6.1. It does not matter how we build A on W_{-1} as long as all components of $\vec{U}$ are empty on $W_{-1}0$, which is clearly the case.

Therefore there is a $\vec{U}$ such that $i \in J$ iff there is a Y where $\vec{U}$, A and Y realize $\mathcal{P}_i$.

6.5.3. *The degree of A.* So far A is computable from F. Given x use F to determine if x enters $\tilde{W}$. If x enters $\tilde{W}$, x will later enter some $W_{i,n}$ (or

W_{-1}). When x enters $W_{i,n}$ at stage s either x will be put into A or restrained out of A with priority β, where $|\beta| = 2e+1$. If restrained at stage s then x enters A iff $F_s \upharpoonright \psi_s(e)$ changes after stage s.

Inside W_{-1} we will code F into A. The easiest way to do this is to create a computable bijection b from ω to W_{-1} and let $b(x) \in A$ iff $x \in F$. So $W_{-1} \cap A$ has the same Turing degree as F and hence A and F have the same Turing degree.

6.5.4. *The coding.* In this section we will sum up the work in the past few sections. The following definition will prove useful:

Definition 6.6. Fix $\mathcal{L}$, an $\mathcal{L}$-interpretation $\vec{U}$ and a computably enumerable set A. Let $\mathcal{J}_{A,\vec{U}}$ (this should also be indexed by $\mathcal{L}$ but $\vec{U}$ determines $\mathcal{L}$) be the set of special $\mathcal{L}$-patterns $\mathcal{P} = (\mathcal{T}, \mathcal{R}, \mathcal{B}, l)$ such that there exists W, such that $\vec{U}$, A and W realize the pattern $\mathcal{P}$.

Theorem 6.7. *Given the $\mathcal{L}$-patterns $\mathcal{P}_i$ defined in Section 3.6, for all computably enumerable F and all sets J, if J is Σ_3^F then there is a computably enumerable set A and an $\mathcal{L}$-interpretation $\vec{U}$ such that $A \equiv_T F$ and $i \in J$ iff $\mathcal{P}_i \in \mathcal{J}_{A,\vec{U}}$.*

$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$ can be written as a statement in the language $L = \{\subseteq\}$. $\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$ is a property of $\mathcal{L}(A)$. That is, if $\mathcal{L}(A)$ is isomorphic to $\mathcal{L}(F)$ via Φ then $\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$ iff $\mathcal{P} \in \mathcal{J}_{F,\Phi(\vec{U})}$. (As it stands this is not true for $\mathcal{L}^*(A)$.) Hence the statement "$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is invariant under automorphisms of $\mathcal{E}$.

Hence the above theorem codes J into A and $\vec{U}$ in a way which is invariant under automorphisms of $\mathcal{E}$. To properly use this coding we must be able to decode it. We would like to be able to decode this in a fashion which is Σ_3^A. The following lemma is a start. We will need a few more theorems before we get the result we want; see Section 8.

Lemma 6.8. *$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$ is Σ_4^0.*

Proof. By Corollary 4.2, whether $\vec{U}$, A, and W realize $\mathcal{P}$ is Σ_4^0. Now just chase through the definition of $\mathcal{J}_{A,\vec{U}}$. □

7. True stage enumerations

We need to take a short computably-theoretic break concerning true stage enumerations and splits which are not computable modulo A. The main goal of this section is to show Theorem 7.3. This theorem is used in Definition 8.1 and the proof of Theorem 8.2.

Definition 7.1. Let $\{A_s\}$ be an enumeration of A. Let

$$\overline{A}_s = \{a_0^s < a_1^s < \ldots\}$$

and

$$\overline{A} = \{a_0 < a_1 < \ldots\}.$$

Then $\{A_s\}$ is a *true stage enumeration* of A if there are infinitely many s such that $a_s^s = a_s$. A stage s is a *true stage* if $a_s^s = a_s$.

