DOMINATING THE ERDŐS-MOSER THEOREM IN REVERSE MATHEMATICS

LUDOVIC PATEY

Abstract. The Erdős-Moser theorem (EM) states that every infinite tournament has an infinite transitive subtournament. This principle plays an important role in the understanding of the computational strength of Ramsey's theorem for pairs (RT^2_2) by providing an alternate proof of RT^2_2 in terms of EM and the ascending descending sequence principle (ADS). In this paper, we study the computational weakness of EM and construct a standard model (ω-model) of simultaneously EM, weak König's lemma and the cohesiveness principle, which is not a model of the atomic model theorem. This separation answers a question of Hirschfeldt, Shore and Slaman, and shows that the weakness of the Erdös-Moser theorem goes beyond the separation of EM from ADS proven by Lerman, Solomon and Towsner.

1. Introduction

Reverse mathematics is a mathematical program whose goal is to classify theorems in terms of their provability strength. It uses the framework of subsystems of second-order arithmetic, with the base theory RCA_0, standing for Recursive Comprehension Axiom. RCA_0 is composed of the basic first-order Peano axioms, together with Δ^0_1-comprehension and Σ^0_1-induction schemes. RCA_0 is usually thought of as capturing *computational mathematics*. This program led to two important observations: First, most "ordinary" (i.e. non set-theoreric) theorems require only very weak set existence axioms. Second, many of those theorems are actually *equivalent* to one of five main subsystems over RCA_0, known as the "Big Five" [21].

However, Ramsey theory is known to provide a large class of theorems escaping this phenomenon. Indeed, consequences of Ramsey's theorem for pairs (RT^2_2) usually belong to their own subsystem. Therefore, they received a lot of attention from the reverse mathematics community [3, 13, 14, 27]. This article focuses on Ramseyan principles below the arithmetic comprehension axiom (ACA). See Soare [29] for a general introduction to computability theory, and Hirschfeldt [11] for a good introduction to the reverse mathematics below ACA.

1.1. Cohesiveness

Cohesiveness is a statement playing a central role in the analysis of Ramsey's theorem for pairs [3]. It can be seen as a sequential version of Ramsey's theorem for singletons and admits characterizations in terms of degrees whose jump computes a path through a $\Pi^{0,\emptyset'}_1$ class [16]. The decomposition of RT^2_2 in terms of COH and stable Ramsey's theorem for pairs (SRT^2_2) has been reused in the analysis of many consequences of Ramsey's theorem [14]. The link between cohesiveness and SRT^2_2 is still an active research subject [4, 8, 12, 26].

Definition 1.1 (Cohesiveness) An infinite set C is $\vec{R}$-cohesive for a sequence of sets $\vec{R} = R_0, R_1, \dots$ if for each $i \in \omega$, $C \subseteq^* R_i$ or $C \subseteq^* \overline{R_i}$. A set C is *p-cohesive* if it is $\vec{R}$-cohesive where $\vec{R}$ is an enumeration of all primitive recursive sets. COH is the statement "Every uniform sequence of sets $\vec{R}$ has an $\vec{R}$-cohesive set."

Jockusch and Stephan [16] studied the degrees of unsolvability of cohesiveness and proved that COH admits a universal instance whose solutions are the p-cohesive sets. They characterized their degrees as those whose jump is PA relative to $\emptyset'$. The author extended this analysis to every computable instance of COH and studied their degrees of unsolvability [26]. Cholak, Jockush and Slaman [3] proved that RT^2_2 is computably equivalent to $\mathsf{SRT}^2_2 + \mathsf{COH}$. Mileti [20] and Jockusch and Lempp [unpublished] formalized this equivalence over RCA_0. Hirschfeldt, Jockusch, Kjos-Hanssen, Lempp and Slaman [13] proved that COH contains a model with no diagonally non-computable function, thus COH does not imply SRT^2_2 over RCA_0. Cooper [6] proved that every degree above $\mathbf{0}'$ is the jump of a minimal degree. Therefore there exists a p-cohesive set of minimal degree.

1.2. The Erdős-Moser theorem

The Erdős-Moser theorem is a principle coming from graph theory. It provides together with the ascending descending principle (ADS) an alternative proof of Ramsey's theorem for pairs (RT^2_2). Indeed, every coloring $f : [\omega]^2 \to 2$ can be seen as a tournament R such that $R(x, y)$ holds if $x < y$ and $f(x, y) = 1$, or $x > y$ and $f(y, x) = 0$. Every infinite transitive subtournament induces a linear order whose infinite ascending or descending sequences are homogeneous for f.

Definition 1.2 (Erdős-Moser theorem) A tournament T on a domain $D \subseteq \omega$ is an irreflexive binary relation on D such that for all $x, y \in D$ with $x \neq y$, exactly one of $T(x, y)$ or $T(y, x)$ holds. A tournament T is *transitive* if the corresponding relation T is transitive in the usual sense. A tournament T is *stable* if $(\forall x \in D)[(\forall^\infty s)T(x, s) \vee (\forall^\infty s)T(s, x)]$. EM is the statement "Every infinite tournament T has an infinite transitive subtournament." SEM is the restriction of EM to stable tournaments.

Bovykin and Weiermann [1] introduced the Erdős-Moser theorem in reverse mathematics and proved that EM together with the chain-antichain principle (CAC) is equivalent to RT^2_2 over RCA_0. This was refined into an equivalence between $\mathsf{EM} + \mathsf{ADS}$ and RT^2_2 by Montalbán (see [1]), and the equivalence still holds between the stable versions of the statements. Lerman, Solomon and Towsner [19] proved that EM is strictly weaker than RT^2_2 by constructing an ω-model of EM which is not a model of the stable ascending descending sequence (SADS). SADS is the restriction of ADS to linear orders of order type $\omega + \omega^*$ [14]. The author noticed in [22] that their construction can be adapted to obtain a separation of EM from the stable thin set theorem for pairs ($\mathsf{STS}(2)$). Wang strengthened this separation by constructing in [31] a standard model of many theorems, including EM, COH and weak König's lemma (WKL) which is neither a model of $\mathsf{STS}(2)$ nor a model of SADS. The author later refined in [24, 26] the forcing technique of Lerman, Solomon and Towsner and showed that it is strong enough to obtain the same separations as Wang.

On the lower bounds side, Lerman, Solomon and Towsner [19] showed that EM implies the omitting partial types principle (OPT) over $\mathsf{RCA}_0 + \mathsf{B}\Sigma^0_2$ and Kreuzer proved in [18] that SEM implies $\mathsf{B}\Sigma^0_2$ over RCA_0. The statement OPT can be thought of as a stating for every set X the existence of a set hyperimmune relative to X. Finally, the author proved in [25] that $\mathsf{RCA}_0 \vdash \mathsf{EM} \to [\mathsf{STS}(2) \vee \mathsf{COH}]$. In particular, every model of EM which is not a model of $\mathsf{STS}(2)$ is also a model of COH. This fact will be reused in this paper since $\mathsf{STS}(2)$ implies the atomic model theorem over RCA_0 [25].

1.3. Domination and the atomic model theorem

The atomic model theorem is a statement coming from model theory. It has been introduced by Hirschfeldt, Shore and Slaman [15] in the settings of reverse mathematics.

Definition 1.3 (Atomic model theorem) A formula $\varphi(x_1, \dots, x_n)$ of T is an *atom* of a theory T if for each formula $\psi(x_1, \dots, x_n)$ we have $T \vdash \varphi \to \psi$ or $T \vdash \varphi \to \neg\psi$ but not both. A theory T is *atomic* if, for every formula $\psi(x_1, \dots, x_n)$ consistent with T, there is an atom $\varphi(x_1, \dots, x_n)$ of T extending it, i.e., one such that $T \vdash \varphi \to \psi$. A model $\mathcal{A}$ of T is *atomic* if every n-tuple from $\mathcal{A}$ satisfies an atom of T. AMT is the statement "Every complete atomic theory has an atomic model".

This strength of the atomic model theorem received a lot of attention from the reverse mathematics community and was subject to many refinements. On the upper bound side, Hirschfeldt, Shore and Slaman [15] proved that AMT is a consequence of SADS over RCA_0. The author [25] proved that the stable thin set theorem for pairs ($\mathsf{STS}(2)$) implies AMT over RCA_0.

On the lower bound side, Hirschfeldt, Shore and Slaman [15] proved that AMT implies the omitting partial type theorem (OPT) over RCA_0. Hirschfeldt and Greenberg, and independently Day, Dzhafarov and Miller, strengthened this result by proving that AMT implies the finite intersection property (FIP) over RCA_0 (see [11]). The principle FIP was first introduced by Dzhafarov and Mummert [9]. Later, Downey, Diamondstone, Greenberg and Turetsky [7] and Cholak, Downey and Igusa [2] proved that FIP is equivalent to the principle asserting, for every set X, the existence of a 1-generic relative to X. In particular, every model of AMT contains 1-generic reals.

The computable analysis of the atomic model theorem revealed the genericity flavor of the statement. More precisely, the atomic model theorem admits a pure computability-theoretic characterization in terms of hyperimmunity relative to a fixed Δ^0_2 function.

Definition 1.4 (Escape property) For every Δ^0_2 function f, there exists a function g such that $f(x) < g(x)$ for infinitely many x.

The escape property is a statement in between hyperimmunity relative to $\emptyset'$ and hyperimmunity. The atomic model theorem is computably equivalent to the escape property, that is, for every complete atomic theory T, there is a $\Delta^{0,T}_2$ function f such that for every function g satisfying the escape property for f, $T \oplus g$ computes an atomic model of T. Conversely, for every Δ^0_2 approximation $\tilde{f}$ of a function f, there is a $\tilde{f}$-computable complete atomic theory such that for every atomic model $\mathcal{M}$, $\tilde{f} \oplus \mathcal{M}$ computes a function satisfying the escape property for f. In particular, the ω-models satisfying AMT are exactly the ones satisfying the escape property. However the formalization of this equivalence requires more than the Σ^0_1 induction scheme. It was proven to hold over $\mathsf{RCA}_0 + \mathsf{I}\Sigma^0_2$ but not $\mathsf{RCA}_0 + \mathsf{B}\Sigma^0_2$ [15, 5], where $\mathsf{I}\Sigma^0_2$ and $\mathsf{B}\Sigma^0_2$ are respectively the Σ^0_2 induction scheme and the Σ^0_2 bounding scheme.

Hirschfeldt, Shore and Slaman [15] asked the following question.

Question 1.5 Does the cohesiveness principle imply the atomic model theorem over RCA_0?

Note that AMT is not computably reducible to COH, since there exists a cohesive set of minimal degree [6], and a computable atomic theory whose computable atomic models bound 1-generic reals [11], but no minimal degree bounds a 1-generic real [32].

In this paper, we answer this question negatively. We shall take advantage of the characterization of AMT by the escape property to create an ω-model $\mathcal{M}$ of EM, WKL and COH simultaneously, together with a Δ^0_2 function f dominating every function in $\mathcal{M}$. Therefore, any Δ^0_2 approximation $\tilde{f}$ of the function f is a computable instance of the escape property belonging to $\mathcal{M}$, but with no solution in $\mathcal{M}$. The function f witnesses in particular that $\mathcal{M} \not\models \mathsf{AMT}$. Our main theorem is the following.

Theorem 1.6 (Main theorem) $\mathsf{COH} \wedge \mathsf{EM} \wedge \mathsf{WKL}$ does not imply AMT over RCA_0.

The proof techniques used to prove the main theorem will be introduced progressively by considering first computable non-reducibility, and then generalizing the diagonalization to Turing ideals by using an effective iterative forcing.

1.4. Definitions and notation

String, sequence. Fix an integer $k \in \omega$. A *string* (over k) is an ordered tuple of integers $a_0, \dots, a_{n-1}$ (such that $a_i < k$ for every $i < n$). The empty string is written ε. A *sequence* (over k) is an infinite listing of integers $a_0, a_1, \dots$ (such that $a_i < k$ for every $i \in \omega$). Given $s \in \omega$, k^s is the set of strings of length s over k and $k^{<s}$ is the set of strings of length $< s$ over k. Similarly, $k^{<\omega}$ is the set of finite strings over k and k^ω is the set of sequences (i.e. infinite strings) over k. Given a string $\sigma \in k^{<\omega}$, we denote by $|\sigma|$ its length. Given two strings $\sigma, \tau \in k^{<\omega}$, σ is a *prefix* of τ (written $\sigma \preceq \tau$) if there exists a string $\rho \in k^{<\omega}$ such that $\sigma\rho = \tau$. Given a sequence X, we write $\sigma \prec X$ if $\sigma = X \restriction n$ for some $n \in \omega$, where $X \restriction n$ denotes the restriction of X to its first

n elements. A *binary string* is a *string* over 2. A *real* is a sequence over 2. We may identify a real with a set of integers by considering that the real is its characteristic function.

Tree, path. A tree $T \subseteq k^{<\omega}$ is a set downward-closed under the prefix relation. A *binary* tree is a tree $T \subseteq 2^{<\omega}$. A sequence $P \in k^\omega$ is a *path* though T if for every $\sigma \prec P$, $\sigma \in T$. A string $\sigma \in k^{<\omega}$ is a *stem* of a tree T if every $\tau \in T$ is comparable with σ. Given a tree T and a string $\sigma \in T$, we denote by $T^{[\sigma]}$ the subtree $\{\tau \in T : \tau \preceq \sigma \vee \tau \succeq \sigma\}$.

Sets, partitions. Given two sets A and B, we denote by $A < B$ the formula $(\forall x \in A)(\forall y \in B)[x < y]$ and by $A \subseteq^* B$ the formula $(\forall^\infty x \in A)[x \in B]$, meaning that A is included in B *up to finitely many elements.* Given a set X and some integer k, a *k-cover of X* is a k-uple $A_0, \dots, A_{k-1}$ such that $A_0 \cup \dots \cup A_{k-1} = X$. We may simply say *$k$-cover* when the set X is unambiguous. A *k-partition* is a k-cover whose sets are pairwise disjoint. A *Mathias condition* is a pair (F, X) where F is a finite set, X is an infinite set and $F < X$. A condition (F_1, X_1) *extends* (F, X) (written $(F_1, X_1) \leq (F, X)$) if $F \subseteq F_1$, $X_1 \subseteq X$ and $F_1 \setminus F \subset X$. A set G *satisfies* a Mathias condition (F, X) if $F \subset G$ and $G \setminus F \subseteq X$. We refer the reader to Chapter 2 in Hirschfeldt [11] for a gentle introduction to effective forcing.

2. The weakness of cohesiveness under computable reducibility

Before proving that COH does not imply AMT over RCA_0, we illustrate the key features of our construction by showing that AMT does not reduce to COH in one step. This one-step reducibility is known as *computable reducibility* [8, 12, 26]. The general construction will consist of iterating this one-step diagonalization to construct a Turing ideal whose functions are dominated by a single Δ^0_2 function.

Definition 2.1 (Computable reducibility) A principle P is *computably reducible* to another principle Q (written $\mathsf{P} \leq_c \mathsf{Q}$) if every P-instance I computes a Q-instance J such that for every solution X to J, $X \oplus I$ computes a solution to I.

The remainder of this section is devoted to the proof of the following theorem.

Theorem 2.2 $\mathsf{AMT} \not\leq_c \mathsf{COH}$

In order to prove Theorem 2.2, we need to construct a Δ^0_2 function f such that for every uniformly computable sequence of sets $\vec{R} = R_0, R_1, \dots$, there is an $\vec{R}$-cohesive set G such that every G-computable function is dominated by f. Thankfully, Jockusch and Stephan [16] proved that for every such sequence of sets $\vec{R}$, every p-cohesive set computes an infinite $\vec{R}$-cohesive set. The sequence of all primitive recursive sets is therefore called a *universal instance.* Hence we only need to build a Δ^0_2 function f and a p-cohesive set G such that every G-computable function is dominated by f to obtain Theorem 2.2.

Given some uniformly computable sequence of sets $\vec{R} = R_0, R_1, \dots$, the usual construction of an $\vec{R}$-cohesive set G is done by a computable Mathias forcing. The forcing conditions are pairs (F, X), where F is a finite set representing the finite approximation of G and X is an infinite, computable reservoir such that $max(F) < min(X)$. The construction of the $\vec{R}$-cohesive set is obtained by building an infinite, decreasing sequence of Mathias conditions, starting with $(\emptyset, \omega)$ and interleaving two kinds of steps. Given some condition (F, X),

- (S1) the *extension* step consists of taking an element x from X and adding it to F, thereby forming the extension $(F \cup \{x\}, X \setminus [0, x])$;
- (S2) the *cohesiveness* step consists of deciding which one of $X \cap R_i$ and $X \cap \overline{R}_i$ is infinite, and taking the chosen one as the new reservoir.

The first step ensures that the constructed set G will be infinite, whereas the second step makes the set G $\vec{R}$-cohesive. Looking at the effectiveness of the construction, the step (S1) is computable, assuming we are given some Turing index of the set X. The step (S2), on the other hand, requires to decide which one of two computable sets is infinite, knowing that at least one of them is. This decision requires the computational power of a PA degree relative to $\emptyset'$ (see [3, Lemma 4.2]). Since we want to build a Δ^0_2 function f dominating every G-computable function, we would like to make the construction of G Δ^0_2. Therefore the step (S2) has to be revised.

2.1. **Effectively constructing a cohesive set**

The above construction leads to two observations. First, at any stage of the construction, the reservoir X of the Mathias condition (F, X) has a particular shape. Indeed, after the first application of stage (S2), the set X is, up to finite changes, of the form $\omega \cap R_0$ or $\omega \cap \overline{R_0}$. After the second application of (S2), it is in one of the following forms: $\omega \cap R_0 \cap R_1$, $\omega \cap R_0 \cap \overline{R}_1$, $\omega \cap \overline{R}_0 \cap R_1$, $\omega \cap \overline{R}_0 \cap \overline{R}_1$, and so on. More generally, given some string $\sigma \in 2^{<\omega}$, we can define R_σ inductively as follows: First, $R_\varepsilon = \omega$, and then, if R_σ has already been defined for some string σ of length i, $R_{\sigma 0} = R_\sigma \cap \overline{R}_i$ and $R_{\sigma 1} = R_\sigma \cap R_i$. By the first observation, we can replace Mathias conditions by pairs (F, σ), where F is a finite set and $\sigma \in 2^{<\omega}$. The pair (F, σ) denotes the Mathias condition $(F, R_\sigma \setminus [0, max(F)])$. A pair (F, σ) is *valid* if R_σ is infinite. The step (S2) can be reformulated as choosing, given some valid condition (F, σ), which one of $(F, \sigma 0)$ and $(F, \sigma 1)$ is valid.

Second, we do not actually need to decide which one of $R_{\sigma 0}$ and $R_{\sigma 1}$ is infinite assuming that R_σ is infinite. Our goal is to dominate every G-computable function with a Δ^0_2 function f. Therefore, given some G-computable function g, it is sufficient to find a finite set S of candidate values for $g(x)$ and make $f(x)$ be greater than the maximum of S. Instead of choosing which one of $R_{\sigma 0}$ and $R_{\sigma 1}$ is infinite, we will explore both cases in parallel. The step (S2) will split some condition (F, σ) into two conditions $(F, \sigma 0)$ and $(F, \sigma 1)$. Our new forcing conditions are therefore tuples $(F_\sigma : \sigma \in 2^n)$ which have to be thought of as 2^n parallel Mathias conditions (F_σ, σ) for each $\sigma \in 2^n$. Note that (F_σ, σ) may not denote a valid Mathias condition in general since R_σ may be finite. Therefore, the step (S1) becomes Δ^0_2, since we first have to check whether R_σ is non-empty before picking an element in R_σ. The whole construction is Δ^0_2 and yields a Δ^0_2 infinite binary tree T. In particular, any degree PA relative to $\emptyset'$ bounds an infinite path though T and therefore bounds a G-cohesive set. However, the degree of the set G is not sensitive in our argument. We only care about the effectiveness of the tree T.