Whether a stage s is a true stage for A (and some fixed enumeration) of A is Π_1^0 (for all t, $a_s^t = a_s^s$) and it is also Δ_0^A.

Lemma 7.2. *If A is not high, then A has a true stage enumeration.*

Proof. Fix some enumeration $\{E_s\}$ of A. Let $g = \mu s[e_x^s = a_x]$ (where e_x^s is defined as in Definition 7.1 w.r.t. $\{E_s\}$). $g \leq_T A$ and hence g fails to dominate some computable function f. Define $A_s = E_{f(s)}$. If $f(s) > g(s)$ then $a_s^s = a_s^{f(s)} = e_s^{g(s)} = a_s$. □

A slightly stronger lemma and its proof can be found in Soare [16, page 209, Lemma XI.1.6].

Theorem 7.3. *Suppose that A has a true stage enumeration. Uniformly in e, there is a computably enumerable $F_e \subseteq W_e$ with a computable enumeration $\{F_{e,s}\}$ such that for all splits, S, of W_e:*

(1) *if (a) S is not computable modulo A then (b) there are infinitely many x and stages $s > x$ such that s is a true stage, $x \in S$, $x \notin F_{e,s}$ and $x \notin A_s$;*
(2) *if (b) then (c) $S \cap F_e$ is not computable modulo A.*

As we mentioned in Section 3.4, S is noncomputable modulo A is a Π_3^0 statement. Since "s is a true stage" is Π_1^0 and Δ_0^A, (b) is a Π_3^0 statement and a Π_2^A statement. Being Π_2^A will prove very useful in the proof of Theorem 8.2.

7.1. **Proof of Theorem 7.3.** To make our lives more sane we will drop the index e. We can safety assume via the Recursion Theorem that we have an index for F. For this proof, we will find the alternative definition of computable modulo A (see Definition 3.16) more useful. The proof will be a tree construction, a standard Π_2^0 tree argument like the previous constructions. But first we will warm up with the requirements.

7.1.1. *The first condition.* The failure of (b) implies that $S \cap \overline{F} \subseteq^* A$. Since we are building F we can stop enumerating balls x into F. Therefore $S \subseteq^* A$ and clearly S is computable modulo A. However we must do this for all splits of W.

Let $\{S_i, \widehat{S_i}\}$ be a list of all pairs of computably enumerable sets. It is a Π_2^0 question to determine whether $\widehat{S_i}$ witnesses that S_i is a split of W. Whether (b) fails to hold for S_i is a Σ_3^0-property. We will assume that this property

is written in the form $\exists j \Theta(j, i)$, where Θ is Π^0_2 (and is different from the Θ discussed in Section 4.1.2).

To meet (1), it is enough to meet the following negative requirements:

$$(\mathcal{N}_{i,j}) \qquad \begin{array}{c} \widehat{S_i} \text{ witnesses that } S_i \text{ is a split of } W \\ \text{and } \Theta(j, i) \text{ then } F \subseteq_{\mathcal{R}} \widehat{S_i} \cup A, \end{array}$$

where $F \subseteq_{\mathcal{R}} \widehat{S_i} \cup A$ iff there is a computable set $R \subseteq F$ such that $F - R \subseteq \widehat{S_i} \cup A$.

Assume that S_i is a split of W witnesses by $\widehat{S_i}$ and that (b) fails for S_i. Then there is a j such that $\Theta(j, i)$. Assume that we met $\mathcal{N}_{i,j}$. Therefore $F - R \subseteq \widehat{S_i} \cup A$, for some computable set $R \subseteq F$. Now since (b) fails for S_i, $S_i \cap \overline{F} \subseteq^* A$. So $S_i \cap F \subseteq \widehat{S_i} \cup R \cup A$. But, since S_i and $\widehat{S_i}$ are disjoint, $S_i \cap F \subseteq R \cup A$. Since R is a computable subset of F which is a subset of W and S_i is a split of W, $X = R \cap S_i$ is computable. So there is a computable set $X = R \cap S_i$ such that $X \subseteq S_i$ and $S_i \subseteq^* X \cup A$ and hence S_i is computable modulo A.

Meeting $\mathcal{N}_{i,j}$ will prove to be rather straightforward. We just try to restrain as much as we can from entering F. The computable subset R of F will just come from the tree construction.

7.1.2. *The second condition.* If (b) fails for S_i then the second condition holds trivially. Assume that (b) holds for S_i. We wish to ensure that no computable set R witnesses that $S_i \cap F$ is computable modulo A. So we must show for all computable sets R either $R \not\subseteq S_i \cap F$ or $S_i \cap F \not\subseteq R \cup A$.

Let $\{R_j, \widehat{R}_j\}$ be another list of all pairs of computably enumerable sets. It is a Π^0_2 question whether $\widehat{R}_j$ witnesses that R_j is computable; $R_j \sqcup \widehat{R}_j = \omega$.

To meet (2), under the condition that (b) holds for S_i, it will be enough to meet the following requirements:

$$(\mathcal{P}_{i,j}) \qquad \begin{array}{c} \widehat{S_i} \text{ witnesses that } S_i \text{ is a split of } W \text{ and} \\ \widehat{R}_j \text{ witnesses that } R_j is \text{ computable } \Longrightarrow \\ R_j \not\subseteq S_i \cap F \vee S_i \cap F \not\subseteq R_j \cup A. \end{array}$$

We meet $\mathcal{P}_{i,j}$ by searching for a ball x and a true stage s such that $x \in S_{i,s}$, $x \notin F_s$ and $x \notin A_s$. Since (b) holds for S_i, there will be infinitely many such balls and stages. Once we have such a ball x we will restrain x from entering F until x enters $\overline{R}_j$. Since we are keeping x out of F, if x enters R_j then $R_j \not\subseteq S_i \cap F$. If x enters $\overline{R}_j$ then we add x to F (this is the only enumeration into F) and $x \in S_i \cap F \cap \overline{R}_j \cap \overline{A}$. In either case, R_j does not witness that $S_i \cap F$ is computable modulo A. Hence if (b) holds for S_i then $S_i \cap F$ is not computable modulo A.

7.1.3. *The tree, true path and approximation to the true path.* Let $\alpha \in 2^{<\omega}$ be of length $2\langle i, j\rangle + 1$. Now $\Theta(j, i)$ is Π^0_2. So lets assume $\Theta(j, i)$ is of the form $(\forall x)(\exists s)\Theta^*(j, i, x, s)$, where $\Theta^*(j, i, x, s)$ is Δ^0_0. (Note that we can and must find $\Theta(j, i)$ and $\Theta^*(j, i, x, s)$ uniformly from i.) Let $l_\alpha(s)$ be the greatest y such that for all $x < y$ if $x \in W_s$ then either $x \in S_{i,s}$ or $x \in \widehat{S}_{i,s}$ and there is a $t \leq s$ where $\Theta^*(j, i, x, t)$. Assume that $\alpha \subseteq f_s$. Let t be the last stage such that $\alpha\hat{}0 \subset f_t$. Then $\alpha\hat{}0 \subseteq f_s$ iff $l_\alpha(s) > l_\alpha(t)$ (or t does not exist); i.e., s is an α-expansionary stage.

Let $\delta \in 2^{<\omega}$ be of length $2\langle i, j\rangle$. Let $l_\delta(s)$ be the greatest y such that for all $x < y$ either $x \in R_{j,s}$ or $x \in \widehat{R}_{j,s}$ and if $x \in W_s$ then either $x \in S_{i,s}$ or $x \in \widehat{S}_{i,s}$. Assume that $\delta \subseteq f_s$. Let t be the last stage such that $\delta\hat{}0 \subset f_t$. Then $\delta\hat{}0 \subset f_s$ iff $l_\delta(s) > l_\delta(t)$ (or t does not exist); i.e., s is a δ-expansionary stage.