2.2. **Dominating the functions computed by a cohesive set**

We have seen in the previous section how to make the construction of a cohesive set more effective by postponing the choices between forcing $G \subseteq^* R_i$ and $G \subseteq^* \overline{R}_i$ to the end of the construction. We now show how to dominate every G-computable function for every infinite path G through the Δ^0_2 tree constructed in the previous section. To do this, we will interleave a third step deciding whether $\Phi^G_e(n)$ halts, and if so, collecting the candidate values of $\Phi^G_e(n)$. Given some Mathias precondition (F, X) (a precondition is a condition where we do not assume that the reservoir is infinite) and some $e, x \in \omega$, we can Δ^0_2-decide whether there is some set $E \subseteq X$ such that $\Phi^{F \cup E}_e(x) \downarrow$. If this is the case, then we can effectively find this a finite set $E \subseteq X$ and compute the value $\Phi^{F \cup E}_e(x)$. If this is not the case, then for every infinite set G satisfying the condition (F, X), the function Φ^G_e will not be defined on input x. In this case, our goal is vacuously satisfied since Φ^G_e will not be a function and therefore we do not need do dominate Φ^G_e. Let us go back to the previous construction. After some stage, we have constructed a condition $(F_\sigma : \sigma \in 2^n)$ inducing a finite tree of depth n. The step (S3) acts as follows for some $x \in \omega$:

(S3) Let $S = \{0\}$. For each $\sigma \in 2^n$ and each $e \leq x$, decide whether there is some finite set $E \subseteq R_\sigma \setminus [0, max(F_\sigma)]$ such that $\Phi^{F_\sigma \cup E}_e(x) \downarrow$. If this is the case, add the value of $\Phi^{F_\sigma \cup E}_e(x)$ to S and set $\tilde{F}_\sigma = F_\sigma \cup E$, otherwise set $\tilde{F}_\sigma = F_\sigma$. Finally, set $f(x) = max(S) + 1$ and take $(\tilde{F}_\sigma : \sigma \in 2^n)$ as the next condition.

Note that the step (S3) is Δ^0_2-computable uniformly in the condition $(F_\sigma : \sigma \in 2^n)$. The whole construction therefore remains Δ^0_2 and so does the function f. Moreover, given some G-computable function g, there is some Turing index e such that $\Phi^G_e = g$. For each $x \geq e$, the step (S3) is applied at a finite stage and decides whether $\Phi^G_e(x)$ halts or not for every set satisfying one of the leaves of the finite tree. In particular, this is the case for the set G and therefore $\Phi_e^G(x) \in S$. By definition of f, $f(x) \geq max(S) \geq \Phi_e^G(x)$. Therefore f dominates the function g.

2.3. The formal construction

Let $\vec{R} = R_0, R_1, \dots$ be the sequence of all primitive recursive sets. We define a Δ^0_2 decreasing sequence of conditions $(\emptyset, \varepsilon) \geq c_0 \geq c_1 \dots$ such that for each $s \in \omega$

(i) $c_s = (F^s_\sigma : \sigma \in 2^s)$ and $|F^s_\sigma| \geq s$ if $R_\sigma \setminus [0, max(F^s_\sigma)] \neq \emptyset$.

(ii) For every $e \leq s$ and every $\sigma \in 2^s$, either $\Phi_e^{F^s_\sigma}(s) \downarrow$ or $\Phi_e^G(s) \uparrow$ for every set G satisfying (F^s_σ, R_σ).

Let P be a path through the tree $T = \{\sigma \in 2^{<\omega} : R_\sigma \text{ is infinite}\}$ and let $G = \bigcup_s F^s_{P \restriction s}$. By (i), for each $s \in \omega$, $|F^s_{P\restriction s}| \geq s$ since $R_{P\restriction s}$ is infinite. Therefore the set G is infinite. Moreover, for each $s \in \omega$, the set G satisfies the condition $(F^{s+1}_{P\restriction s+1}, R_{P\restriction s+1})$, so $G \subseteq^* R_{P\restriction s+1} \subseteq R_s$ if $P(s) = 1$ and $G \subseteq^* R_{P\restriction s+1} \subseteq \overline{R}_s$ if $P(s) = 0$. Therefore G is $\vec{R}$-cohesive.

For each $s \in \omega$, let $f(s) = 1 + max(\Phi_e^{F^s_\sigma}(s) : \sigma \in 2^s, e \leq s)$. The function f is Δ^0_2. We claim that it dominates every G-computable function. Fix some e such that Φ_e^G is total. For every $s \geq e$, let $\sigma = P \restriction s$. By (ii), either $\Phi_e^{F^s_\sigma}(s) \downarrow$ or $\Phi_e^G(s) \uparrow$ for every set G satisfying (F^s_σ, R_σ). Since $\Phi_e^G(s) \downarrow$, the first case holds. By definition of f, $f(s) \geq \Phi_e^{F^s_\sigma}(s) = \Phi_e^G(s)$. Therefore f dominates the function Φ_e^G. This completes the proof of Theorem 2.2.

3. The weakness of EM under computable reducibility

We now strengthen the analysis of the previous section by proving that the atomic model theorem is not computably reducible to the Erdős-Moser theorem. Theorem 2.2 is an immediate consequence of this result since $[\mathsf{AMT} \vee \mathsf{COH}] \leq_c \mathsf{EM}$ (see [25]). After this section, we will be ready to iterate the construction in order to build an ω-model of $\mathsf{EM} \wedge \mathsf{COH}$ which is not a model of AMT.

Theorem 3.1 $\mathsf{AMT} \not\leq_c \mathsf{EM}$

Before proving Theorem 3.1, we start with an analysis of the combinatorics of the Erdős-Moser theorem. Just as we did for cohesiveness, we will show how to build solutions to EM through Δ^0_2 constructions, postponing the Π^0_2 choices to the end.

3.1. The combinatorics of the Erdős-Moser theorem

The standard way of building an infinite object by forcing consists of defining an increasing sequence of finite approximations, and taking the union of them. Unlike COH where every finite set can be extended to an infinite cohesive set, some finite transitive subtournaments may not be extensible to an infinite one. We therefore need to maintain some extra properties which will guarantee that the finite approximations are extendible. The nature of these properties constitue the core of the combinatorics of EM.

Lerman, Solomon and Towsner [19] proceeded to an analysis of the Erdős-Moser theorem. They showed in particular that it suffices to ensure that the finite transitive subtournament F has infinitely many *one-point extensions*, that is, infinitely many elements x such that $F \cup \{x\}$ is transitive, to extend F to an infinite transitive subtournament (see [19, Lemma 3.4]). This property is sufficient to add elements one by one to the finite approximation. However, when adding elements by block, we shall maintain a stronger invariant. We will require that the reservoir is included in a minimal interval of the finite approximation F. In this section, we reintroduce the terminology of Lerman, Solomon and Towsner [19] and give a presentation of the combinatorics of the Erdős-Moser theorem motivated by its computational analysis.

Definition 3.2 (Minimal interval) Let R be an infinite tournament and $a, b \in R$ be such that $R(a, b)$ holds. The *interval* (a, b) is the set of all $x \in R$ such that $R(a, x)$ and $R(x, b)$ hold. Let $F \subseteq R$ be a finite transitive subtournament of R. For $a, b \in F$ such that $R(a, b)$ holds, we say that (a, b) is a *minimal interval of* F if there is no $c \in F \cap (a, b)$, i.e., no $c \in F$ such that $R(a, c)$ and $R(c, b)$ both hold.

Fix a computable tournament R, and consider a pair (F, X) where

(i) F is a finite R-transitive set representing the *finite approximation* of the infinite R-transitive subtournament we want to construct

(ii) X is an infinite set disjoint from F, included in a minimal interval of F and such that $F \cup \{x\}$ is R-transitive for every $x \in X$. In other words, X is an infinite set of one-point extensions. Such a set X represents the *reservoir*, that is, a set of candidate elements we may add to F later on.

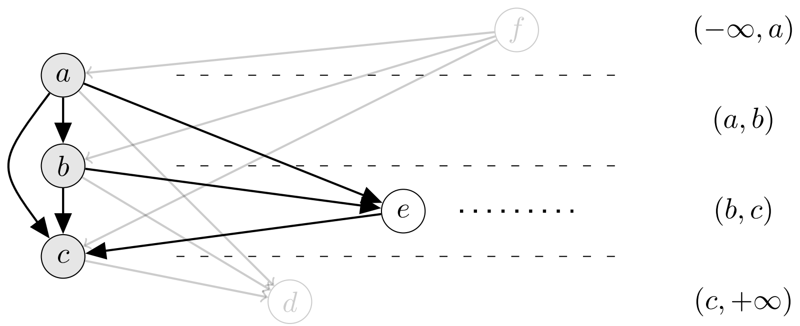

FIGURE 1. In this figure, $F = \{a, b, c\}$ is a transitive set, $X = \{d, e, f, \dots\}$ a set of one-point extensions, $(b, c) = \{e, \dots\}$ a minimal interval of F and $(F, X \cap (b, c))$ an EM condition. The elements d and f are not part of the minimal interval (b, c).

The infinite set X ensures extensibility of the finite set F into an infinite R-transitive subtournament. Indeed, by applying the Erdős-Moser theorem to R over the domain X, there exists an infinite R-transitive subtournament $H \subseteq X$. One easily checks that $F \cup H$ is R-transitive. The pair (F, X) is called an Erdős-Moser condition in [23]. A set G *satisfies* an EM condition (F, X) if it is R-transitive and satisfies the Mathias condition (F, X). In order to simplify notation, given a tournament R and two sets E and F, we denote by $E \to_R F$ the formula $(\forall x \in E)(\forall y \in F)R(x, y)$.

Suppose now that we want to add a finite number of elements of X into F to obtain a finite T-transitive set $\tilde{F} \supseteq F$, and find an infinite subset $\tilde{X} \subseteq X$ such that $(\tilde{F}, \tilde{X})$ has the above mentioned properties. We can do this in a few steps:

1. Choose a finite (not necessarily R-transitive) set $E \subset X$.
2. Any element $x \in X \setminus E$ induces a 2-partition $\langle E_0, E_1 \rangle$ of E by setting $E_0 = \{y \in E : R(y, x)\}$ and $E_1 = \{y \in E : R(x, y)\}$. Consider the coloring f which associates to any element of $X \setminus E$ the corresponding 2-partition $\langle E_0, E_1 \rangle$ of E.
3. As E is finite, there exists finitely many 2-partitions of E, so f colors each element of $X \setminus E$ into finitely many colors. By Ramsey's theorem for singletons applied to f, there exists a 2-partition $\langle E_0, E_1 \rangle$ of E together with an infinite subset $\tilde{X} \subseteq X \setminus E$ such that for every $x \in \tilde{X}$, $f(x) = \langle E_0, E_1 \rangle$. By definition of f and E_i, $E_0 \to_R \tilde{X} \to_R E_1$.
4. Take any R-transitive subset $F_1 \subseteq E_i$ for some $i < 2$ and set $\tilde{F} = F \cup F_1$. The pair $(\tilde{F}, \tilde{Y})$ satisfies the required properties (see [23, Lemma 5.9] for a proof).

From a computational point of view, if we start with a computable condition (F, X), that is, where X is a computable set, we end up with a computable extension $(\tilde{F}, \tilde{Y})$. Remember that our goal is to define a Δ^0_2 function f which will dominate every G-computable function for some solution G to R. For this, we need to be able to $\emptyset'$-decide whether $\Phi^G_e(n) \downarrow$ or $\Phi^G_e(n) \uparrow$ for every solution G to R satisfying some condition (F, X). More generally, given some Σ^0_1 formula φ, we focus on the computational power required to decide a question of the form

Q1: Is there an R-transitive extension $\tilde{F}$ of F in X such that $\varphi(\tilde{F})$ holds?

Trying to apply naively the algorithm above requires a lot of computational power. In particular, step 3 requires to choose a true formula among finitely many $\Pi^{0,X}_2$ formulas. Such a step needs the power of PA degree relative to the jump of X. We shall apply the same trick as for cohesiveness, consisting in not trying to choose a true $\Pi^{0,X}_2$ formula, but instead parallelizing the construction. Given a finite set $E \subset X$, instead of finding an infinite subset $\tilde{Y} \subset X \setminus E$ whose members induce a 2-partition of E, we will construct as many extensions of (F, X) as there are 2-partitions of E. The question now becomes

Q2: Is there a finite set $E \subseteq X$ such that for every 2-partition $\langle E_0, E_1\rangle$ of E, there exists an R-transitive subset $F_1 \subseteq E_i$ for some $i < 2$ such that $\varphi(F \cup F_1)$ holds?

This question is $\Sigma^{0,X}_1$, which is good enough for our purposes. If the answer is positive, we will try the witness F_1 associated to each 2-partition of E in parallel. Note that there may be some 2-partition $\langle E_0, E_1\rangle$ of E such that the set $Y = \{x \in X \setminus E : E_0 \to_R \{x\} \to_R E_1\}$ is finite, but this is not a problem since there is *at least* one good 2-partition such that the corresponding set is infinite. The whole construction yields again a tree of pairs (F, X).

If the answer is negative, we want to ensure that $\varphi(\tilde{F})$ will not hold at any further stage of the construction. For each $n \in \omega$, let H_n be the set of the n first elements of X. Because the answer is negative, for each $n \in \omega$, there exists a 2-partition $\langle E_0, E_1\rangle$ of H_n such that for every R-transitive subset $F_1 \subseteq E_i$ for any $i < 2$, $\varphi(F \cup F_1)$ does not hold. Call such a 2-partition an *avoiding* partition of H_n. Note that if $\langle E_0, E_1\rangle$ is an avoiding partition of H_{n+1}, then $\langle E_0 {\upharpoonright} n, E_1 {\upharpoonright} n\rangle$ is an avoiding partition of H_n. So the set of avoiding 2-partitions of some H_n forms an infinite tree T. Moreover, the predicate "$\langle E_0, E_1\rangle$ is an avoiding partition of H_n" is Δ^{0,H_n}_1 so the tree T is $\Delta^{0,X}_1$. The collection of the infinite paths through T forms a non-empty $\Pi^{0,X}_1$ class $\mathcal{C}$ defined as the collection of 2-partitions $Z_0 \cup Z_1 = X$ such that for every $i < 2$ and every R-transitive subset $F_1 \subseteq Z_i$, $\varphi(F \cup F_1)$ does not hold.

The natural next step would be to apply weak König's lemma to obtain a 2-partition of X such that for every finite R-transitive subset F_1 of any of its parts, $\varphi(F \cup F_1)$ does not hold. By the low basis theorem, we could take the 2-partition to be low over X and the whole construction would remain Δ^0_2. However, when iterating the construction, we will be given only finite pieces of tournaments since the tournament may depend on an oracle being constructed at a previous iteration. In this setting, it will be impossible to compute a member of the $\Pi^{0,X}_1$ class $\mathcal{C}$ of 2-partitions, since we will have access to only a finite piece of the corresponding tree T. In order to get progressively prepared to the iterated forcing, we will not apply WKL and will work with Π^0_1 classes of 2-partitions. Therefore, if the answer is negative, we duplicate the finite R-transitive F into two sets $F_0 = F_1 = F$, and commit F_i to take from now on its next elements from X_i for some 2-partition $X_0 \cup X_1 = X$ belonging to the Π^0_1 class $\mathcal{C}$ of 2-partitions witnessing the negative answer. Iterating the process by asking several questions leads to tuples $(F_0, \dots, F_{k-1}, \mathcal{C})$ where F_i is a finite R-transitive set taking its elements from the ith part of the class $\mathcal{C}$ of k-partitions. This notion of forcing will be defined formally in a later section.

3.2. Enumerating the computable infinite tournaments

Proving that some principle P does not computably reduce to Q requires to create a P-instance X such that *every* X-computable Q-instance has a solution Y such that $Y \oplus X$ does not compute a solution to X. In the case of AMT $\not\leq_c$ COH, we have been able to restrict ourselves to only one instance of COH, since Jockusch and Stephan [16] showed it admits a universal instance. It is currently unknown whether the Erdős-Moser theorem admits a universal instance, that is, a computable infinite tournament such that for every infinite transitive subtournament H and for every computable infinite tournament T, H computes an infinite transitive T-subtournament. See [23] for an extensive study of the existence of universal instances for principles in reverse mathematics.

Since we do not know whether EM admits a universal instance, we will need to diagonalize against the solutions to every computable EM-instance. In fact, we will prove a stronger result. We will construct a Δ^0_2 function f and an infinite set G which is eventually transitive simultaneously for every computable infinite tournament, and such that f dominates every G-computable function. There exists no computable sequence of sets containing all computable sets. Therefore it is not possible to computably enumerate every infinite computable tournament. However, one can define an infinite, computable, binary tree such that every infinite path computes such a sequence. See the notion of sub-uniformity defined by Mileti in [20] for details. By the low basis theorem, there exists a low set bounding a sequence containing, among others, every infinite computable tournament. As we shall prove below, for every set C and every uniformly C-computable sequence of infinite tournaments $\vec{R}$, there exists a set G together with a $\Delta^{0,C}_2$ function f such that

(i) G is eventually R-transitive for every $R \in \vec{R}$
(ii) If $\Phi^{G \oplus C}_e$ is total, then it is dominated by f for every $e \in \omega$.

Thus it suffices to choose C to be our low set and $\vec{R}$ to be a uniformly C-computable sequence of infinite tournaments containing every computable tournament to deduce the existence of a set G together with a Δ^0_2 function f such that

(i) G is eventually R-transitive for every infinite, computable tournament R
(ii) If $\Phi^{G \oplus C}_e$ is total, then it is dominated by f for every $e \in \omega$

By the computable equivalence between AMT and the escape property, there exists a computable atomic theory T such that every atomic model computes a function g not dominated by f. If $\mathsf{AMT} \leq_c \mathsf{EM}$, then there exists an infinite, computable tournament R such that every infinite R-transitive subtournament computes a model of T, hence computes a function g not dominated by f. As the set G is, up to finite changes, an infinite R-transitive subtournament, G computes such a function g, contradicting our hypothesis. Therefore $\mathsf{AMT} \not\leq_c \mathsf{EM}$.

3.3. **Cover classes**

In this part, we introduce some terminology about classes of k-covers. Recall that a k-cover of some set X is a k-uple $A_0, \dots, A_{k-1}$ such that $A_0 \cup \dots \cup A_{k-1} = X$. In particular, the sets are not required to be pairwise disjoint.

Cover class. We identify a k-cover $Z_0 \cup \dots \cup Z_{k-1}$ of some set X with the k-fold join of its parts $Z = \bigoplus_{i<k} Z_i$, and refer this as a *code* for the cover. A *k-cover class* of some set X is a tuple $\langle k, X, \mathcal{C} \rangle$ where $\mathcal{C}$ is a collection of codes of k-covers of X. We will be interested in Π^0_1 k-cover classes. A *part* of a k-cover class $\langle k, X, \mathcal{C} \rangle$ is a number $\nu < k$. Informally, a part ν represents the collection of all Z_ν, where $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$. For the simplicity of notation, we may use the same letter $\mathcal{C}$ to denote both a k-cover class $(k, X, \mathcal{C})$ and the actual collection of k-covers $\mathcal{C}$. We then write $dom(\mathcal{C})$ for X and $parts(\mathcal{C})$ for k.

Restriction of a cover. Given some k-cover $Z = Z_0 \oplus \dots \oplus Z_{k-1}$ of some set X and given some set $Y \subseteq X$, we write $Z \upharpoonright Y$ for the k-cover $(Z_0 \cap Y) \oplus \dots \oplus (Z_{k-1} \cap Y)$ of Y. Similarly, given some cover class $(k, X, \mathcal{C})$ and some set $Y \subseteq X$, we denote by $\mathcal{C} \upharpoonright Y$ the cover class $(k, Y, \mathcal{D})$ where $\mathcal{D} = \{Z \upharpoonright Y : Z \in \mathcal{C}\}$. Given some part ν of $\mathcal{C}$ and some set E, we write $\mathcal{C}^{[\nu, E]}$ for the cover class $(k, X, \mathcal{D})$ where $\mathcal{D} = \{Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C} : E \subseteq Z_\nu\}$.

Refinement. The collection of cover classes can be given a natural partial order as follows. Let $m \geq k$ and $f : m \to k$. An m-cover $V_0 \oplus \dots \oplus V_{m-1}$ of Y *f-refines* a k-cover $Z_0 \oplus \dots \oplus Z_{k-1}$ of X if $Y \subseteq X$ and $V_\nu \subseteq Z_{f(\nu)}$ for each $\nu < m$. Given two cover classes $(k, X, \mathcal{C})$ and $(m, Y, \mathcal{D})$ and some function $f : m \to k$, we say that $\mathcal{D}$ *f-refines* $\mathcal{C}$ if for every $V \in \mathcal{D}$, there is some $Z \in \mathcal{C}$ such that V f-refines Z. In this case, we say that *part ν of $\mathcal{D}$ refines part $f(\nu)$ of $\mathcal{C}$*.

Acceptable part. We say that part ν of $\mathcal{C}$ is *acceptable* if there exists some $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ such that Z_ν is infinite. Part ν of $\mathcal{C}$ is *empty* if for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, $Z_\nu = \emptyset$. Note that if $\mathcal{C}$ is non-empty and $dom(\mathcal{C})$ is infinite, then $\mathcal{C}$ has at least one acceptable part. Moreover, if $\mathcal{D} \leq_f \mathcal{C}$ and part ν of $\mathcal{D}$ is acceptable, then so is part $f(\nu)$ of $\mathcal{C}$. The converse does not hold in general.

3.4. **The forcing notion**

We now get into the core of our forcing argument by defining the forcing notion which will be used to build an infinite set eventually transitive for every infinite computable tournament.

Fix a set C and a uniformly C-computable sequence of infinite tournaments $R_0, R_1, \dots$ We construct our set G by a forcing whose conditions are tuples $(\alpha, \vec{F}, \mathcal{C})$ where

(a) $\mathcal{C}$ is a non-empty $\Pi^{0,C}_1$ k-cover class of $[t, +\infty)$ for some $k, t \in \omega$; $\alpha \in t^{<\omega}$
(b) $(F_\nu \setminus [0, \alpha(i))) \cup \{x\}$ is R_i-transitive for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, every $x \in Z_\nu$, every $i < |\alpha|$ and each $\nu < k$
(c) Z_ν is included in a minimal R_i-interval of $F_\nu \setminus [0, \alpha(i))$ for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, every $i < |\alpha|$ and each $\nu < k$.

A condition $(\beta, \vec{E}, \mathcal{D})$ *extends* $(\alpha, \vec{F}, \mathcal{C})$ (written $(\beta, \vec{E}, \mathcal{D}) \leq (\alpha, \vec{F}, \mathcal{C})$) if $\beta \succeq \alpha$ and there exists a function $f : parts(\mathcal{D}) \to parts(\mathcal{C})$ such that the following holds:

(i) $(E_\nu, dom(\mathcal{D}))$ Mathias extends $(F_{f(\nu)}, dom(\mathcal{C}))$ for each $\nu < parts(\mathcal{D})$
(ii) $\mathcal{D}$ f-refines $\mathcal{C}^{[f(\nu), E_\nu \setminus F_{f(\nu)}]}$ for each $\nu < parts(\mathcal{D})$

One may think of a condition $(\alpha, \vec{F}, \mathcal{C})$ with, say, $parts(\mathcal{C}) = k$, as k parallel Mathias conditions which are, up to finite changes, Erdős-Moser conditions simultaneously for the tournaments $R_0, \dots, R_{|\alpha|-1}$. Given some $i < |\alpha|$, the value $\alpha(i)$ indicates at which point the sets $\vec{F}$ start being R_i-transitive. More precisely, for every part $\nu < k$ and every k-cover $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, $(F_\nu \setminus [0, \alpha(i)), Z_\nu)$ is an Erdős-Moser condition for R_i for each $i < |\alpha|$. Indeed, because of clause (i), the elements $E_\nu \setminus F_{f(\nu)}$ added to E_ν come from $dom(\mathcal{C})$ and because of clause (ii), these elements must come from the part $f(\nu)$ of the class $\mathcal{C}$, otherwise $\mathcal{C}^{[f(\nu), E_\nu \setminus F_{f(\nu)}]}$ would be empty and so would be $\mathcal{D}$.

Of course, there may be some parts ν of $\mathcal{C}$ which are non-acceptable, that is, such that Z_ν is finite for every k-cover $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$. However, by the infinite pigeonhole principle, Z_ν must be infinite for at least one $\nu < k$. Choosing α to be in $t^{<\omega}$ instead of $\omega^{<\omega}$ ensures that all elements added to $\vec{F}$ will have to be R_i-transitive simultaneously for each $i < |\alpha|$, as the elements are taken from $dom(\mathcal{C})$ and therefore are greater than the threshold $\alpha(i)$ for each $i < |\alpha|$. A *part* of a condition $c = (\alpha, \vec{F}, \mathcal{C})$ is a pair $\langle c, \nu \rangle$, where $\nu < parts(\mathcal{C})$. For the simplicity of notation, we may identify a part $\langle c, \nu \rangle$ of a condition with the part ν of the corresponding cover class $\mathcal{C}$. It must however be clear that a part depends on the condition c.

We start with a few basic lemmas reflecting the combinatorics described in the subsection 3.1. They are directly adapted from the basic properties of an Erdős-Moser condition proven in [23]. The first lemma states that each element of the finite transitive tournaments $\vec{F}$ behaves uniformly with respect to the elements of the reservoir, that is, is beaten by every element of the reservoir or beats all of them.

Lemma 3.3 For every condition $c = (\alpha, \vec{F}, \mathcal{C})$, every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, every part ν of $\mathcal{C}$, every $i < |\alpha|$ and every $x \in F_\nu \setminus [0, \alpha(i))$, either $\{x\} \to_{R_i} Z_\nu$ or $Z_\nu \to_{R_i} \{x\}$.

Proof. By property (c) of the condition c, there exists a minimal R_i-interval (u, v) of $F_\nu \setminus [0, \alpha(i))$ containing Z_ν. Here, u and v may be respectively $-\infty$ and $+\infty$. By definition of an interval, $\{u\} \to_{R_i} Z_\nu \to_{R_i} \{v\}$. By definition of a minimal interval, $R_i(x, u)$ or $R_i(v, x)$ holds. Suppose the former holds. By transitivity of $F_\nu \setminus [0, \alpha(i))$, for every $y \in Z_\nu$, $R_i(x, y)$ holds, since both $R_i(x, u)$ and $R_i(u, y)$ hold. Therefore $\{x\} \to_{R_i} Z_\nu$. In the latter case, by symmetry, $Z_\nu \to_{R_i} \{x\}$. □

The second lemma is the core of the combinatorics of the Erdős-Moser theorem. It provides sufficient properties to obtain a valid extension of a condition. Properties (i) and (ii) are simply the definition of an extension. Properties (iii) and (iv) help to propagate properties (b) and (c) from a condition to its extension. We shall see empirically that properties (iii) and (iv) are simpler to check than (b) and (c), as the former properties match exactly the way we add elements to our finite tournaments $\vec{F}$. Therefore, ensuring that these properties are satisfied usually consists of checking that we followed the standard process of adding elements to $\vec{F}$.

Lemma 3.4 Fix a condition $c = (\alpha, \vec{F}, \mathcal{C})$ where $\mathcal{C}$ is a k-cover class of $[t, +\infty)$. Let $E_0, \dots, E_{m-1}$ be finite sets, $\mathcal{D}$ be a non-empty $\Pi^{0,C}_1$ m-cover class of $[t', +\infty)$ for some $t' \geq t$ and $f : m \to k$ be a function such that for each $i < |\alpha|$ and $\nu < m$,

(iii) E_ν is R_i-transitive
(iv) $V_\nu \to_{R_i} E_\nu$ or $E_\nu \to_{R_i} V_\nu$ for each $V_0 \oplus \dots \oplus V_{m-1} \in \mathcal{D}$

Set $H_\nu = F_{f(\nu)} \cup E_\nu$ for each $\nu < m$. If properties (i) and (ii) of an extension are satisfied for $d = (\alpha, \vec{H}, \mathcal{D})$ with witness f, then d is a valid condition extending c.

Proof. All we need is to check properties (b) and (c) for d in the definition of a condition. We prove property (b). Fix an $i < |\alpha|$, some part ν of $\mathcal{D}$, and an $x \in V_\nu$ for some $V_0 \oplus \dots \oplus V_{m-1} \in \mathcal{D}$. In order to prove that $(F_{f(\nu)} \cup E_\nu) \setminus [0, \alpha(i)) \cup \{x\}$ is R_i-transitive, it is sufficient to check that the set contains no 3-cycle. Fix three elements $u < v < w \in (F_{f(\nu)} \cup E_\nu) \setminus [0, \alpha(i)) \cup \{x\}$.

- Case 1: $\{u, v, w\} \cap F_{f(\nu)} \setminus [0, \alpha(i)) \neq \emptyset$. Then $u \in F_{f(\nu)} \setminus [0, \alpha(i))$ as $F_{f(\nu)} < E_\nu < \{x\}$ and $u < v < w$. By property (ii), there is some $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ such that $E_\nu \cup \{x\} \subseteq Z_{f(\nu)}$. If $v \in F_{f(\nu)}$, then by property (b) of the condition c on $Z_{f(\nu)}$, $\{u, v, w\}$ is R_i-transitive. If $v \not\in F$, then by Lemma 3.3, $\{u\} \to_{R_i} Z_{f(\nu)}$ or $Z_{f(\nu)} \to_{R_i} \{u\}$, so $\{u, v, w\}$ is R_i-transitive since $v, w \in Z_{f(\nu)}$.
- Case 2: $\{u, v, w\} \cap F_{f(\nu)} \setminus [0, \alpha(i)) = \emptyset$. Then at least $u, v \in E_\nu$ because $E_\nu < \{x\}$. If $w \in E_\nu$ then $\{u, v, w\}$ is R_i-transitive by R_i-transitivity of E_ν. In the other case, $w = x \in V_\nu$. As $E_\nu \to_{R_i} V_\nu$ or $V_\nu \to_{R_i} E_\nu$, $\{u, v\} \to_{R_i} \{w\}$ or $\{w\} \to_{R_i} \{u, v\}$ and $\{u, v, w\}$ is R_i-transitive.

We now prove property (c) for d. Fix some $V_0 \oplus \dots \oplus V_{m-1} \in \mathcal{D}$, some part ν of $\mathcal{D}$ and some $i < |\alpha|$. By property (ii), there is some $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ such that $E_\nu \cup V_\nu \subseteq Z_{f(\nu)}$. By property (c) of the condition c, $Z_{f(\nu)}$ (and so V_ν) is included in a minimal R_i-interval (u, v) of $F_{f(\nu)} \setminus [0, \alpha(i))$. Here again, u and v may be respectively $-\infty$ and $+\infty$. By assumption, either $E_\nu \to_{R_i} V_\nu$ or $V_\nu \to_{R_i} E_\nu$. As E_ν is a finite R_i-transitive set, it has a minimal and a maximal element, say x and y. If $E_\nu \to_{R_i} V_\nu$ then V_ν is included in the R_i-interval (y, v). Symmetrically, if $V_\nu \to_{R_i} E_\nu$ then V_ν is included in the R_i-interval (u, x). To prove minimality for the first case, assume that some w is in the interval (y, v). Then $w \not\in F_{f(\nu)} \setminus [0, \alpha(i))$ by minimality of the interval (u, v) with respect to $F_{f(\nu)} \setminus [0, \alpha(i))$, and $w \not\in E_\nu$ by maximality of y. Minimality for the second case holds by symmetry. ☐

Now we have settled the necessary technical lemmas, we start proving lemmas which will be directly involved in the construction of the transitive subtournament. The following simple progress lemma states that we can always find an extension of a condition in which we increase both the finite approximations corresponding to the acceptable parts and the number of tournaments for which we are transitive simultaneously. Moreover, this extension can be found uniformly.

Lemma 3.5 (Progress) For every condition $c = (\alpha, \vec{F}, \mathcal{C})$ and every $s \in \omega$, there exists an extension $d = (\beta, \vec{E}, \mathcal{D})$ such that $|\beta| \geq s$ and $|E_\nu| \geq s$ for every acceptable part ν of $\mathcal{D}$. Furthermore, such an extension can be found C'-effectively, uniformly in c and s.

Proof. Fix a condition $c = (\alpha, \vec{F}, \mathcal{C})$. First note that for every $\beta \succeq \alpha$ such that $\beta(i) > max(F_\nu : \nu < parts(\mathcal{C}))$ whenever $|\alpha| \leq i < |\beta|$, $(\beta, \vec{F}, \mathcal{C})$ is a condition extending c. Therefore it suffices to prove that for every such condition c and every part ν of $\mathcal{C}$, we can C'-effectively find a condition $d = (\alpha, \vec{H}, \mathcal{D})$ refining c with witness $f : parts(\mathcal{D}) \to parts(\mathcal{C})$ such that f forks only parts refining part ν of $\mathcal{C}$, and either every such part μ of $\mathcal{D}$ is empty or $|H_\mu| > |F_\nu|$. Iterating the process finitely many times enables us to conclude.

Fix some part ν of $\mathcal{C}$ and let $\mathcal{D}$ be the collection of $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ such that $Z_\nu = \emptyset$. We can C'-decide whether or not $\mathcal{D}$ is empty. If $\mathcal{D}$ is non-empty, then $(\alpha, \vec{F}, \mathcal{D})$ is a valid extension of c with the identity function as witness and such that part ν of $\mathcal{D}$ is empty. If $\mathcal{D}$ is empty, we can C'-computably find some $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ and pick some $x \in Z_\nu$. Consider the

C-computable $2^{|\alpha|}$-partition $(X_\rho : \rho \in 2^{|\alpha|})$ of ω defined by

$$X_\rho = \{y \in \omega : (\forall i < |\alpha|)[R_i(y,x) \leftrightarrow \rho(i) = 1]\}$$

Let $\tilde{\mathcal{D}}$ be the cover class refining $\mathcal{C}^{[\nu,x]}$ such that part ν of $\tilde{\mathcal{D}}$ has $2^{|\alpha|}$ forks induced by the $2^{|\alpha|}$-partition $\vec{X}$. Define $\vec{H}$ by $H_\mu = F_\mu$ if μ refines a part different from ν, and $H_\mu = F_\nu \cup \{x\}$ if μ refines part ν of $\mathcal{C}$. The forking according to $\vec{X}$ ensures that property (iv) of Lemma 3.4 holds. By Lemma 3.4, $d = (\alpha, \vec{H}, \tilde{\mathcal{D}})$ is a valid extension of c. □

3.5. The strategy

Thanks to Lemma 3.5, we can define an infinite, C'-computable decreasing sequence of conditions $(\varepsilon, \emptyset, \{\omega\}) \geq c_0 \geq c_1 \geq \dots$ such that for each $s \in \omega$,

1. $|\alpha_s| \geq s$.
2. $|F_{s,\nu}| \geq s$ for each acceptable part ν of $\mathcal{C}_s$

where $c_s = (\alpha_s, \vec{F}_s, \mathcal{C}_s)$. As already noticed, if some acceptable part μ of $\mathcal{C}_{s+1}$ refines some part ν of $\mathcal{C}_s$, part ν of $\mathcal{C}_s$ is also acceptable. Therefore, the set of acceptable parts forms an infinite, finitely branching C'-computable tree $\mathcal{T}$. Let P be any infinite path through $\mathcal{T}$. The set $H(P) = (\bigcup_s F_{s,P(s)})$ is infinite, and $H(P) \setminus [0, \alpha_{i+1}(i))$ is R_i-transitive for each $i \in \omega$.

Our goal is to build a C'-computable function dominating every function computed by $H(P)$ for at least one path P trough $\mathcal{T}$. However, it requires too much computational power to distinguish acceptable parts from non-acceptable ones, and even some acceptable part may have only finitely many extensions. Therefore, we will dominate the functions computed by $H(P)$ for *every* path P trough $\mathcal{T}$.

At a finite stage, a condition contains finitely many parts, each one representing the construction of a transitive subtournament. As in the construction of a cohesive set, it suffices to check one by one whether there exists an extension of our subtournaments which will make terminate a given functional at a given input. In the next subsection, we develop the framework necessary to decide such a termination at a finite stage.

3.6. Forcing relation

As a condition $c = (\alpha, \vec{F}, \mathcal{C})$ corresponds to the construction of multiple subtournaments $F_0, F_1, \dots$ at the same time, the forcing relation will depend on which subtournament we are considering. In other words, the forcing relation depends on the part ν of $\mathcal{C}$ we focus on.

Definition 3.6 Fix a condition $c = (\alpha, \vec{F}, \mathcal{C})$, a part ν of $\mathcal{C}$ and two integers e, x.

1. $c \Vdash_\nu \Phi_e^{G \oplus C}(x) \uparrow$ if $\Phi_e^{(F_\nu \cup F_1) \oplus C}(x) \uparrow$ for all $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ and all subsets $F_1 \subseteq Z_\nu$ such that F_1 is R_i-transitive simultaneously for each $i < |\alpha|$.
2. $c \Vdash_\nu \Phi_e^{G \oplus C}(x) \downarrow$ if $\Phi_e^{F_\nu \oplus C}(x) \downarrow$.

The forcing relations defined above satisfy the usual forcing properties. In particular, let $c_0 \geq c_1 \geq \dots$ be an infinite decreasing sequence of conditions. This sequence induces an infinite, finitely branching tree of acceptable parts $\mathcal{T}$. Let P be an infinite path trough $\mathcal{T}$. If $c_s \Vdash_{P(s)} \Phi_e^{G \oplus C}(x) \uparrow$ (respectively $c_s \Vdash_{P(s)} \Phi_e^{G \oplus C}(x) \downarrow$) at some stage s, then $\Phi_e^{H(P) \oplus C}(x) \uparrow$ (respectively $\Phi_e^{H(P) \oplus C}(x) \downarrow$).

Another important feature of this forcing relation is that we can decide C'-uniformly in its parameters whether there is an extension forcing $\Phi_e^{G \oplus C}(x)$ to halt or to diverge. Deciding this relation with little computational power is useful because our C'-computable dominating function will need to decide termination $\Gamma^{G \oplus C}(x)$ to check whether it has to dominate the value outputted by $\Gamma^{G \oplus C}(x)$.

Lemma 3.7 For every condition $c = (\alpha, \vec{F}, \mathcal{C})$ and every pair of integers $e, x \in \omega$, there exists an extension $d = (\alpha, \vec{H}, \mathcal{D})$ such that for each part ν of $\mathcal{D}$

$$d \Vdash_\nu \Phi_e^{G \oplus C}(x) \uparrow \quad \vee \quad d \Vdash_\nu \Phi_e^{G \oplus C}(x) \downarrow$$

Furthermore, such an extension can be found C'-effectively, uniformly in c, e and x.

Proof. Given a condition c and two integers $e, x \in \omega$, let $I_{e,x}(c)$ be the set of parts ν of c such that $c \not\Vdash_\nu \Phi_e^{G \oplus C}(x) \downarrow$ and $c \not\Vdash_\nu \Phi_e^{G \oplus C}(x) \uparrow$. Note that $I_{e,x}(c)$ is C'-computable uniformly in c, e and x. It suffices to prove that given such a condition c and a part $\nu \in I_{e,x}(c)$, one can C'-effectively find an extension d with witness f such that $f(I_{e,x}(d)) \subseteq I_{e,x}(c) \setminus \{\nu\}$. Applying iteratively the operation enables us to conclude.

Fix a condition $c = (\alpha, \vec{F}, \mathcal{C})$ where $\mathcal{C}$ is a k-cover class, and fix some part $\nu \in I_{e,x}(c)$. The strategy is the following: either we can fork part ν of $\mathcal{C}$ into enough parts so that we force $\Phi_e^{G \oplus C}(x)$ to diverge on each forked part, or we can find an extension forcing $\Phi_e^{G \oplus C}(x)$ to converge on part ν without forking. Hence, we ask the following question.

Q2: Is it true that for every k-cover $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, for every $2^{|\alpha|}$-partition $\bigcup_{\rho \in 2^\alpha} X_\rho = Z_\nu$, there is some $\rho \in 2^{|\alpha|}$ and some finite set F_1 which is R_i-transitive for each $i < |\alpha|$ simultaneously, and such that $\Phi_e^{(F_\nu \cup F_1) \oplus C}(x) \downarrow$?