Let $f = \liminf f_s$. It is not hard to show that $\alpha\hat{}0 \subset f$ iff $\widehat{S}_i$ witnesses that S_i is a split of W and $\Theta(j, i)$ and $\delta\hat{}0 \subset f$ iff $\widehat{S}_i$ witnesses that S_i is a split of W and $\widehat{R}_j$ witnesses that R_j is a computable set.

7.1.4. *The construction of F.* Let $\delta \in 2^{<\omega}$ be of length $2\langle i, j\rangle = k$ and let $\beta = \delta\hat{}0$. Assume that $\beta \subseteq f_t$, where $t \leq s$, and β has not been initialized since stage t. If β does not have a witness, x_β, to $\mathcal{P}_{i,j}$ at stage s and there is a ball x such that $x > k$, $x \in S_{i,s}$, $x \notin F_s$, $x \notin A_s$, x is not a witness for any $\gamma \subset \beta$ or $\gamma <_L \beta$, if $\gamma\hat{}0 \subseteq \beta$ where γ is of length $2\langle i, j\rangle + 1$ then $x \in \widehat{S}_{i,s}$, and for all stages t', if $x \leq t' \leq s$, it is not the case that $f_{t'} \leq_L \beta$ then choose the least such ball x as β's witness. If β's witness enters A then it is released and β starts looking for a new witness. If x_β enters $\widehat{R}_j$ then we add x_β to F. Otherwise x_β remains out of F.

7.1.5. *The verification.* Assume that $\widehat{S}_i$ witnesses that S_i is a split of W and $\Theta(j, i)$. Let $\gamma = \alpha\hat{}0 \subset f$ be such that the length of α is $2\langle i, j\rangle + 1$. Define R as follows: Given x wait for a stage $s \geq x$ such that $\gamma \subseteq f_s$. $x \in R$ iff $x \in F_s$. By the construction of F, except for finitely many balls not in A (the final witness for $\beta \subset \gamma$ or $\beta <_L \gamma$), if $x \notin R$ enters F then $x \in \widehat{S}_i$. So $\mathcal{N}_{i,j}$ is met.

It remains to show that $\mathcal{P}_{i,j}$ is met. Let $\beta = \delta\hat{}0 \subseteq f$ where δ is of length $2\langle i, j\rangle$. It is enough to show that there is a stage s after which β's witness does not change. We will do this by induction on β. So there is a stage t such that if $\gamma \subset \beta$ or $\gamma <_L \beta$ and γ has a permanent witness it has one by stage t and for all stages $s > t$, $f_s \not<_L \beta$.

Let

$$G = \{\alpha : \alpha\hat{}0 \subseteq \beta \text{ and } |\alpha| = 2\langle i, j\rangle + 1 = 2\langle i_\alpha, j\rangle + 1\}.$$

So (b) fails for S_{i_α} and hence (b) fails for $S = \bigcup\{S_{i_\alpha} : \alpha \in G\}$. Hence if (b) holds for S_i then (b) must also hold for $S_i \cap \bigcap\{\widehat{S}_{i_\alpha} : \alpha \in G\}$. Therefore it is only a matter of time before β gets a permanent witness. □

8. The definability of the double jump

In this section we will focus on the decoding that is needed to show the double jump is definable in the computably enumerable sets.

Since Theorem 7.3 is uniform we can make the following definition:

Definition 8.1. Suppose A has a true stage enumeration, then there is a computable function $\hat{e}$ such that, for all e, $W_{\hat{e}} = F_e$, where F_e is from Theorem 7.3.