If the answer is no, then by forking the part ν of $\mathcal{C}$ into $2^{|\alpha|}$ parts, we will be able to force $\Phi_e^{G \oplus C}(x)$ to diverge. Let $m = k + 2^{|\alpha|} - 1$ and define the function $f : m \to k$ by $f(\mu) = \mu$ if $\mu < k$ and $f(\mu) = \nu$ otherwise. Let $\mathcal{D}$ be the collection of all m-covers $V_0 \oplus \dots \oplus V_{m-1}$ which f-refine some $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ and such that for every part μ of $\mathcal{D}$ f-refining part ν of $\mathcal{C}$ and every subset $F_1 \subseteq V_\mu$ which is R_i-transitive simultaneously for each $i < |\alpha|$, $\Phi_e^{F_\nu \cup F_1}(x) \uparrow$. Note that $\mathcal{D}$ is a $\Pi_1^{0,C}$ m-cover class f-refining $\mathcal{C}$. Moreover $\mathcal{D}$ is non-empty since the answer to *Q2* is no. Let $\vec{E}$ be defined by $E_\mu = F_\mu$ if $\mu < k$ and $E_\mu = F_\nu$ otherwise. The condition $d = (\alpha, \vec{E}, \mathcal{D})$ extends c with witness f. For every part μ of $\mathcal{D}$ f-refining part ν of $\mathcal{C}$, $d \Vdash_\mu \Phi_e^{G \oplus C}(x) \uparrow$, therefore $f(I_{e,x}(d)) \subseteq I_{e,x}(c) \setminus \{\nu\}$.

Suppose now that the answer is yes. By compactness, we can C'-effectively find a finite set $E \subseteq Z_\nu$ for some $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ such that for every $2^{|\alpha|}$-partition $(E_\rho : \rho \in 2^{|\alpha|})$ of E, there is some $\rho \in 2^{|\alpha|}$ and some set $F_1 \subseteq E_\rho$ which is R_i-transitive simultaneously for each $i < |\alpha|$ and such that $\Phi_e^{(F_\nu \cup F_1) \oplus C}(x) \downarrow$. There are finitely many $2^{|\alpha|}$-partitions of E. Let n be the number of such partitions. These partitions induce a finite C-computable n-partition of $dom(\mathcal{C})$ defined for each $(E_\rho : \rho \in 2^{|\alpha|})$ by

$$X_{\langle E_\rho : \rho \in 2^{|\alpha|} \rangle} = \left\{ y \in dom(\mathcal{C}) : (\forall i < |\alpha|) \left\{ \begin{array}{l} \text{if } \rho(i) = 0 \text{ then } E_\rho \to_{R_i} \{y\} \\ \text{if } \rho(i) = 1 \text{ then } \{y\} \to_{R_i} E_\rho \end{array} \right\} \right.$$

Let $\tilde{\mathcal{D}}$ be the $\Pi_1^{0,C}$ $(k+n-1)$-cover class refining $\mathcal{C}^{[\nu,E]}$ and such that part ν of $\mathcal{C}^{[\nu,E]}$ is refined accordingly to the above partition of $dom(\mathcal{C})$. Let $f : k + n - 1 \to k$ be the refining function witnessing it. Define $\vec{H}$ as follows. For every part μ of $\mathcal{D}$, refining part ν of $\mathcal{C}^{[\nu,E]}$, by definition of $\tilde{\mathcal{D}}$, there is some $2^{|\alpha|}$-partition $\langle E_\rho : \rho \in 2^{|\alpha|} \rangle$ of E such that for every $V_0 \oplus \dots V_{k+n-2} \in \tilde{\mathcal{D}}$, $V_\mu \subseteq X_{\langle E_\rho : \rho \in 2^{|\alpha|} \rangle}$. By choice of E, there exists some set $F_1 \subseteq E_\rho$ for some $\rho \in 2^{|\alpha|}$ which is R_i-transitive simultaneously for each $i < |\alpha|$ and such that $\Phi_e^{(F_\nu \cup F_1) \oplus C}(x) \downarrow$. This set F_1 can be found C'-effectively. Set $H_\mu = F_\nu \cup F_1$. For every part μ of $\tilde{\mathcal{D}}$ which refines some part ξ of $\mathcal{C}^{[\nu,E]}$ different from ν, set $H_\mu = F_\xi$. By Lemma 3.4, $d = (\alpha, \vec{H}, \tilde{\mathcal{D}})$ is a valid condition extending c. Moreover, for every part μ of $\tilde{\mathcal{D}}$ refining part ν of $\mathcal{C}$, $d \Vdash_\mu \Phi_e^{G \oplus C}(x) \downarrow$. Therefore $f(I_{e,x}(d)) \subseteq I_{e,x}(c) \setminus \{\nu\}$. □

3.7. **Construction**

We are now ready to construct our infinite transitive subtournament $H(P)$ together with a C'-computable function f dominating every $H(P) \oplus C$-computable function. Thanks to Lemma 3.5 and Lemma 3.7, we can C'-compute an infinite descending sequence of conditions $(\epsilon, \emptyset, 1^{<\omega}) \geq c_0 \geq c_1 \geq \dots$ such that at each stage $s \in \omega$,

1. $|\alpha_s| \geq s$
2. $|F_{s,\nu}| \geq s$ for each acceptable part ν of $\mathcal{C}_s$
3. $c_s \Vdash_\nu \Phi_e^{G \oplus C}(x) \downarrow$ or $c_s \Vdash_\nu \Phi_e^{G \oplus C}(x) \uparrow$ for each part ν of $\mathcal{C}_s$ if $\langle e, x \rangle = s$

where $c_s = (\alpha_s, \vec{F}_s, \mathcal{C}_s)$. Property 1 ensures that the resulting set with be eventually transitive for every tournament in $\vec{R}$. Property 2 makes the subtournaments infinite. Last, property 3 enables us to C'-decide at a finite stage whether a functional terminates on a given input, with the transitive subtournament as an oracle.

Define the C'-computable function $f : \omega \to \omega$ as follows: On input x, the function f looks at all stages s such that $s = \langle e, x\rangle$ for some $e \leq x$. For each such stage s, and each part ν in $\mathcal{C}_s$, the function C'-decides whether $c_s \Vdash_\nu \Phi_e^{G\oplus C}(x)\downarrow$ or $c_s \Vdash_\nu \Phi_e^{G\oplus C}(x)\uparrow$. In the first case, f computes the value $\Phi_e^{F_{s,\nu}\oplus C}(x)$. Having done all that, f returns a value greater than the maximum of the computed values.

Fix any infinite path P trough the infinite tree $\mathcal{T}$ of the acceptable parts induced by the infinite descending sequence of conditions. We claim that f dominates every function computed by $H(P) \oplus C$. Fix any Turing index $e \in \omega$ such that $\Phi_e^{H(P)\oplus C}$ is total. Consider any input $x \geq e$ and the corresponding stage $s = \langle e, x\rangle$. As $\Phi_e^{H(P)\oplus C}$ is total, $c_s \nVdash_{P(s)} \Phi_e^{G\oplus C}(x)\uparrow$, hence by property 3, $c_s \Vdash_{P(s)} \Phi_e^{G\oplus C}(x)\downarrow$. By construction, $f(x)$ computes the value of $\Phi_e^{F_{s,P(s)}\oplus C}(x)$ and returns a greater value. As $F_{s,P(s)}$ is an initial segment of $H(P)$, $\Phi_e^{F_{s,P(s)}\oplus C}(x) = \Phi_e^{H(P)\oplus C}(x)$ and therefore $f(x) > \Phi_e^{H(P)\oplus C}(x)$. This completes the proof of $\mathsf{AMT} \not\leq_c \mathsf{EM}$.

We identify a k-cover $Z_0 \cup \dots \cup Z_{k-1}$ of some set X with the k-fold join of its parts

4. The domination framework

The actual proof of Theorem 3.1 is slightly stronger than its statement as it creates a degree $\mathbf{d}$ bounding EM together with a computable instance X of AMT such that $\mathbf{d}$ bounds no solution to X. Therefore, having solutions to multiple tournaments in parallel is not enough to compute a solution to X. One may however ask whether *sequential* applications of EM (that is, defining a tournament such that every transitive subtournament will be used to define another tournament and so on) is enough to compute a solution to X.

Answering negatively this question requires to diagonalize against solutions Y_0 to computable instances of EM, but also against solutions Y_1 to Y_0-computable instances of EM and so on. The difficulty comes from the fact that diagonalizations happen at finite stages, at which we have only access to a finite approximation of Y_0, and so to a finite part of the Y_0-computable instances of EM. Thankfully, we only need a finite piece of an EM-instance to diagonalize against its solutions.

In this section, we develop a framework for building an ω-structure $\mathcal{M}$ satisfing some principle P such that every function in $\mathcal{M}$ is dominated by a single $\emptyset'$-computable function. Since by definition, the first-order part of an ω-structure is the set of standard natural numbers, ω-structures are characterized by their second-order part. An ω-structure satisfies RCA_0 if and only if its second-order part is a Turing ideal, i.e., a set of reals $\mathcal{I}$ closed under the effective join and the Turing reduction.

The whole construction will be done by iterating uniformly and $\emptyset'$-effectively the forcing constructions presented in the previous sections. We will not directly deal with the concrete forcing notion used for constructing solutions to EM-instances. Instead, we will manipulate an abstract partial order of forcing conditions. Abstracting the construction has several advantages:

1. It enables the reader to focus on the operations which are the essence of the construction. The reader will not be distracted by the implementation subtleties of EM which are not insightful to understand the overall structure.
2. The construction is more modular. We will be able to implement modules for EM and WKL independently, and combine them in section 6 to obtain a proof that $\mathsf{EM} \wedge \mathsf{WKL}$ does not imply AMT, and this without changing the main construction. This also enable reverse mathematicians to prove that other principles do not imply AMT without having to reprove the administrative aspects of the construction.

We shall illustrate our definitions with the case of COH in order to give a better intuition about the abstract operators we will define. As explained in section 3, the separation of COH

from AMT is already a consequence of the separation of EM from AMT. Therefore implementing the framework with COH is only for illustration purposes.

4.1. **Support**

The first step consists of defining the abstract partial order which will represent the partial order of forcing conditions. We start with an analysis of the common aspects of the different forcing notions encountered until yet, in order to extract their essence and define the abstract operators. In the following, we shall employ *stage* to denote a temporal step in the construction. An *iteration* is a spatial step representing progress in the construction of the Turing ideal. Multiple iterations are handled at a single stage.

Parts of a condition. When constructing cohesive sets for COH or transitive subtournaments for EM, we have been working in both cases with *conditions* representing parallel Mathias conditions. We shall therefore associate to our abstract notion of condition a notion of *part* representing one of the solutions we are building. A single abstract condition will have multiple *parts* representing the various candidate solutions constructed in parallel for the same instance.

For example, in the forcing notion for COH, a condition $c = (F_\nu : \nu \in 2^n)$ can be seen as 2^n parallel Mathias conditions (F_ν, R_ν) where $R_0, R_1, \dots$ is the universal instance of COH. In this setting, the parts of c are the pairs $\langle c, \nu \rangle$ for each $\nu \in 2^n$. One may be tempted to get rid of the notion of condition and directly deal with its parts since in COH, a condition is only the tuple of its parts. However, in the forcing notion $c = (\vec{F}, \mathcal{C})$ for EM, the parts are interdependent since adding element to some F_ν will remove inconsistent covers from $\mathcal{C}$ and therefore may restrict the reservoirs of the other parts.

Satisfaction. As explained, a part represents the construction of one solution, whereas a condition denotes multiple solutions in parallel. We can formalize this intuition by defining a *satisfaction function* which, given a part of a condition, returns the collection of the sets satisfying it. For example, a set G satisfies part ν of the COH condition $c = (F_\nu : \nu \in 2^n)$ if it satisfies the Mathias condition $(F_\nu, R_\nu \setminus [0, max(F)])$, in other words, if $F_\nu \subseteq G$ and $G \setminus F_\nu \subseteq R_\nu \setminus [0, max(F_\nu)]$.

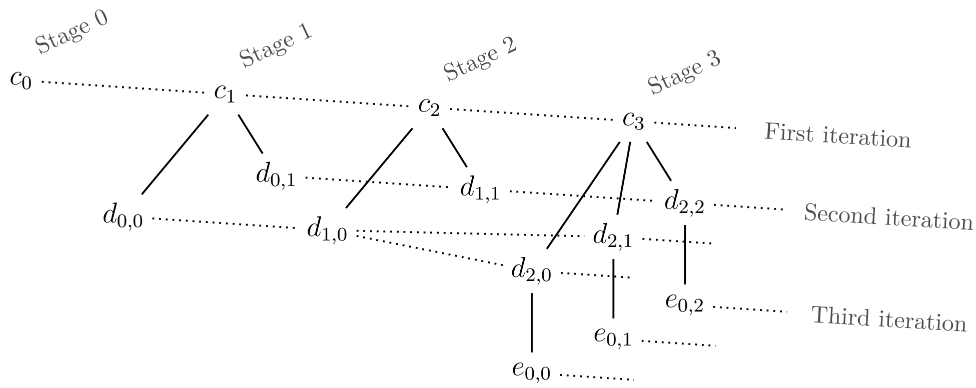

FIGURE 2. Example of construction of a Turing ideal by an iterative forcing in which conditions may have multiple parts. The nodes are the conditions, the dotted edges are condition extensions and the plain edges are the parts of the conditions.

Initial condition. In a standard (i.e. non-iterative) forcing, we build an infinite decreasing sequence of conditions, starting from one initial condition c_0. In COH, this initial condition is $(\emptyset, \varepsilon)$, where ε is the empty string. Since $R_\varepsilon = \omega$, this coincides with the standard initial Mathias condition $(\emptyset, \omega)$. In an iterative forcing, we add progressively new iterations by starting a new decreasing sequence of conditions below each part of the parent condition. Since COH admits a universal instance, there is no need to choose which instance we want to solve at each iteration. However, in the case of EM, we will take a new EM-instance each time, so that the resulting Turing ideal is the second-order part of an ω-model satisfying EM. Therefore, an EM-condition is in fact a condition $(\vec{F}, \mathcal{C}, R)$ where R is an instance of EM. The chosen instance of EM will be decided at the initialization of a new iteration and will be preserved by condition extension. The choice of the instance depends only on the iteration level. Therefore we can define an initialization function which, given some integer, returns the initial condition together with the chosen instance.

Parameters. The difficulty of the iterative forcing comes from the fact that an instance of the principle P may depend on the previous iterations. During the construction, the partial approximations of the previous iterations become more and more precise, enabling the instance at the next iteration to be defined on a larger domain. In the definition of our abstract partial order, we will use a formal parameter D which will represent the join of the constructed solutions in the previous iterations. For example, in the formal definition of the partial order for COH, we will say that some condition $d = (E_\mu : \mu \in 2^m)$ extends another condition $c = (F_\nu : \nu \in 2^n)$ if $m \geq n$, and $(E_\mu, R^D_\mu \setminus [0, max(E_\mu)])$ Mathias extends $(F_\nu, R^D_\nu \setminus [0, max(F_\nu)])$ for each $\nu \preceq \mu$. This syntactic constraints has to be understood as $(E_\mu, R^X_\mu \setminus [0, max(E_\mu)])$ Mathias extends $(F_\nu, R^X_\nu \setminus [0, max(F_\nu)])$ for every set $X = X_0 \oplus \dots \oplus X_{n-1}$ such that X_i satisfies the ancestor of d in the iteration axis at the ith level. In the case of COH, only a finite initial segment of X is needed to witness the extension.

We are now ready to define the notion of module support.

Definition 4.1 (Module support) A *Module support* is a tuple $\langle \mathbb{P}, \mathbb{U}, \mathtt{parts}, \mathtt{init}, \mathtt{sat} \rangle$ where

(1) $(\mathbb{P}, \leq_\mathbb{P})$ is a partial order. The set $\mathbb{P}$ has to be thought of as the set of forcing conditions. Therefore, the elements of $\mathbb{P}$ will be called *conditions*.
(2) $\mathbb{U}$ is a set of *parts*. The notion of part is due to the fact that most of our forcing conditions represent multiple objects built in parallel.
(3) $\mathtt{parts} : \mathbb{P} \to \mathcal{P}_{fin}(\mathbb{U})$ is a computable function which, given some condition $c \in \mathbb{P}$, gives the finite set of parts associated to c.
(4) $\mathtt{init} : \mathbb{N} \to \mathbb{P}$ is a computable function which, given some integer n representing the iteration level, provides the initial condition of the forcing at the nth iteration.
(5) $\mathtt{sat} : \mathbb{U} \to \mathcal{P}(2^\omega)$ is a function which, given some part ν of some condition c, returns the collections of sets satisfying it.

Furthermore, a module support is required to satisfy the following property:

(a) If $d \leq_\mathbb{P} c$ for some $c, d \in \mathbb{P}$, then there is a function $f : \mathtt{parts}(d) \to \mathtt{parts}(c)$ such that $\mathtt{sat}(\nu) \subseteq \mathtt{sat}(f(\nu))$ for each $\nu \in \mathtt{parts}(d)$. We may write it $d \leq_f c$ and say that f is the *refinement function witnessing* $d \leq_\mathbb{P} c$.

Given two conditions $c, d \in \mathbb{P}$ such that $d \leq_f c$, we say that f *forks* part ν of c if $|f^{-1}(\nu)| \geq 2$. This forking notion will be useful in the definition of a module. Let us illustrate the notion of module support by defining one for COH.

Module support for COH. Define the tuple $\langle \mathbb{P}, \mathbb{U}, \mathtt{parts}, \mathtt{init}, \mathtt{sat} \rangle$ as follows: $\mathbb{P}$ is the collection of all conditions $(F_\nu : \nu \in 2^n)$ where F_ν is a finite set of integers. Given some $d = (E_\mu : \mu \in 2^m)$ and $c = (F_\nu : \nu \in 2^n)$, $d \leq_\mathbb{P} c$ if $m \geq n$, and $(E_\mu, R^D_\mu \setminus [0, max(E_\mu)])$ Mathias extends $(F_\nu, R^D_\nu \setminus [0, max(F_\nu)])$ for each $\nu \preceq \mu$. Let $\mathbb{U}$ be the set of all pairs $\langle (F_\nu : \nu \in 2^n), \nu \rangle$ where $\nu \in 2^n$. Given some condition $c = (F_\nu : \nu \in 2^n) \in \mathbb{P}$, $\mathtt{parts}(c) = \{\langle c, \nu \rangle : \nu \in 2^n\}$. For every level $n \geq 0$, $\mathtt{init}(n) = (\emptyset, \varepsilon)$. Define $\mathtt{sat}(\langle (F_\nu : \nu \in 2^n), \nu \rangle)$ to be the collection of all sets satisfying the Mathias condition $(F_\nu, R_\nu \setminus [0, max(F_\nu)])$.

We now check that property (a) holds. Let $d = (E_\mu : \mu \in 2^m)$ and $c = (F_\nu : \nu \in 2^n)$ be such that $d \leq_\mathbb{P} c$. In particular, $m \geq n$. Define $f : \mathbb{U} \to \mathbb{U}$ by $f(\langle d, \mu \rangle) = \langle c, \mu{\upharpoonright}n \rangle$. We claim that f is a refinement function witnessing $d \leq_\mathbb{P} c$. $\mathtt{sat}(\langle d, \mu \rangle)$ is the collection of sets satisfying the Mathias condition $\tilde{d} = (E_\mu, R_\mu \setminus [0, max(E_\mu)))$ and $\mathtt{sat}(\langle c, \mu{\upharpoonright}n \rangle)$ is the collection of sets satisfying $\tilde{c} = (F_{\mu\upharpoonright n}, R_{\mu\upharpoonright n} \setminus [0, max(F_{\mu\upharpoonright n})))$. Since $\mu{\upharpoonright}n \preceq \mu$, $\tilde{d}$ Mathias extends $\tilde{c}$ by definition of sat. Considering the Mathias conditions, every set satisfying $\tilde{d}$ satisfies $\tilde{c}$, so $\mathtt{sat}(\langle d, \mu\rangle) \subseteq \mathtt{sat}(f(\langle d, \mu\rangle))$. Therefore the property (a) of a module support is satisfied.

4.2. **Modules**

We previously defined the abstract structure we shall use as a support of the construction. The next step consists of enriching this structure with a few more operators which will enable us to decide Σ^0_1 properties over the constructed sets. The success or failure in forcing some property will depend on the parts of a condition. Note that at a finite stage, we handle a finite tree of conditions. We can therefore cover all cases by asking finitely many questions. Let us go back to the COH example, and more precisely how we decided Σ^0_1 properties over it.

Iteration 1. At the first iteration, we would like to decide whether the $\Sigma^{0,G}_1$ formula

$$\psi(G) = (\exists s, m)(\Phi^G_{e,s}(n) \downarrow = m)$$

will hold, where G is a formal parameter denoting the constructed set. Furthermore, we want to collect the value of $\Phi^G_e(n)$ if it halts. The formula $\psi(G)$ can be seen as a *query*, whose *answers* are either $\langle \mathtt{No}\rangle$ if $\psi(G)$ does not hold, or a tuple $\langle \mathtt{Yes}, s, m\rangle$ such that $\Phi^G_{e,s}(n) \downarrow = m$ if $\psi(G)$ holds. Given some condition $c = (F_\nu : \nu \in 2^n)$, we can ask on each part $\langle c, \nu\rangle$ whether the formula $\varphi(G)$ will hold or not, by *boxing* the query $\psi(G)$ into a Σ^0_1 query ϕ without the formal parameter G, such that ϕ holds if and only if we can find an extension d of c forcing $\psi(G)$ on the parts of d refining part ν of c. Concretely, we can define ϕ as follows:

$$\phi = (\exists F_1 \subseteq R_\nu \setminus [0, max(F_\nu)])\psi(F_\nu \cup F_1)$$

This query can be $\emptyset'$-decided. If the formula ϕ holds, we can effectively find some answer to ϕ, that is, a tuple $\langle \mathtt{Yes}, F_1, s, m\rangle$ such that $F_1 \subseteq R_\nu \setminus [0, max(F_\nu)]$ and $\Phi^{F_\nu \cup F_1}_{e,s}(n) \downarrow = m$. The extension d obtained by adding F_1 to F_ν forces $\psi(G)$ to hold for every set G satisfying the part $\langle d, \nu\rangle$ of the condition d. The answer to $\psi(G)$ is obtained by forgetting the set F_1 from the answer to ϕ. On the other hand, if the formula ϕ does not hold, the answer is $\langle \mathtt{No}\rangle$ and c already forces $\psi(G)$ not to hold.

Iteration 2. At the second iteration, we work with conditions $c_1 = (E_\mu : \mu \in 2^m)$ which are below some part ν of some condition $c_0 = (F_\nu : \nu \in 2^n)$ living at the first iteration level. We want to decide Σ^{0,G_0,G_1}_1 formulas, where G_0 and G_1 are formal parameters denoting the sets contructed at the first iteration and at the second iteration, respectively. We basically want to answer queries of the form

$$\varphi(G_0, G_1) = (\exists s, m)(\Phi^{G_0 \oplus G_1}_{e,s}(n) \downarrow = m)$$

We will ask this question on each part of c_1. By the same boxing process as before applied relative to c_1, we obtain a formula $\psi(G_0)$ getting rid of the formal parameter G_1, and defined by

$$\psi(G_0) = (\exists E_1 \subseteq R^{G_0}_\mu \setminus [0, max(E_\mu)])\varphi(G_0, E_\mu \cup E_1)$$

The formula $\psi(G_0)$ is a now a query at the first iteration level. We can apply another boxing to $\psi(G_0)$ relative to c_0 to obtain a Σ^0_1 formula ϕ without any formal parameter.

$$\phi = (\exists F_1 \subseteq R_\nu \setminus [0, max(F_\nu)])\psi(F_\nu \cup F_1)$$

This formula can again be $\emptyset'$-decided. If it holds, an answer $a = \langle \mathtt{Yes}, F_1, E_1, s, m\rangle$ can be given. At the first iteration level, we unbox the answer a to obtain a tuple $b = \langle \mathtt{Yes}, E_1, s, m\rangle$ and an extension d_0 of c_0. The extension d_0 forces the tuple b to answer the query $\psi(G_0)$ and is obtained by adding F_1 to F_ν. At the second iteration level, we unbox again the answer b to obtain a tuple $\langle \mathtt{Yes}, s, m\rangle$ and an extension d_1 to c_1, forcing $\langle \mathtt{Yes}, s, m\rangle$ to answer the query $\varphi(G_0, G_1)$. The whole decision process is summarized in Figure 3.

Progress. We may also want to force some specific properties required by the principle P. In the case of Ramsey-type principles, we need to force the set G to be infinite. This can be done with the following query for each k:

$$(\exists n)[n \in G \wedge n > k]$$

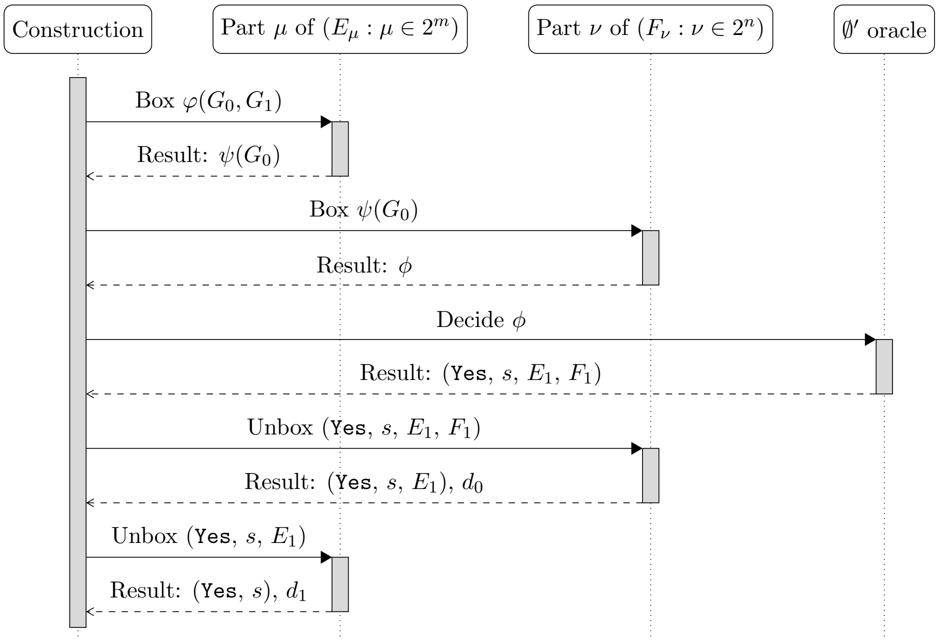

FIGURE 3. This sequence diagram shows the boxing process of the Σ_1^{0,G_0,G_1} formula $\varphi(G_0, G_1) = (\exists s)\Phi_{e,s}^{G_0 \oplus G_1}(x) \downarrow$ into a Σ_1^0 formula without formal parameters in order to decide it. The formula $\varphi(G_0, G_1)$ is boxed into a formula $\psi(G_0) = (\exists E_1 \subseteq R_\mu^{G_0} \setminus [0, max(E_\mu)])\varphi(G_0, E_\mu \cup E_1)$ which is itself boxed into $\phi = (\exists F_1 \subseteq R_\nu \setminus [0, max(F_\nu)])\psi(F_\nu \cup F_1)$.

The progress query can take various forms, depending on the considered principle. For example, in WKL, we need to force the path to be infinite by asking the following question for each k:

$$(\exists \sigma \in 2^k)[\sigma \prec G]$$

We will therefore define some progress operator which outputs some query that the construction will force to hold or not. We will choose the actual forcing notions so that the formula can be forced to hold for at least one part of each condition. The parameter k will not be given to the operator, since it can be boxed into the current condition, in a monadic style.

We are now ready to define the notion of module as a module support enriched with some boxing, unboxing and progress abstract operators. In what follows, $Query[\vec{X}]$ is the set of all Σ_1^0 formulas with $\vec{X}$ as formal parameters, and $Ans[\vec{X}]$ is the set of their answers.

Definition 4.2 (Module) A *module* is a tuple $\langle \mathbb{S}, \mathtt{box}, \mathtt{unbox}, \mathtt{prog} \rangle$ where

(1) $\mathbb{S} = \langle \mathbb{P}, \mathbb{U}, \mathtt{parts}, \mathtt{init}, \mathtt{sat} \rangle$ is a module support.
(2) $\mathtt{box} : \mathbb{U} \times Query[D, G] \to Query[D]$ is a computable boxing function which, given some part ν of some condition $c \in \mathbb{P}$ and some Σ_1^0 formula $\varphi(D, G)$, outputs a Σ_1^0 formula $\psi(D)$.
(3) $\mathtt{unbox} : \mathbb{U} \times Ans[D] \to \mathbb{P} \times (\mathbb{U} \to \mathbb{U}) \times (\mathbb{U} \to Ans[D, G])$ is a computable function which, given some part ν of some condition $c \in \mathbb{P}$ and some answer a to a Σ_1^0 formula $\psi(D)$ encoding a Σ_1^0 formula $\varphi(D, G)$, outputs a tuple $\langle d, f, g \rangle$ such that $d \leq_f c$ where f forks only part ν of c, and for every part μ of d such that $f(\mu) = \nu$, and every set $G \in \mathtt{sat}(\mu)$, $g(\mu)$ is an answer to $\varphi(D, G)$.
(3) $\mathtt{prog} : \mathbb{U} \to Query[D, G]$ is a computable function which provides a question forcing some progress in the solution. It usually asks whether we can force the partial approximation to be defined on a larger domain.

Let us go back to the COH case. Define the COH module $\langle \mathbb{S}, \mathtt{box}, \mathtt{unbox}, \mathtt{prog} \rangle$ as follows: $\mathbb{S}$ is the COH module support previously defined. Given some condition $c = (F_\nu : \nu \in 2^n)$, some $\nu \in 2^n$ and some query $\varphi(D, G)$, $\texttt{box}(\langle c, \nu \rangle, \varphi)$ is the query $\psi(D)$ defined by

$$\psi(D) = (\exists F_1 \subseteq R^D_\nu \setminus [0, max(F_\nu)])\varphi(D, F_\nu \cup F_1)$$

Set $\texttt{unbox}(\langle c, \nu \rangle, \langle \texttt{No} \rangle) = \langle c, id, g \rangle$ where id is the identity function and $g(\nu) = \langle \texttt{No} \rangle$. Given an answer $a = \langle \texttt{Yes}, F_1, a_1 \rangle$ to the question $\psi(D)$, $\texttt{unbox}(\langle c, \nu \rangle, a) = \langle d, f, g \rangle$ where $d = (E_\mu : \mu \in 2^n)$ is an extension of c such that $E_\nu = F_\nu \cup F_1$, and $E_\mu = F_\mu$ whenever $\mu \neq \nu$. The function $f : \mathbb{U} \to \mathbb{U}$ is defined by $f(\langle d, \mu \rangle) = \langle c, \mu \rangle$ for each $\mu \in 2^n$. The function $g : \mathbb{U} \to Ans[D, G]$ is the constant function defined by $g(\langle d, \mu \rangle) = \langle \texttt{Yes}, a \rangle$.

We claim that f is a refinement function witnessing $d \leq c$. For every $\mu \neq \nu$, $E_\mu = F_\mu$ so $(E_\mu, R^D_\mu \setminus [0, max(E_\mu)])$ Mathias extends $(F_\mu, R^D_\mu \setminus [0, max(F_\mu)])$. By definition of an answer, $F_1 \subseteq R^D_\nu \setminus max(F_\nu))$ so $(E_\nu, R^D_\nu \setminus [0, max(E_\nu)])$ Mathias extends $(F_\nu, R^D_\nu \setminus [0, max(F_\nu)])$. Therefore $d \leq_{\mathbb{P}} c$. Last, $\texttt{prog}(\langle c, \nu \rangle)$ is the query

$$\psi(D, G) = (\exists x \in G)[x > max(F_\nu)]$$

When considering cohesiveness, we must ensure an additional kind of progress. Indeed, we must partition the reservoir according to $(R_\sigma : \sigma \in 2^n)$ for larger and larger n. We can slightly modify the forcing notion for COH and "hack" this kind of progress in the $\texttt{unbox}$ operator by making it return a condition whose parts are split accordingly. Since the separation of EM from AMT entails the separation of COH from AMT, we will not go into the details for fixing this progress issue.

4.3. Construction

We will construct an infinite sequence of trees of conditions by stages, such that each level corresponds to one iteration. We will add progressively more and more iterations, so that the limit tree is of infinite depth. In order to simplify the presentation of the construction, we need to introduce some additional terminology.

Definition 4.3 (Stage tree) A *stage tree* is a finite tree T whose nodes are conditions and whose edges are parts of conditions. It is defined inductively as follows: A *stage tree of depth 0* is a condition. A *stage tree of depth* $n+1$ is a tuple $\langle c, h \rangle$ where c is a condition and h is a function such that $h(\nu)$ is a stage tree of depth n for each $\nu \in \texttt{parts}(c)$.

We consider that the stage subtree $h(\nu)$ is linked to c by an edge labelled by ν. The *root* of T is itself if T is a stage tree of depth 0. If $T = \langle c, h \rangle$ then the root of T is c. According to our notation on trees, if $T = \langle c, h \rangle$, we write $T^{[\nu]}$ to denote $h(\nu)$. We also write $T{\upharpoonright}k$ to denote the restriction of T to its stage subtree of depth k. At each stage s of the construction, we will end up with a stage tree of depth s. The initial stage tree will be $T_0 = \texttt{init}(0)$. There is a natural notion of stage tree extension induced by the extension of its conditions.

Definition 4.4 (Stage tree extension) A stage tree T_1 of depth n *extends* a stage tree T_0 of depth 0 if there is a function f such that $c_1 \leq_f T_0$ where c_1 is the root of T_1. We say that f is a *refinement tree of depth 0* and write $T_1 \leq_f T_0$. A stage tree $T_1 = \langle c_1, h_1 \rangle$ of depth $n+1$ *extends* a stage tree $T_0 = \langle c_0, h_0 \rangle$ of depth $m+1$ if there is a function f such that $c_1 \leq_f c_0$ and a function r such that $r(\nu)$ is a refinement tree of depth m such that $h_1(\nu) \leq_{r(\nu)} h_0(f(\nu))$ for each part ν of c_1. The tuple $R = \langle f, r \rangle$ is a *refinement tree of depth* $m+1$ and we write $T_1 \leq_R T_0$.

Note that if $T_1 \leq_R T_0$, where T_1 is a stage tree of depth n and T_0 is a stage tree of depth m, then $n \geq m$. We may also write $T_1 \leq T_0$ if there is a refinement tree R of depth m such that $T_1 \leq_R T_0$.

At each stage, we will extend the current stage tree to a stage tree of larger depth and whose conditions force more and more properties. The resulting sequence of stage trees $T_0 \geq T_1 \geq \dots$ can be seen as a 2-dimensional tree with the following axes:

- The *stage axis* is a temporal dimension. Let $c_0 \geq c_1 \geq \dots$ be such that c_s is the root of T_s for each stage s. As we saw in the computable non-reducibility case, the

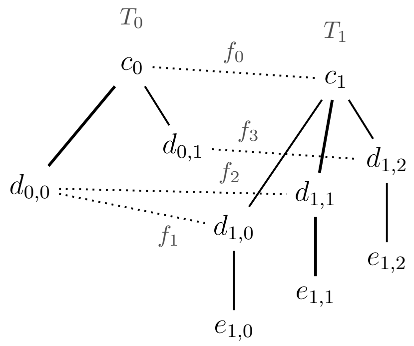

FIGURE 4. In this example, the refinement tree R whose nodes are $\{f_0, f_1, f_2, f_3\}$ witnesses the extension of the stage tree T_0 whose nodes are $\{c_0, d_{0,0}, d_{0,1}\}$ by the stage tree T_1 whose nodes are $\{c_1, d_{1,0}, d_{1,1}, d_{1,2}, e_{1,0}, e_{1,1}, e_{1,2}\}$. The condition c_1 has three parts, and f_0-refines the condition c_0 which has two parts. The conditions $d_{1,0}$, $d_{1,1}$ and $d_{1,2}$ have only one part. The path $c_1 - d_{1,1} - e_{1,1}$ trough the tree T_1 R-refines the path $c_0 - d_{0,0}$ through the tree T_0.

parts of this sequence forms an infinite, finitely branching tree. Let P be any infinite path through this tree. More formally, P is a sequence $\nu_0, \nu_1, \dots$ such that ν_{s+1} is a part of c_{s+1} refining the part ν_s in c_s for each s. Consider now the sequence of stage trees $T_0^{[\nu_0]} \geq T_1^{[\nu_1]} \geq \dots$ The sequence lives at the second iteration level, below the path P. Its roots induce another infinite, finitely branching tree, and so on. Therefore, at each level, we can define an infinite, finitely branching tree of parts, once we have fixed the path P through the tree of parts at the previous level.

– The *iteration axis* is a spatial (or vertical) dimension corresponding to the depth. The notion of stage tree makes explicit the finite tree obtained when fixing a stage. A path through a stage tree corresponds to the choices made at each level, between the different parts of a condition. We did not define the notion of acceptable part in this framework. Therefore, the choice of the part is delegated to the module, which will have to justify that at least one of the parts is extensible.

Definition 4.5 (Partial path) A *partial path* ρ through a stage tree T of depth n is defined inductively as follows: A partial path through a stage tree T of depth 0 is a part of T. A partial path through a stage tree $T = \langle c, h\rangle$ of depth $n+1$ is either a part of c, or a sequence ν, ρ where ν is a part of c and ρ is a partial path through $h(\nu)$. A *path* through T is a partial path of length $n+1$.

We denote by $P(T)$ and by $PP(T)$ the collection of paths and partial paths through T, respectively. Note that a partial path has length at least 1. The notion of refinement between partial paths is defined in the natural way. We can also extend the notation $T^{[\rho]}$ to partial paths ρ through T with the obvious meaning. There is also a notion of satisfaction of a stage tree induced by the sat operator.

Definition 4.6 (Stage tree satisfaction) A set G_0 *satisfies* a partial path ν_0 through a stage tree T of depth 0 if $G_0 \in \mathtt{sat}(\nu_0)$. A tuple of sets $G_0, G_1, \dots, G_k$ *satisfies* a partial path $\nu_0, \dots, \nu_k$ through a stage tree $T = \langle c, h\rangle$ of depth $n+1$ if $G_0 \in \mathtt{sat}(\nu_0)$ and $k = 0$, or $G_1, \dots, G_k$ satisfies the partial path $\nu_1, \dots, \nu_k$ through the stage tree $h(\nu_0)$. A tuple of sets $G_0, G_1, \dots, G_k$ *satisfies* a stage tree T of depth n if it satisfies a partial path through T.

Again, if $G_0, \dots, G_k$ satisfies a stage tree of depth n, then $k \leq n$. The notion of satisfaction induces a forcing relation. We say that T *forces* some formula $\varphi(D, G)$ below a partial path $\rho = \nu_0, \dots, \nu_k$ (written $T \Vdash_\rho \varphi(U, G)$ if for every tuple of sets $G_0, \dots, G_k$ satisfying $\nu_0, \dots, \nu_k$, $\varphi(\bigoplus_{i<k} G_i, G_k)$ holds.

We now prove a few lemmas stating that we can compose locally the abstract operators to obtain some global behavior. The first trivial lemma shows how to increase the size of a stage tree. This is where we use the operator `init`.

Lemma 4.7 (Growth lemma) For every stage tree T_0 of depth n and every m, there is a stage tree T_1 of depth $n+1$ such that $T_1{\upharpoonright}n = T_0$, and whose leaves are $\mathtt{init}(m)$. Moreover, T_1 can be computably found uniformly in T_0.