Theorem 8.2. *Suppose that A has a true stage enumeration. $\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$ iff there is an e and there is a split S of W_e such that there are infinitely many x and stages $s > x$ where s is a true stage, $x \in S$, $x \notin F_{e,s}$ and $x \notin A_s$ and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_{\hat{e}}, S \cap F_e)$. Hence "$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is Σ^A_3.*

Proof. If $\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$ then there is a W_e such that for all $\vec{B}$ there is a split S of W_e where S is not computable modulo A and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}, S)$. Let $\vec{B} = \vec{B}_{\hat{e}}$. Now we get a split S of W_e such that S is not computable modulo A holds and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_{\hat{e}}, S)$. But then, by Theorem 7.3, there are infinitely many x and stages $s > x$ such that s is a true stage, $x \in S$, $x \notin F_{e,s}$ and $x \notin A_s$. Notice that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_{\hat{e}}, S)$ implies $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_{\hat{e}}, S \cap F_e)$ (see the comment at the end of Section 3.3).

Assume that there is a split S of W_e such that there are infinitely many x and stages $s > x$ where s is a true stage, $x \in S$, $x \notin F_{e,s}$ and $x \notin A_s$ (this is (b) from Theorem 7.3) and $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_{\hat{e}}, S \cap F_e)$. By Theorem 7.3, $S \cap F_e$ is not computable modulo A. We have $F_e = W_{\hat{e}} \subseteq W_e$. Hence there is a split $S \cap F_e$ of $W_{\hat{e}}$ such that $\varphi_{\mathcal{P}}(A, \vec{U}, \vec{B}_{\hat{e}}, S \cap F_e)$ and $S \cap F_e$ is not computable modulo A. Therefore, by Theorem 4.1, $\vec{U}$, A and $W_{\hat{e}}$ realize $\mathcal{P}$ and $\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$. □

Corollary 8.3. *"$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is Σ^A_3.*

Proof. By Lemma 7.2 and Theorem 8.2, we only need worry about A where A is high. "$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is Σ^0_4, by Lemma 6.8. So "$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is $\Sigma^{\mathbf{0}''}_2$. Since A is high, "$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is $\Sigma^{A'}_2$. Hence "$\mathcal{P} \in \mathcal{J}_{A,\vec{U}}$" is Σ^A_3. □

Definition 8.4. Let $\varphi_J(A)$ be the $\mathcal{L}_{\omega_1,\omega}$ sentence: There exists an $\mathcal{L}$-interpretation $\vec{U}$ such that for all i, $i \in J$ iff $\mathcal{P}_i \in \mathcal{J}_{A,\vec{U}}$.

By Theorem 6.7, for all computably enumerable F and for all sets J if J is Σ^F_3 then there is a computably enumerable A such that $\varphi_J(A)$. By

Corollary 8.3, if $\varphi_J(A)$ then J is Σ^A_3. If $\varphi_J(A)$ and $\mathbf{b} \geq_T A$ then, by Theorem 6.7 and Corollary 8.3, there is a B such that $B \in \mathbf{b}$ and $\varphi_J(B)$. So the coding (at least the current version) is closed upward in the sense that it cannot be used to separate A from any $\mathbf{b}$ such that $\mathbf{b} \geq_T A$.

$\varphi_J(A)$ is a property of $\mathcal{L}(A)$. If $\mathcal{L}(A)$ and $\mathcal{L}(B)$ are isomorphic then $\varphi_J(A)$ iff $\varphi_J(B)$. Hence $\varphi_J(A)$ is invariant under automorphisms of $\mathcal{E}$.

Theorem 8.5. *Let $\mathcal{C} = \{\mathbf{a} : \mathbf{a}$ is the Turing degree of a Σ^0_3 set $J \geq_T \mathbf{0}''\}$. Let $\mathcal{D} \subseteq \mathcal{C}$ such that $\mathcal{D}$ is upward closed in $\mathcal{C}$. Then there is an $\mathcal{L}(A)$ property $\varphi_{\mathcal{D}}(A)$ such that*

$$(\forall \text{c.e. } F)[F'' \in \mathcal{D} \iff (\exists A)[\varphi_{\mathcal{D}}(A) \text{ and } A \equiv_T F]].$$

Proof. Let $\varphi_{\mathcal{D}}(A)$ be the infinite disjunction of all $\varphi_J(A)$ where J is a Π^0_3 set whose Turing degree is in $\mathcal{D}$.