Proof. The proof is done inductively on the depth of T_0. In the base case, T_0 is a stage tree of depth 0 and is therefore a condition c_0. Let h be the function such that $h(\nu) = \mathtt{init}(m)$ for each $\nu \in \mathtt{parts}(c_0)$. The tuple $T_1 = \langle c_0, h\rangle$ is a stage tree of depth 1 such that $T_1{\upharpoonright}0 = c_0 = T_0$. It can be computably found uniformly in T_0 since `init` and `parts` are computable. Suppose now that $T_0 = \langle c_0, h_0\rangle$ is a stage tree of depth $n+1$. By induction hypothesis, we can define a function h_1 such that for each $\nu \in \mathtt{parts}(c_0)$, $h_1(\nu)$ is a stage tree of depth $n+1$ and $h_1(\nu){\upharpoonright}n = h_0(\nu)$. The tuple $T_1 = \langle c_0, h_1\rangle$ is a stage tree of depth $n+2$ such that $T_1{\upharpoonright}n+1 = \langle c_0, h_0\rangle = T_0$. ☐

We will always apply the growth lemma in the case $m = n+1$. However, the full statement was necessary to apply the induction hypothesis. Note that, since $T_1{\upharpoonright}n = T_0$, we have $T_1 \leq T_0$ as witnessed by taking the refinement tree of identity functions. The next lemma states that we can, given some stage tree T_0 and some query $\varphi(D,G)$, obtain another stage tree $T_1 \leq T_0$ in which we have decided $\varphi(D,G)$ at every part of every condition in T_0. Its proof is non-trivial since when forcing some property, we may increase the number of branches of the stage tree. We need therefore to define some elaborate decreasing property to prove termination of the procedure. The query lemma is assumed yet and will be proven in subsection 4.4.

Lemma 4.8 (Query lemma) Let T_0 be a stage tree of depth n and $q : PP(T) \to Query[D,G]$ be a computable function. There is a stage tree $T_1 \leq T_0$ of depth n such that every partial path ξ through T_1 refines a partial path ρ through T_0 for which $T_1 \Vdash_\xi q(\rho)$ or $T_1 \Vdash_\xi \neg q(\rho)$, Moreover, T_1 and the function of answers $a : PP(T_1) \to Ans[D,G]$ can be Δ^0_2-found uniformly from T_0.

The following domination lemma is a specialization of the query lemma by considering queries about termination of programs.

Lemma 4.9 (Domination lemma) For every stage tree T_0 of depth n, there is a stage of tree $T_1 \leq T_0$ of depth n and a finite set $U \subset \omega$ such that for every tuple $G_0, \dots, G_n$ satisfying T_1 and every $e, x, i \leq n$, $\Phi_e^{G_0\oplus\dots\oplus G_i}(x) \in U$ whenever $\Phi_e^{G_0\oplus\dots\oplus G_i}(x)$ halts. Moreover, T_1 and U can be Δ^0_2-found uniformly from T_0.

Proof. Apply successively the query lemma with $q(\xi) = (\exists s, m)\Phi_{e,s}^{D\oplus G}(x)\downarrow= m$ for each $e, x \leq n$, in order to obtain the tree T_1 together with an upper bound k to the answers to $q(\rho)$. We claim that the set $U = [0,k]$ satisfies the desired property. Let $G_0, \dots, G_n$ be a tuple satisfying T_1, and let $e, x, i < n$ be such that $\Phi_e^{G_0\oplus\dots\oplus G_i}(x)\downarrow$. By definition of satisfaction, there is some partial path ρ through T_1 such that $G_0, \dots, G_i$ satisfies ρ. By the query lemma, $T_1 \Vdash_\rho (\exists s, m)\Phi_{e,s}^{D\oplus G}(x)\downarrow= m$ or $T_1 \Vdash_\rho \neg(\exists s, m)\Phi_{e,s}^{D\oplus G}(x)\downarrow= m$. Since $\Phi_e^{G_0\oplus\dots\oplus G_i}(x)\downarrow$, the former holds, and k is greater than the answer to the query, so is greater than m. Uniformity is inherited from the query lemma. ☐

We construct an infinite Δ^0_2 sequence of finite trees of conditions $T_0 \geq T_1 \geq \dots$ as follows: At stage 0, we start with a stage tree T_0 of depth 0 defined by $\mathtt{init}(0)$. At each stage $s > 0$, assuming we have defined a stage tree T_{s-1} of depth $s-1$, act as follows:

(S1) *Growth*: Apply the growth lemma to obtain a stage tree $T_s^1 \leq T_{s-1}$ of depth s. Intuitively, this step adds a new iteration and therefore ensures that the construction will have eventually infinitely many levels of iteration.

(S2) *Progress*: Apply to T_s^1 the query lemma with $q = \mathtt{prog}$ to obtain a stage tree $T_s^2 \leq T_s^1$ such that the progress function is forced at each partial path. This step ensures that for every tuple $G_0, G_1, \dots$ such that $G_0, \dots, G_k$ satisfies each T_s, $s \geq k$, the progress query will have been decided on G_i infinitely many times.

(S3) *Domination*: Apply to T^2_s the domination lemma to obtain a stage tree $T_s \leq T^2_s$ and a finite set U such that for every tuple $G_0, \dots, G_s$ satisfying T_s and every $e, x, i \leq s$, if $\Phi_e^{G_0 \oplus \dots \oplus G_i}(x)$ halts, then its value will be in U. Since the whole construction is Δ^0_2 and we uniformly find such a set U, this step enables us to define a Δ^0_2 function which will dominate every function in the Turing ideal.

4.4. **Queries**

In this section, we develop the tools necessary to prove the query lemma (Lemma 4.8). Given some stage tree T_0 and some query function $q : PP(T) \to Query[D, G]$, the query lemma states that we can find a stage tree T_1 extending T_0 and which forces either $q(\rho)$ or its negation on each partial path through T_1 refining the partial ρ through T_0. The stage tree T_0 is finite and has therefore finitely many partial paths. The naive algorithm would consist of taking an arbitrary partial path ρ through T_0, then decide $q(\rho)$ thanks to the process illustrated in Figure 3 and extend T_0 into a stage tree T_1 which forces $q(\rho)$ or its negation on every path refining ρ. One may expect to obtain the query lemma by iterating this process finitely many times.

The termination of the algorithm depends on the shape of the extension T_1 obtained after deciding $q(\rho)$. We need to ensure that we made some progress so that we will have covered all paths at some point. Let us look more closely at the construction of the extension T_1. Given some query $\varphi(D, G)$ and some part ν, we call the $\texttt{unbox}(\nu, \varphi)$ operator to obtain another query $\psi(D)$ getting rid of the forcing variable G. Using $\emptyset'$, we obtain an answer a to the formula $\varphi(\emptyset)$ and then call $\texttt{unbox}(\nu, a)$ to obtain some extension forking only ν, and forcing either $\varphi(D, G)$ or its negation on every part refining ν. This extension may therefore increase the number of parts, but ensures some progress on each of the forked parts.

If T_0 is a stage tree of depth 0, the termination of the process is clear. Indeed, T_0 is a condition c_0 and the partial paths through T_0 are simply the parts of c_0. We end up with a stage tree T_1 of depth 0 corresponding to some condition c_1, on which we have decided $\varphi(D, G)$ for every part of c_1 refining some part ν of c_0. Since we have not forked any other part than ν, the number of undecided parts strictly decreases. A condition has finitely many parts, so the process terminates after at most $|\texttt{parts}(c_0)|$ steps.

The progress becomes much less clear if T_0 is a stage tree of depth 1. When trying to decide some query on some path ν_0, ν_1 through T_0, we need to extend both the root, and the conditions below each part μ refining ν_0. The overall number of undecided paths may increase, and therefore a simple cardinality argument is not enough to deduce termination. Note that this algorithm has some common flavor with the *hydra game* introduced by Kirby and Paris [17] and whose termination is not provable in Peano arithmetic. Thankfully, our problem is much simpler and its termination can be proven by elementary means.

In Figure 5, we give an example of one step in the decision process, starting with a stage tree T_0 of depth 1 with three undecided paths, and ending up with some stage tree T_1 having four undecided paths ($c_1 - d_{1,0} - \mu_{1,1}$, $c_1 - d_{1,0} - \mu_{1,2}$, $c_1 - d_{1,0} - \mu_{1,1}$ and $c_1 - d_{1,0} - \mu_{1,2}$). Thankfully, the $\texttt{unbox}$ operator forks only the part on which it answers the query. Therefore, at the next step, we will be able to consider only one of the parts of c_1 at a time. The induced subtree has two undecided paths, so there is also some progress.

We now define some relation $\sqsubset$ between two stage trees T_0 and T_1 of depth n. It describes the relation between the stage tree T_0 and the extension T_1 obtained after applying one step of the query algorithm. More precisely, $T_2 \sqsubset T_0$ if T_2 is the subtree of T_1 on which we have removed every decided paths.

Definition 4.10 Given two stages trees $T_1 \leq T_0$ of depth n, we define the relations $T_1 \sqsubseteq T_0$ and $T_1 \sqsubset T_0$ by mutual induction as follows. If T_0 and T_1 are conditions and $T_1 \leq_f T_0$, then $T_1 \sqsubseteq T_0$ if f does not fork any part of T_0. If moreover f is not surjective, then $T_1 \sqsubset T_0$. If $T_1 = \langle c_1, h_1 \rangle$, $T_0 = \langle c_0, h_0 \rangle$ and $c_1 \leq_f c_0$, then $T_1 \sqsubseteq T_0$ if for every part ν of c_1, if f forks part $f(\nu)$ of c_0 then

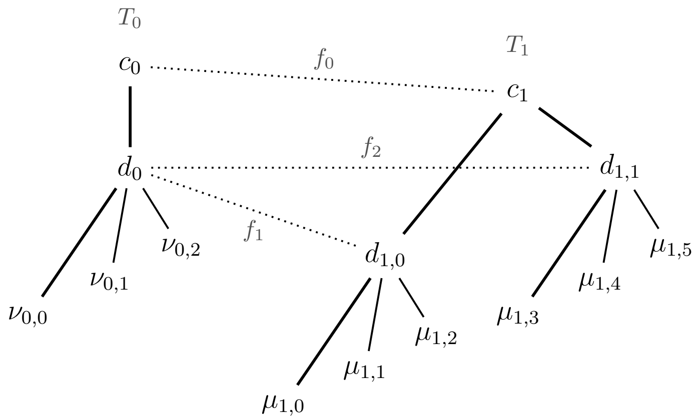

FIGURE 5. In this example, we start with a stage tree T_0 of depth 1 and want to decide some query $\varphi(D, U)$ for each of its three paths. We choose one path $\rho = c_0 - d_0 - \nu_{0,0}$, call $\texttt{box}(\nu_{0,0}, \varphi)$ to obtain a query $\psi(D)$, then call $\texttt{box}(\lambda, \psi)$, where λ is the unique part of c_0. We obtain a query $\phi(D)$, $\emptyset'$-compute some answer a to $\phi(\emptyset)$ and call $\texttt{unbox}(\lambda, a)$ to obtain some extension c_1 of c_0 and some answering function $b : \texttt{parts}(c_1) \to Ans[D, G]$. This extension forks the part λ into two parts. Below each part λ_i in c_1, we call $\texttt{unbox}(\lambda_i, b(\lambda_i))$ to obtain an extension of $d_{1,i}$ forcing $\varphi(D, G)$ below the parts refining $\nu_{0,0}$.

$h_1(\nu) \sqsubset h_0(f(\nu))$, otherwise $h_1(\nu) \sqsubseteq h_0(f(\nu))$. If moreover there is some part μ of c_0 such that $h_1(\nu) \sqsubset h_0(\mu)$ for every part ν of c_1 refining μ, then $T_1 \sqsubset T_0$.

One easily proves by mutual induction over the depth of the trees the following facts:

(i) Both $\sqsubset$ and $\sqsubseteq$ are transitive
(ii) If $T_1 \sqsubset T_0$ then $T_1 \sqsubseteq T_0$
(iii) If $T_2 \sqsubseteq T_1$ and $T_1 \sqsubset T_0$ then $T_2 \sqsubset T_0$
(iv) If $T_2 \sqsubset T_1$ and $T_1 \sqsubseteq T_0$ then $T_2 \sqsubset T_0$

Assuming that $\sqsubset$ truly represents the relation between a stage tree and its extension after one step of query, the following lemma can be understood as stating that the naive algorithm used in the proof of the query lemma terminates.

Lemma 4.11 The relation $T_1 \sqsubset T_0$ is well-founded.

Proof. By induction over the depth of the stage trees. Suppose that $T_0 \sqsupset T_1 \sqsupset \dots$ is an infinite decreasing sequence of stage trees of depth 0. In particular, the T's are conditions and $T_0 \geq_{f_0} T_1 \geq_{f_1} \dots$ for some functions f_i which are injective, but not surjective. Therefore the number of parts strictly decreases in ω, contradiction.

Suppose now that $T_0 \sqsupset T_1 \sqsupset \dots$ is an infinite decreasing sequence of stage trees of depth $n+1$, where $T_i = \langle c_i, h_i \rangle$ and $c_i \geq_{f_i} c_{i+1}$. Let S be the set of parts ν in some c_i which will fork at a later c_j. This S induces a finitely branching tree. If S is finite, then there is some j such that no part of c_k will ever fork for every $k \geq j$. By the infinite pigeonhole principle, there we can construct an infinite, decreasing sequence of trees of depth n, contradicting our induction hypothesis. So suppose that S is infinite. By König's lemma, there is an infinite sequence of parts, the later refining the former, such that they fork. Each time a conditions fork, the subtree is strictly decreasing, so we can define an infinite decreasing sequence of stage trees of depth n, again contradicting our induction hypothesis. □

Given some stage tree T_1 of depth $i < n$, a *completion of* T_1 *to* n is a stage tree T_2 of depth n such that $T_1 {\upharpoonright} i = T_0$. If $T_1 \leq T_0 {\upharpoonright} i$ for some stage tree T_0 of depth n, T_0 induces a completion T_2 of T_1 to n by setting $T_2^{[\xi]} = T_0^{[\rho]}$ for every path ξ through T_1 refining some path ρ through $T_0 {\upharpoonright} i$. One easily checks that $T_2 \leq T_0$. Such a stage tree is called the *trivial completion of* T_1

by T_0. The following technical lemma will be useful for applying the induction hypothesis in Lemma 4.14.

Lemma 4.12 Let T_0, T_1 be two stage trees of depth $n+1$ and T_2 be a stage tree of depth n and S_0 be a set of paths through $T_0{\upharpoonright}n$ such that

(i) $T_2 \sqsubseteq T_0{\upharpoonright}n$, $P(T_2) \subseteq P(T_1{\upharpoonright}n)$ and $T_1 \leq T_0$
(ii) S_0 is the set of paths through $T_0{\upharpoonright}n$ refined by some path through T_2.
(iii) For every path $\xi \in P(T_2)$, $T_1^{[\xi]} \sqsubseteq T_0^{[\rho]}$ where ξ refines the path $\rho \in S_0$
(iv) For every path $\xi \in P(T_1{\upharpoonright}n) \setminus P(T_2)$, ξ refines some path $\rho \in P(T_0{\upharpoonright}n) \setminus S_0$ and $T_1^{[\xi]} \sqsubset T_0^{[\rho]}$

Then $T_1 \sqsubseteq T_0$. Moreover, if $T_2 \sqsubset T_0{\upharpoonright}n$ then $T_1 \sqsubset T_0$.

Proof. By induction over n. In the base case, $T_0{\upharpoonright}n$, $T_1{\upharpoonright}n$ and T_2 are conditions c_0, c_1 and c_2 such that $c_2 \leq_f c_0$ and $c_1 \leq_g c_0$ for some refinement functions f and g. We easily have $T_1 \sqsubseteq T_0$ since $T_1^{[\mu]} \sqsubseteq T_0^{[g(\nu)]}$ for every part μ of c_2 (and therefore of c_1), and since $T_1^{[\mu]} \sqsubset T_0^{[g(\nu)]}$ whenever μ is a part of c_1 which is not a part of c_2. By $c_2 \sqsubseteq c_0$, the only places where a fork can happen is when μ is not in c_2.

We now want to prove that $T_1 \sqsubset T_0$ whenever $c_2 \sqsubset c_0$. Since $c_2 \sqsubset c_0$, f is injective, but not surjective. We need to prove that there is some part ν of T_1 such that $T_1^{[\nu]} \sqsubset T_0^{[g(\nu)]}$. We have two cases. In the first case, f and g have the same domain. In this case $f = g$ and since f is not surjective, there is some part of c_0 witnessing the strictness of $T_1 \sqsubset T_0$. In the second case, there is some part ν in c_1 but not c_2. By (iv), $g(\nu) \not\in S_1$. The part $g(\nu)$ of c_0 witnesses the stricteness of $T_1 \sqsubset T_0$.

In the induction case, $T_0{\upharpoonright}n = \langle c_0, h_0\rangle$, $T_1{\upharpoonright}n = \langle c_1, h_1\rangle$ and $T_2 = \langle c_2, h_2\rangle$ such that $c_2 \leq_f c_0$ and $c_1 \leq_g c_0$ for some refinement functions f and g. For every part ν in c_1, we have two cases: In the first case, ν is not in c_2. By (iv), any path ξ through $h_1(\nu)$ refines some path ρ in $h_0(g(\nu))$ such that $h_1(\nu)^{[\xi]} \sqsubset h_0(g(\nu))^{[\rho]}$. By the induction hypothesis applied to $h_0(\nu)$, $h_1(\nu)$ and the empty tree, $h_1(\nu) \sqsubset h_0(g(\nu))$. In the second case, ν is also in c_2. By the induction hypothesis applied to $h_0(\nu)$, $h_1(\nu)$ and $h_2(\nu)$, $h_1(\nu) \sqsubseteq h_0(g(\nu))$. We again easily have $T_1 \sqsubseteq T_0$ since $h_1(\nu) \sqsubseteq h_0(g(\nu))$ for every part ν in c_1 and since whenever g forks some part μ of c_0, either the parts ν of c_1 refining μ are all in c_2 in which case $h_2(\nu) \sqsubset h_0(\mu){\upharpoonright}n$ by the definition of the partial order and then we have $h_1(\nu) \sqsubset h_0(\mu)$, or none of the parts ν of c_1 refining μ are in c_2, in which case we have $h_1(\nu) \sqsubset h_0(\mu)$. By the same case analysis as in the base case, we deduce that $T_1 \sqsubset T_0$ if moreover $T_2 \sqsubset T_0{\upharpoonright}n$. ☐

Definition 4.13 (Stage tree substration) Given a stage tree T of depth n and a set S of paths through T, we define $T - S$ inductively as follows: If T is a stage tree of depth 0, then S is a set of parts of T and $T - S$ is the condition whose parts are $\texttt{parts}(T) \setminus S$. If $T = \langle c, h\rangle$ is a stage tree of depth $n+1$, then S is a set of paths of the form $\nu\rho$ where ν is a part of c and ρ is a path through $h(\nu)$. For each part ν, let $S_\nu = \{\rho : \nu\rho \in S\}$. The stage tree $T - S$ is defined by $\langle c, h_1\rangle$ where $h_1(\nu) = h(\nu) - S_\nu$ for each part ν of c.

Intuitively, $T - S$ is the maximal subtree of T such that $P(T - S) = P(T) \setminus S$. Beware, even if we may remove every part of a condition, we do not remove the condition from the tree. The following lemma uses the well-founded partial order defined previously to show that we can make some progress in deciding the queries. In what follows, the set T_0 can be thought of as the stage tree we obtain after having applied finitely many steps of query and S_0 are the paths through the tree T_0 for which we have already decided the query $\varphi(D, G)$. The lemma describes the relation between the state (T_1, S_1) obtained from (T_0, S_0) after having applied one more step.

Lemma 4.14 Let T_0 be a stage tree of depth n, S_0 be a set of paths through T_0 and let $\varphi(D, G)$ be a query. For every path $\rho \not\in S_0$ through T_0, there exists a stage tree $T_1 \leq T_0$ of depth n and a set S_1 of paths through T_1 such that

(i) $T_1 \Vdash_\xi \varphi(D, G)$ or $T_1 \Vdash_\xi \neg\varphi(D, G)$ for every path ξ through T_1 refining ρ.
(ii) $T_1 - S_1 \sqsubset T_0 - S_0$
(iii) Every path in S_1 refines either a path in S_0 or ρ.

Moreover, T_1 and the function of answers $a : P(T_1) \to Ans[D, G]$ can be $\emptyset'$-effectively computed uniformly in T_1 and $\varphi(D, G)$.

Proof. By induction over n. If T_0 is a stage tree of depth 0, then it is a condition c_0 and the paths through T_0 are the parts of c_0. Let ν be such a part. Let $\psi(D)$ be the query $\texttt{box}(\nu, \varphi)$. We can $\emptyset'$-compute an answer a_0 to $\psi(\emptyset)$. Let $\langle c_1, f, a\rangle = \texttt{unbox}(\nu, a_0)$ be such that $c_1 \leq_f c_0$, f forks only part ν of c_0 and for every part μ of c_1 such that $f(\mu) = \nu$, $c_1 \Vdash_\mu \varphi(D, G)$ or $c_1 \Vdash_\mu \neg\varphi(D, G)$ and $a(\mu)$ answers $\varphi(D, G)$ accordingly. Take $S_1 = \{\mu : f(\mu) = \nu \vee f(\mu) \in S_0\}$. The property (i) holds by definition of c_1 and (iii) holds by definition of S_1. Since the only forked part is ν and no part of $c_1 - S_1$ refines ν, $c_1 - S_1 \sqsubset c_0 - S_0$, so the property (ii) also holds. This completes the base case.

Suppose now that T_0 is a stage tree of depth $n+1$. The paths through T_0 are of the form $\rho\nu$ where ρ is a path through $T_0{\upharpoonright}n$ and ν is a part of the root of $T_0^{[\rho]}$. Fix any such path. Let $\psi(D)$ be the query $\texttt{box}(\nu, \varphi)$ and let $\phi(D, G)$ be the formula $\psi(D \oplus G)$. By induction hypothesis on $T_0{\upharpoonright}n$, there is a stage tree $T_2 \leq T_0{\upharpoonright}n$ and a set S_2 such that

(i) $T_2 \Vdash_\xi \phi$ or $T_2 \Vdash_\xi \neg\phi$ for every path ξ through T_2 refining ρ
(ii) $T_2 - S_2 \sqsubset T_0{\upharpoonright}n - S_0{\upharpoonright}n$
(iii) Every path in S_2 refines either a path in $S_0{\upharpoonright}n$ or ρ.

Moreover, still by induction hypothesis, we have a function $a : P(T_2) \to Ans[D, G]$ answering the queries. We define a completion of T_2 into a stage tree T_1 of depth $n+1$ as follows: For each path ξ through T_2 refining ρ, let $T_1^{[\xi]}$ be the condition c_ξ such that $\langle c_\xi, f_\xi, a_\xi\rangle = \texttt{unbox}(\nu, a(\xi))$. For each path ξ through T_2 which refines some path τ through T_0 different from ρ, let $T_1^{[\xi]} = T_0^{[\tau]}$. By construction, $T_1 \leq T_0$ since c_ξ f_ξ-refines $T_0^{[\rho]}$ whenever ξ refines ρ and since any condition refines itself. Let S_1 be the collection of paths $\xi\mu$ through T_1 such that $\xi \in S_2$ and either ξ refines ρ and $f_\xi(\mu) = \nu$, or $\xi\mu$ refines a path in S_0. Since $(T_0{\upharpoonright}n - S_0{\upharpoonright}n) \sqsubseteq (T_0 - S_0){\upharpoonright}n$, we have $T_2 - S_2 \sqsubset (T_0 - S_0){\upharpoonright}n$. We can therefore apply Lemma 4.12 to $T_0 - S_0$, $T_1 - S_1$, and $T_2 - S_2$, to obtain $T_1 - S_1 \sqsubset T_0 - S_0$. Define the answer function $b : P(T_1) \to Ans[D, G]$ by $b(\xi\mu) = a_\xi(\mu)$ for each path ξ through T_2 refining ρ. This function b is found $\emptyset'$-effectively since the $\texttt{unbox}$ operator is computable. ☐

The following lemma simply iterates Lemma 4.14 and uses the well-foundedness of the relation $\sqsubset$ to deduce that we can find some extension on which the queries are decided for every path.

Lemma 4.15 Let T_0 be a stage tree of depth n and let $q : P(T_0) \to Query[D, G]$ be a function. There is a stage tree $T_1 \leq T_0$ of depth n such that $T_1 \Vdash_\xi q(\rho)$ or $T_1 \Vdash_\xi \neg q(\rho)$ for every path ξ through T_1 refining some path ρ through T_0. Moreover, T_1 and the function of answers $a : P(T_1) \to Ans[D, G]$ can be $\emptyset'$-effectively computed uniformly in T_1 and q.

Proof. Using Lemma 4.14, define a sequence of tuples $\langle T_0, S_0, \rho_0, \tau_0\rangle, \langle T_1, S_1, \rho_1, \tau_1\rangle, \dots$ starting with T_0, $S_0 = \emptyset$, $\rho_0 = \tau_0 \in P(T_0)$ and such that for each i

(i) $T_{i+1} \leq T_i$ is a stage tree of depth n, S_i is a set of paths through T_i
(ii) ρ_i is a path through $T_i - S_i$ refining the path τ_i through T_0.
(iii) $T_{i+1} \Vdash_\xi q(\tau_i)$ or $T_{i+1} \Vdash_\xi \neg q(\tau_i)$ for every path ξ through T_{i+1} refining ρ_i
(iv) $T_{i+1} - S_{i+1} \sqsubset T_i - S_i$
(v) Every path in S_{i+1} refines either a path in S_i or ρ_i.

By Lemma 4.11, the relation $\sqsubset$ is well-founded, so the sequence has to be finite by (iv). Let k be the maximal index of the sequence. By maximality of k and by Lemma 4.14, $P(T_k) - S_k = \emptyset$. Therefore, $P(T_k) = S_k$. Since $S_0 = \emptyset$ and by (v), we can prove by induction over k that for every path ξ through T_k, there is some stage $i < k$ such that ξ refines ρ_i. Thus, by (iii) and by stability of the forcing relation under refinement, $T_k \Vdash_\xi q(\tau_i)$ or $T_k \Vdash_\xi \neg q(\tau_i)$. Therefore

T_k satisfies the statement of the lemma. The uniformity is inherited from the uniformity of Lemma 4.14. □

Last, we prove the query lemma by iterating the previous lemma at every depth of the stage tree, to decide the queries on the partial paths.

Proof of the query lemma. Let T_0 be a stage tree of depth n and $q : PP(T_0) \to Query[U, G]$ be a function. Using Lemma 4.15, define a decreasing sequence of stage trees $T_0 \geq \cdots \geq T_n$ of depth n such that for each $i < n$,

(i) T_{i+1} is the trivial completion of $T_{i+1}{\upharpoonright}i + 1$ by T_i.
(ii) $T_{i+1} \Vdash_\xi q(\tau)$ or $T_{i+1} \Vdash_\xi \neg q(\tau)$ for every path ξ through $T_{i+1}{\upharpoonright}i + 1$ refining some path τ through $T_0{\upharpoonright}i + 1$.

To do this, at stage $i < n$, apply Lemma 4.15 to T_i with the query function $r : PP(T_i) \to Query[U, G]$ defined by $r(\rho) = q(\tau)$ for each path ρ through $T_i{\upharpoonright}i + 1$ refining some path τ through $T_0{\upharpoonright}i + 1$. Since the forcing relation is stable by refinement, the stage tree T_n satisfies the statement of the query lemma. The uniformity is again inherited from the uniformity of Lemma 4.15. □

This completes the presentation of the framework. We will now define a module for the Erdős-Moser theorem. In section 6, we will see how to compose modules to obtain stronger separations.

5. The weakness of EM over ω-models

Now we have settled the domination framework, it suffices to implement the abstract module to obtain ω-structures which do not satisfy AMT. We have illustrated the notion of module by implementing one for COH. An immediate consequence is the existence of an ω-model of COH which is not a model of AMT. In this section, we shall extend this separation to the Erdős-Moser theorem. As noted before, every ω-model of EM which is not a model of AMT is also a model of COH. This section is devoted to the proof of the following theorem.

Theorem 5.1 There exists an ω-model of EM which is not a model of AMT.

At first sight, the forcing notion introduced in section 3 seems to have a direct mapping to the abstract notion of forcing defined in the domination framework. However, unlike cohesiveness where the module implementation was immediate, the Erdős-Moser theorem raises new difficulties:

– The Erdős-Moser theorem is not known to admit a universal instance. We will therefore need to integrate the information about the instance in the notion of condition. Moreover, the `init` operator will have to choose accordingly some new instance of EM at every iteration level. We need to make `init` computable, but the collection of every infinite computable tournament functionals is not even computably enumerable.
– The notion of EM condition introduced in section 3 contains a $\Pi^{0,R}_1$ property ensuring extensibility. Since the tournament R depends on the previous iteration which is being constructed, we have only access to a finite part of R. We need therefore to ensure that whatever the extension of the finite tournament is, the condition will be extendible.

We shall address the above-mentioned problems one at a time in subsections 5.1 and 5.2.

5.1. Enumerating the infinite tournaments

In section 3, we were also confronted to the problem of enumerating all infinite tournaments and solved it by relativizing the construction to a low subuniform degree in order to obtain a low sequence of infinite tournaments containing at least every infinite computable tournament. We cannot apply the same trick to handle the construction of an ω-model of EM as solutions to some computable tournaments may bound new tournaments and so on. However, as we shall see, we can restrict ourselves to primitive recursive tournaments to generate an ω-model of EM.

Given a sequence of sets $X_0, X_1, \dots$, define $\mathcal{M}_{\vec{X}}$ to be the ω-structure whose second-order part is the Turing ideal generated by $\vec{X}$, that is,

$$\{Y \in 2^\omega : (\exists i)[Y \leq_T X_0 \oplus \dots \oplus X_i]\}$$

Lemma 5.2 There exists a uniformly computable sequence of infinite, primitive recursive tournament functionals $T_0, T_1, \dots$ such that for every sequence of sets $X_0, X_1, \dots$ such that X_i is an infinite transitive subtournament of $T_i^{X_0 \oplus \dots \oplus X_{i-1}}$ for each $i \in \omega$,

$$\mathcal{M}_{\vec{X}} \not\models \mathsf{AMT} \to \mathcal{M}_{\vec{X}} \models \mathsf{EM} \wedge \mathsf{COH}$$

Proof. As $\mathsf{RCA}_0 \vdash \mathsf{SEM} \wedge \mathsf{COH} \to \mathsf{EM}$, it suffices to prove that for every set X,

(i) for every stable, infinite, X-computable tournament R, there exists an infinite X-p.r. tournament T such that every infinite T-transitive subtournament X-computes an infinite R-transitive subtournament.

(ii) for every X-computable complete atomic theory T and every uniformly X-computable sequence of sets $\vec{R}$, there exists an infinite X-p.r. tournament such that every infinite transitive subtournament X-computes either an $\vec{R}$-cohesive set or an atomic model of T.

(i) Fix a set X and a stable, infinite, X-computable tournament R. Let $\tilde{f} : \omega \to 2$ be the X'-computable function defined by $\tilde{f}(x) = 0$ if $(\forall^\infty s)R(s,x)$ and $\tilde{f}(x) = 1$ if $(\forall^\infty s)R(x,s)$. By Schoenfield's limit lemma [28], there exists an X-p.r. function $g : \omega^2 \to 2$ such that $\lim_s g(x,s) = \tilde{f}(x)$ for every $x \in \omega$. Considering the X-p.r. tournament T such that $T(x,y)$ holds iff $x < y$ and $g(x,y) = 1$ or $x > y$ and $g(x,y) = 0$, every infinite T-transitive subtournament X-computes an infinite R-transitive subtournament.

(ii) Jockusch and Stephan proved in [16] that for every set X, and every uniformly X-computable sequence of sets $\vec{R}$, every p-cohesive set relative to X computes an $\vec{R}$-cohesive set. The author proved in [25] that for every X-computable complete atomic theory T, there exists an X'-computable coloring $f : \omega \to \omega$ such that every infinite set Y *thin* for f (i.e. such that $f(Y) \neq \omega$) X-computes an atomic model of T. He also proved that for every such X'-computable coloring $f : \omega \to \omega$, there exists an infinite, X-p.r. tournament R such that every infinite transitive subtournament is either p-cohesive, or X-computes an infinite set thin for f. □

We can therefore fix this computable enumeration $T_0, T_1, \dots$ of tournament functionals, and make `init`(n) return an empty condition paired with T_n. Thus, taking at each iteration an infinite set satisfying one of the parts, we obtain an ω-model of EM.

5.2. The new Erdős-Moser conditions

Fix some primitive recursive tournament functional R. According to the analysis of the Erdős-Moser presented in section 3, we would like to define the forcing conditions to be tuples $(\vec{F}, \mathcal{C})$ where

(a) $\mathcal{C}$ is a non-empty $\Pi^{0,D}_1$ k-cover class of $[t, +\infty)$ for some $k, t \in \omega$

(b) $F_\nu \cup \{x\}$ is R^D-transitive for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, every $x \in Z_\nu$ and each $\nu < k$

(c) Z_ν is included in a minimal R^D-interval of F_ν for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ and each $\nu < k$.

However, at a finite stage, we have only access to a finite part of D, and therefore we cannot express the properties (a-c). Indeed, we may have made some choices about the F's such that $F_\nu \cup \{x\}$ is not R^D-transitive for every part ν, every D satisfying the previous iterations and cofinitely many $x \in \omega$. We need therefore to choose the F's carefully enough so that whatever the extension of the finite tournament to which we have access, we will be able to extend at least one of the F's.

The initial condition $(\{\emptyset\}, \{\omega\})$ satisfies the properties (a-c) no matter what D is, since $\{\omega\}$ does not depend on D. Let us have a closer look at the question $Q2$ asked in section 3. For the sake of simplification, we will consider that the question is asked below the unique part of the initial condition. It therefore becomes:

Q3: Is there a finite set $E \subseteq \omega$ such that for every 2-partition $\langle E_0, E_1\rangle$ of E, there exists an R^D-transitive subset $F_1 \subseteq E_i$ for some $i < 2$ such that $\varphi(D, F_1)$ holds?

Notice that this is a syntactic question since it depends on the purely formal variable D representing the effective join of the sets constructed in the previous iterations. Thanks to the usual query process, we are able to transform it into a concrete Σ^0_1 formula getting rid of the formal parameter D, and obtain some answer that the previous layers guarantee to hold for every set D satisfying the previous iterations.

If the answer is negative, then by compactness, for every set D satisfying the previous iterations, there is a 2-partition $Z_0 \cup Z_1 = \omega$ such that for every $i < 2$ and every R^D-transitive subset $G \subseteq Z_i$, $\varphi(D, G)$ does not hold. For every set D, the $\Pi^{0,D}_1$ class $\mathcal{C}$ of such 2-partitions $Z_0 \oplus Z_1$ is therefore guaranted to be non-empty. Note again that since D is a syntactic variable, the class $\mathcal{C}$ is also syntactic, and purely described by finite means.

If the answer is positive, then we are given some finite set $E \subseteq \omega$ witnessing it. Moreover, we are guaranted that for every set D satisfying the previous iterations, for every 2-partition $\langle E_0, E_1\rangle$ of E, there exists an R^D-transitive subset $F_1 \subseteq E_i$ for some $i < 2$ such that $\varphi(D, F_1)$ holds. In we knew the set D, we would choose one "good" 2-partition $\langle E_0, E_1\rangle$ as we do in section 3. However, this choice depends on infinitely many bits of information of D. We will need therefore to try every 2-partition in parallel.

There is one more difficulty. With this formulation, we are not able to find the desired extension, since D is syntactic, and therefore we do not know how to identify the color i and the actual set F_1 given some 2-partition $\langle E_0, E_1\rangle$. Thankfully, we can slightly modify the question to ask to provide the witness F_1 for each such a partition in the answer.

Q4: Is there a finite set $E \subseteq \omega$ and a finite function g such that for every 2-partition $\langle E_0, E_1\rangle$ of E, $g(\langle E_0, E_1\rangle) = F_1$ for some $i < 2$ and some R^D-transitive subset $F_1 \subseteq E_i$ such that $\varphi(D, F_1)$ holds?

The question $Q4$ is equivalent to the question $Q3$, but provides a constructive witness g in the case of a positive answer as well. We can even formulate the question so that we know the relation R^D over the set F_1. Thus we are able to talk about minimal R^D-intervals of F_1.

Now, we can extend the initial condition $(\{\emptyset\}, \{\omega\})$ into some condition $(\vec{F}, \mathcal{C})$ as follows: For each 2-partition $\langle E_0, E_1\rangle$ of E, letting $F_1 = g(\langle E_0, E_1\rangle)$, for every minimal R^D-interval I, we create a part $\nu = \langle E_0, E_1, I\rangle$ and set $F_\nu = F_1$. Take some $t' > max(\vec{F})$ and let $\mathcal{C}$ be the $\Pi^{0,D}_1$ class of covers $\bigoplus_\nu Z_\nu$ of $[t', +\infty)$ such that for every part $\nu = \langle E_0, E_1, I\rangle$

(b') $F_\nu \cup \{x\}$ is R^D-transitive for every $x \in Z_\nu$

(c') Z_ν is included in the minimal R^D-interval I

Fix some set D satisfying the previous iterations. We claim that $\mathcal{C}$ is non-empty. Any element $x \in [t', +\infty)$ induces a 2-partition $g(x) = \langle E_0, E_1\rangle$ of E by setting $E_0 = \{y \in E : R^D(y, x)\}$ and $E_1 = \{y \in E : R^D(x, y)\}$. On the other hand, for every 2-partition $\langle E_0, E_1\rangle$ of E, we can define a partition of $[t', +\infty)$ by setting $Z_{\langle E_0,E_1\rangle} = \{x \in [t', +\infty) : g(x) = \langle E_0, E_1\rangle\}$. By definition, $E_0 \to_{R^D} Z_{\langle E_0,E_1\rangle} \to_{R^D} E_1$. Therefore, the cover $\bigoplus_\nu Z_\nu$ of $[t', +\infty)$ defined by

$$Z_\nu = \begin{cases} Z_{\langle E_0,E_1\rangle} & \text{if } \nu = \langle E_0, E_1, I\rangle, I = (max(F_\nu), +\infty) \text{ and } F_\nu \subseteq E_0 \\ Z_{\langle E_0,E_1\rangle} & \text{if } \nu = \langle E_0, E_1, I\rangle, I = (-\infty, min(F_\nu)) \text{ and } F_\nu \subseteq E_1 \\ \emptyset & \text{otherwise} \end{cases}$$

is in $\mathcal{C}$ and witnesses the non-emptiness of $\mathcal{C}$.

The problem of having access to only a finite part of the class $\mathcal{C}$ appears more critically when considering the question below some part ν of an arbitrary condition $c = (\vec{F}, \mathcal{C})$. The immediate generalization of the question $Q4$ is the following.

Q5: For every cover $X_0 \oplus \cdots \oplus X_{k-1} \in \mathcal{C}$, is there a finite set $E \subseteq X_\nu$ and a finite function g such that for every 2-partition $\langle E_0, E_1\rangle$ of E, $g(\langle E_0, E_1\rangle)$ is a finite R^D-transitive subset of some E_j such that $\varphi(D, F_\nu \cup g(\langle E_0, E_1\rangle))$ holds?

As usual, although this question is formulated in a Π^0_2 manner, it can be turned into a $\Sigma^{0,D}_1$ query using a compactness argument.

Q5': Is there some $r \in \omega$, a finite sequence of finite sets $E^0, \dots, E^{r-1}$ and a finite sequence of functions $g^0, \dots, g^{r-1}$ such that

(1) *for every $X_0 \oplus \dots \oplus X_{k-1} \in \mathcal{C}$, there is some $i < r$ such that $E^i \subseteq X_\nu$*
(2) *for every $i < r$ and every 2-partition $\langle E_0, E_1 \rangle$ of E^i, $g^i(\langle E_0, E_1 \rangle)$ is a finite R^D-transitive subset of some E_j such that $\varphi(D, F_\nu \cup g^i(\langle E_0, E_1 \rangle))$ holds?*

In the case of a negative answer, we can apply the standard procedure consisting in refining the $\Pi^{0,D}_1$ class $\mathcal{C}$ into some $\Pi^{0,D}_1$ class $\mathcal{D}$ forcing $\varphi(D, G)$ not to hold on every part refining the part ν in c. The class $\mathcal{D}$ is non-empty since we can construct a member of it from a witness of failure of $Q5$. The problem appears when the answer is positive. We are given some finite sequence $E^0, \dots, E^{r-1}$ and a finite sequence of functions $g^0, \dots, g^{r-1}$ satisfying (i) and (ii). For every D, there is some $X_0 \oplus \dots \oplus X_{k-1} \in \mathcal{C}$ and some $i < r$ such that $E^i \subseteq X_\nu$, but this i may depend on D. We cannot choose some E^i as we used to do in section 3.