If $F'' \in \mathcal{D}$ then there is a Π^0_3 set J in the above disjunction such that J is Σ^F_3 and hence, by Theorem 6.7, there is an A such that $A \equiv F$ and $\varphi_J(A)$.

Assume there is an A such that $A \equiv F$ and $\varphi_J(A)$, for some J in the above disjunction. Then, by Theorem 8.2, J is Σ^A_3, and, since J is Π^0_3, J is Δ^A_2 and Δ^F_2. So $J \leq_T F''$ and $F'' \in \mathcal{D}$. □

This is our main result and from it follow the three corollaries mentioned in the introduction (Corollaries 1.3, 1.4, and 1.5).

References

[1] Peter Cholak. Notes on 3 theorems by Leo Harrington. Handwritten Notes, 1994. 1.5

[2] Peter Cholak. Automorphisms of the lattice of recursively enumerable sets. *Mem. Amer. Math. Soc.*, 113(541):viii+151, 1995. ISSN 0065-9266. 1.2, 1.7

[3] Peter Cholak and Leo A. Harrington. Δ^0_3-automorphisms of the computably enumerable sets. In preparation, . 1.5, 6.1

[4] Peter Cholak and Leo A. Harrington. Extension theorems and automorphisms of the computably enumerable sets. In preparation, . 1.5

[5] Leo A. Harrington and André Nies. Coding in the partial order of enumerable sets. *Adv. Math.*, 133(1):133–162, 1998. ISSN 0001-8708. 1.5

[6] Leo A. Harrington and Robert I. Soare. Noninvariance and low computably enumerable sets. To appear. 1.2

[7] Leo A. Harrington and Robert I. Soare. Definability, automorphisms, and dynamic properties of computably enumerable sets. *Bull. Symbolic Logic*, 2(2):199–213, 1996. ISSN 1079-8986. 1.1, 1.2

[8] Leo A. Harrington and Robert I. Soare. The Δ^0_3-automorphism method and noninvariant classes of degrees. *J. Amer. Math. Soc.*, 9(3):617–666, 1996. ISSN 0894-0347. 1.2, 2.1

[9] A. H. Lachlan. Degrees of recursively enumerable sets which have no maximal supersets. *J. Symbolic Logic*, 33:431–443, 1968. 1.1

[10] Manuel Lerman and Robert I. Soare. d-simple sets, small sets, and degree classes. *Pacific J. Math.*, 87(1):135–155, 1980. ISSN 0030-8730. 1.1

[11] W. Maass, Richard A. Shore, and M. Stob. Splitting properties and jump classes. *Israel J. Math.*, 39:210–224, 1981. 1.2

[12] D. A. Martin. Classes of recursively enumerable sets and degrees of unsolvability. *Z. Math. Logik Grundlag. Math.*, 12:295–310, 1966. 1.1

[13] James C. Owings. Recursion, metarecursion, and inclusion. *J. Symbolic Logic*, 32:173–178, 1967.

[14] Joseph R. Shoenfield. Degrees of classes of recursively enumerable sets. *J. Symbolic Logic*, 41:695–696, 1976. 1.1

[15] Robert I. Soare. The new extension theorem. To appear.

[16] Robert I. Soare. *Recursively Enumerable Sets and Degrees*. Perspectives in Mathematical Logic, Omega Series. Springer–Verlag, Heidelberg, 1987. 1, 4.1.1, 6.3, 7

Department of Mathematics, University of Notre Dame, Notre Dame, IN 46556-5683

E-mail address: Peter.Cholak.1@nd.edu

Department of Mathematics, University of California, Berkeley, CA 94720-3840

E-mail address: leo@math.berkeley.edu