Following our moto, if we are not able to make a choice, we will try every possible case in parallel. The idea is to define a condition $d = (\vec{E}, \mathcal{D})$ and a refinement function f forking the part ν into various parts, each one representing a possible scenario. For every part μ of c which is different from ν, create a part μ in d and set $E_\mu = F_\mu$. For every $i < r$ and every 2-partition $\langle E_0, E_1 \rangle$ of E^i, create a part $\mu = \langle i, E_0, E_1 \rangle$ in d refining ν and set $E_\mu = F_\nu \cup g^i(\langle E_0, E_1 \rangle)$. Accordingly, let $\mathcal{D}$ be the $\Pi^{0,D}_1$ class of covers $\bigoplus_\mu Y_\mu$ of $[t, +\infty)$ such that there is some $i < r$ and some cover $X_0 \oplus \dots \oplus X_{k-1} \in \mathcal{C}$ satisfying first $E^i \subseteq X_\nu$, second $Y_\mu \subseteq X_{f(\mu)}$ for each part μ of d and third $Y_\mu = \emptyset$ if $\mu = \langle j, E_0, E_1 \rangle$ for some $j \neq i$.

The class $\mathcal{D}$ f-refines $\mathcal{C}$, but does not f-refine $\mathcal{C}^{[\nu, E^i]}$ for some fixed $i < r$. Because of this, the condition d does not extends the condition c in the sense of section 3. We shall therefore generalize the operator $\cdot \mapsto \mathcal{C}^{[\nu, \cdot]}$ to define it over tuples of sets.

Restriction of a cover class. Given some cover class $(k, Y, \mathcal{C})$, some part ν of $\mathcal{C}$ and some r-tuple $E^0, \dots, E^{r-1}$ of finite sets, we denote by $\mathcal{C}^{[\nu, \vec{E}]}$ the cover class $(k + r - 1, Y, \mathcal{D})$ such that $\mathcal{D}$ is the collection of

$$X_0 \oplus \dots \oplus X_{\nu-1} \oplus Z_0 \oplus \dots \oplus Z_{r-1} \oplus X_{\nu+1} \oplus \dots \oplus X_{k-1}$$

such that $X_0 \oplus \dots \oplus X_{k-1} \in \mathcal{D}$ and there is some $i < r$ such that $E^i \subseteq X_\nu$, $Z_i = X_\nu$ and $Z_j = \emptyset$ for every $j \neq i$. In particular, $\mathcal{C}^{[\nu, \vec{E}]}$ refines $\mathcal{C}$ with some refinement function f which forks the part ν into r different parts. Such a function f is called the *refinement function witnessing the restriction.*

We need to define the notion of extension between conditions accordingly. A condition $d = (\vec{E}, \mathcal{D})$ *extends* a condition $c = (\vec{F}, \mathcal{C})$ (written $d \leq c$) if there is a function $f : parts(\mathcal{D}) \to parts(\mathcal{C})$ such that the following holds:

(i) $(E_\nu, dom(\mathcal{D}))$ Mathias extends $(F_{f(\nu)}, dom(\mathcal{C}))$ for each $\nu \in parts(\mathcal{D})$
(ii) Every $\bigoplus_\mu Y_\mu \in \mathcal{D}$ f-refines some $\bigoplus_\nu X_\nu \in \mathcal{C}$ such that for each part μ of d, either $E_\mu \setminus H_{f(\mu)} \subseteq X_{f(\mu)}$, or $Y_\mu = \emptyset$.

Note that this notion of extension is coarser than the one defined in section 3. Unlike with the previous notion of extension, there may be from now on some part μ of d refining the part ν of c, such that (E_μ, Y_μ) does not Mathias extend (F_ν, X_ν) for some $\bigoplus_\mu Y_\mu \in \mathcal{D}$ and every $\bigoplus_\nu X_\nu \in \mathcal{C}$, but in this case, we make (E_μ, Y_μ) non-extendible by ensuring that $Y_\mu = \emptyset$.

5.3. **Implementing the Erdős-Moser module**

We are now ready to provide a concrete implementation of a module support and a module for EM. Define the tuple $\mathbb{S}^{\mathsf{EM}} = \langle \mathbb{P}, \mathbb{U}, \mathtt{parts}, \mathtt{init}, \mathtt{sat} \rangle$ as follows: $\mathbb{P}$ is the collection of all conditions $(\vec{F}, \mathcal{C}, R)$ where R is a primitive recursive tournament functional and

(a) $\mathcal{C}$ is a non-empty $\Pi^{0,D}_1$ k-cover class of $[t, +\infty)$ for some $k, t \in \omega$
(b) $F_\nu \cup \{x\}$ is R^D-transitive for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$, every $x \in Z_\nu$ and each $\nu < k$
(c) Z_ν is included in a minimal R^D-interval of F_ν for every $Z_0 \oplus \dots \oplus Z_{k-1} \in \mathcal{C}$ and each $\nu < k$.

Once again, $\mathcal{C}$ is actually a $\Pi^{0,D}_1$ formula denoting a non-empty $\Pi^{0,D}_1$ class. A condition $d = (\vec{E}, \mathcal{D}, T)$ *extends* $c = (\vec{F}, \mathcal{C}, R)$ (written $d \leq c$) if $R = T$ and there exists a function $f : parts(\mathcal{D}) \to parts(\mathcal{C})$ such that the properties (i) and (ii) mentioned above hold.

Given some condition $c = (\vec{F}, \mathcal{C}, R)$, $\texttt{parts}(c) = \{\langle c, \nu\rangle : \nu \in parts(\mathcal{C})\}$. Define $\mathbb{U}$ as $\bigcup_{c \in \mathbb{P}} \texttt{parts}(c)$, that is, the set of all pairs $\langle (\vec{F}, \mathcal{C}, R), \nu\rangle$ where $\nu \in parts(\mathcal{C})$. The operator $\texttt{init}(n)$ returns the condition $(\{\emptyset\}, \{\omega\}, R_n)$ where R_n is the nth primitive recursive tournament functional. Last, define $\texttt{sat}(\langle c, \nu\rangle)$ to be the collection of all R^D-transitive subtournaments satisfying the Mathias precondition (F_ν, X_ν) where X_ν is *non-empty* for some $\bigoplus_\nu X_\nu \in \mathcal{C}$. The additional non-emptiness requirement of X_ν in the definition of the $\texttt{sat}$ operator enables us to "disable" some part by setting $X_\nu = \emptyset$. Without this requirement, the property (i) of a module support would not be satisfied. Moreover, since every cover class has an acceptable part, there is always one part ν in $\mathcal{C}$ such that $\texttt{sat}(\langle c, \nu\rangle) \neq \emptyset$.

Lemma 5.3 The tuple $\mathbb{S}^{\mathsf{EM}}$ is a module support.

Proof. We must check that if $d \leq_{\mathbb{P}} c$ for some $c, d \in \mathbb{P}$, then there is a function $g : \texttt{parts}(d) \to \texttt{parts}(c)$ such that $\texttt{sat}(\nu) \subseteq \texttt{sat}(g(\nu))$ for each $\nu \in \texttt{parts}(d)$. Let $d = (\vec{E}, \mathcal{D}, R)$ and $c = (\vec{F}, \mathcal{C}, R)$ be such that $d \leq_{\mathbb{P}} c$. By definition, there is a function $f : parts(\mathcal{D}) \to parts(\mathcal{C})$ satisfying the properties (i-ii). Let $g : \texttt{parts}(d) \to \texttt{parts}(c)$ be defined by $g(\langle d, \nu\rangle) = \langle c, f(\nu)\rangle$. We claim that g is a refinement function witnessing $d \leq_{\mathbb{P}} c$. Let G be any set in $\texttt{sat}(\langle d, \nu\rangle)$. We will prove that $G \in \texttt{sat}(\langle c, f(\nu)\rangle)$. The set G is an R^D-transitive subtournament satisfying the Mathias condition (E_ν, X_ν) where $X_\nu \neq \emptyset$ for some $\bigoplus_\nu X_\nu \in \mathcal{D}$. By (ii), since X_ν is non-empty, there is some $\bigoplus_\mu Y_\mu \in \mathcal{C}$ such that $E_\nu \setminus F_{f(\nu)} \subseteq Y_{f(\nu)}$ and $X_\nu \subseteq Y_{f(\nu)}$. It suffices to show that (F_ν, X_ν) Mathias extends $(F_{f(\nu)}, Y_{f(\nu)})$ to deduce that G satisfies the Mathias condition $(F_{f(\nu)}, Y_{f(\nu)})$ and finish the proof. By (i), $F_{f(\nu)} \subseteq E_\nu$. Since $E_\nu \setminus F_{f(\nu)} \subseteq Y_{f(\nu)}$ and $X_\nu \subseteq Y_{f(\nu)}$, we are done. □

We next define an implementation of the module $\mathbb{M}^{\mathsf{EM}} = \langle \mathbb{S}^{\mathsf{EM}}, \texttt{box}, \texttt{unbox}, \texttt{prog}\rangle$ as follows. Given some condition $c = (\vec{F}, \mathcal{C}, R)$, some $\nu \in parts(\mathcal{C})$ and some Σ^0_1 formula $\varphi(D, G)$, $\texttt{unbox}(\langle c, \nu\rangle, \varphi)$ returns the Σ^0_1 formula $\psi(D)$ which holds if there is a finite sequence of finite sets $E^0, \dots, E^{r-1}$ and a finite sequence of functions $g^0, \dots, g^{r-1}$ such that

1. for every $X_0 \oplus \dots \oplus X_{k-1} \in \mathcal{C}$, there is some $i < r$ such that $E^i \subseteq X_\nu$
2. for every $i < r$ and every 2-partition $\langle E_0, E_1\rangle$ of E^i, $g^i(\langle E_0, E_1\rangle)$ is a finite R^D-transitive subset of some E_j such that $\varphi(D, F_\nu \cup g^i(\langle E_0, E_1\rangle))$ holds.

If the answer to $\psi(D)$ is $\langle \texttt{No}\rangle$, $\texttt{unbox}(\langle c, \nu\rangle, \langle \texttt{No}\rangle)$ returns the tuple $\langle d, f, b\rangle$ where $d = (\vec{E}, \mathcal{D}, R)$ is a condition such that $d \leq_f c$ and defined as follows. For every part $\mu \neq \nu$ of c, create a part μ in d and set $E_\mu = F_\mu$. Furthermore, fork the part ν into two parts ν_0 and ν_1 in d and set $E_{\nu_i} = F_\nu$ for each $i < 2$. Define $\mathcal{D}$ to be the $\Pi^{0,D}_1$ class of all covers $\bigoplus_\mu Y_\mu$ f-refining some cover $\bigoplus_\nu X_\nu \in \mathcal{C}$ and such that for every $i < 2$ and every finite R^D-transitive set $E \subseteq Y_{\nu_i}$, $\varphi(D, F_\nu \cup E)$ does not hold. Moreover, $b : \texttt{parts}(c) \to Ans[D, G]$ is the constant function $\langle \texttt{No}\rangle$.

Suppose now that the answer to $\psi(D)$ is $a = \langle \texttt{Yes}, r, E^0, \dots, E^{r-1}, f^0, \dots, f^{r-1}, a'\rangle$ where a' is a function which on every $i < r$ and every 2-partition $\langle E_0, E_1\rangle = i$, returns an answer to $\varphi(D, F_\nu \cup g^i(\langle E_0, E_1\rangle))$. The function $\texttt{unbox}(\langle c, \nu\rangle, a)$ returns the tuple $\langle d, f, b\rangle$ where d is a condition such that $d \leq_f c$ and whose definition has been described in subsection 5.2. The function $b : \texttt{parts}(d) \to Ans[D, G]$ returns on every part $\mu = \langle i, E_0, E_1\rangle$ the tuple $\langle \texttt{Yes}, a'(i, E_0, E_1)\rangle$.

Last, given some condition $c = (\vec{F}, \mathcal{C}, R)$ and some $\nu \in parts(\mathcal{C})$, $\texttt{prog}(\langle c, \nu\rangle)$ is the query $\varphi(D, G) = (\exists n)[n \in G \wedge n > max(F_\nu)]$. Note that we cannot force $\neg\varphi(D, G)$ on every part $\langle c, \nu\rangle$, since every cover class has an acceptable part. Applying the query lemma infinitely many times on the progress operator ensures that if we take any path through the infinite tree of the acceptable parts, the resulting R^D-transitive subtournament will be infinite.

Lemma 5.4 The tuple $\mathbb{M}^{\mathsf{EM}}$ is a module.

Proof. We need to ensure that given some part ν of some condition $c = (\vec{F}, \mathcal{C}, R)$ and some answer a to a Σ^0_1 formula $\psi(D) = \mathtt{box}(\langle c, \nu \rangle, \varphi)$ where $\varphi(D, G)$ is a Σ^0_1 formula, $\mathtt{unbox}(\langle c, \nu \rangle, a)$ outputs a tuple $\langle d, f, b \rangle$ where $d = (\vec{E}, \mathcal{D}, R)$ is a condition such that $d \leq_f c$ where f forks only part ν of c, and for every part μ of d such that $f(\langle d, \mu \rangle) = \langle c, \nu \rangle$, and every set $G \in \mathtt{sat}(\langle d, \mu \rangle)$, $b(\langle d, \mu \rangle)$ is an answer to $\varphi(D, G)$.

Suppose that $a = \langle \mathtt{No} \rangle$. By definition of $\mathtt{sat}(\langle d, \mu \rangle)$ and by construction of d, G is R^D-transitive and satisfies the Mathias condition (E_{ν_i}, Y_{ν_i}) for some $i < 2$ and some cover $\bigoplus_\mu Y_\mu \in \mathcal{D}$. In particular, $E_{\nu_i} = F_\nu$ and Y_{ν_i} is such that for every finite R^D-transitive set $E \subseteq Y_{\nu_i}$, $\varphi(D, F_\nu \cup E)$ does not hold. In particular, taking $E = G \setminus F_\nu$, $\varphi(D, G)$ does not hold.

Suppose now that $a = \langle \mathtt{Yes}, r, E^0, \dots, E^{r-1}, f^0, \dots, f^{r-1}, a' \rangle$ where a' is a function which on every $i < r$ and every 2-partition $\langle E_0, E_1 \rangle = i$, returns an answer to $\varphi(D, F_\nu \cup g^i(\langle E_0, E_1 \rangle))$. By definition of $\mathtt{sat}(\langle d, \mu \rangle)$ and by construction of d, G is R^D-transitive and satisfies the Mathias condition (E_μ, Y_μ) for some cover $\bigoplus_\mu Y_\mu \in \mathcal{D}$, where $\mu = \langle i, E_0, E_1 \rangle$. By construction of d, $E_\mu = F_\nu \cup g^i(\langle E_0, E_1 \rangle)$, and by definition of g^i, $\varphi(D, F_\nu \cup g^i(\langle E_0, E_1 \rangle))$. Since G satisfies (E_μ, Y_μ), $F_\nu \cup g^i(\langle E_0, E_1 \rangle) \subseteq G$ and $G \setminus E_\mu \subseteq Y_\mu$. Therefore $\varphi(D, G)$ holds. □

5.4. The separation

We have defined a module $\mathbb{M}^{\mathsf{EM}}$ for the Erdős-Moser theorem. In this subsection, we explain how we create an ω-model of EM which is not a model of AMT from the infinite sequence of stage trees constructed in subsection 4.3. Given the uniform enumeration $R_0, R_1, \dots$ of all primitive recursive tournament functionals, we shall define an infinite sequence of sets $X_0, X_1, \dots$ together with a Δ^0_2 function f such that for every s,

1. X_{s+1} is an infinite, transitive subtournament of $R^{X_0 \oplus \dots \oplus X_s}$
2. f dominates every $X_0 \oplus \dots \oplus X_s$-computable function.

By 2, any Δ^0_2 approximation $\tilde{f}$ of the function f is a computable instance of the escape property with no solution in $\mathcal{M}_{\vec{X}}$, that is, such that no function in $\mathcal{M}_{\vec{X}}$ escapes f. By the computable equivalence between the escape property and the atomic model theorem (see subsection 1.3), $\mathcal{M}_{\vec{X}} \not\models \mathsf{AMT}$. By Lemma 5.2, $\mathcal{M}_{\vec{X}} \models \mathsf{EM} \wedge \mathsf{COH}$.

Start with $X_0 = \emptyset$ and the Δ^0_2 enumeration $T_0 \geq T_1 \geq \dots$ of stage trees constructed in subsection 4.3, and let $c_0 \geq c_1 \geq \dots$ be the sequence of their roots. The set U of their parts form an infinite, finitely branching tree, whose structure is given by the refinement functions. Moreover, by the construction of the sequence $T_0, T_1, \dots$, for every s, there is some part ν in c_{s+1} refining some part μ in c_s and which forces $\mathtt{prog}(\mu)$. Call such a part ν a *progressing part.* We may also consider that every part of c_0 is a progressing part, for the sake of uniformity. By the implementation of $\mathtt{prog}$, if ν is a progressing part which refines some part μ, μ is also a progressing part. Therefore, the set the progressing parts forms an infinite subtree U_1 of U.

Let $\nu_0, \nu_1, \dots$ be an infinite path through U_1. Notice that $\mathtt{sat}(\nu_s) \neq \emptyset$. Indeed, if $\mathtt{sat}(\nu_s) = \emptyset$, then the part ν_s is empty in $\mathcal{C}_s$, where $c_s = (\vec{E}_s, \mathcal{C}_s)$, and therefore we cannot find some progressing part ν_{s+1} refining ν_s. Therefore, the set $\bigcap_s \mathtt{sat}(\nu_s)$ is non-empty. Let $X_1 \in \bigcap_s \mathtt{sat}(\nu_s)$. By definition of $\mathtt{sat}(\nu_s)$, X_1 is a transitive subtournament of R^{X_0}. By definition of $\mathtt{prog}$, for every s and every set $G \in \mathtt{sat}(\nu_s)$, there is some $n \in G$ such that $n > s$. Therefore, the set X_1 is infinite, so the property 1 is satisfied.

Repeat the procedure with the sequence of stage trees $T_1^{[\nu_1]} \geq T_2^{[\nu_2]} \geq \dots$ and so on. We obtain an infinite sequence of sets $X_0, X_1, \dots$ satisfying the property 1. Let f be the Δ^0_2 function which on input x, returns $max(U_x)+1$ where U_x is the finite set stated in the domination lemma (Lemma 4.9) for stage trees of depth x. Fix some Turing index e such that $\Phi_e^{X_0 \oplus \dots \oplus X_i}$ is total. By the domination lemma, for every $x \geq max(e, i)$, $\Phi_e^{X_0 \oplus \dots \oplus X_i}(x) \in U_x < f(x)$. Therefore the function f dominates every $X_0 \oplus \dots \oplus X_i$-computable function. This finishes the proof of Theorem 5.1.

6. Separating combined principles from AMT

The domination framework has two purposes. First, it emphasizes on the key elements of the construction and gets rid of the implementation technicalities by abstracting the main operations into operators. Second, it enables us to separate conjunctions of principles from AMT, using the ability to compose modules into a compound one. In this section, we will take advantage of the latter to prove that EM is not strong enough to prove AMT, even when allowing compactness arguments.

Theorem 6.1 There is an ω-model of $\mathsf{EM} \wedge \mathsf{COH} \wedge \mathsf{WKL}$ which is not a model of AMT.

In subsection 6.1, we will show how to compose multiple modules to obtain separations of conjunctions of principles from AMT. Then, in subsection 6.2, we will provide a module for WKL and will show how to choose properly the sequence of sets $X_0, X_1, \dots$ to obtain an ω-model of WKL.

6.1. Composing modules

When building the second-order part $\mathcal{I}$ of an ω-model of a countable collection of principles $\mathsf{P}_0, \mathsf{P}_1, \dots$, we usually interleave the instances of the various P's so that each instance receives attention after a finite number of iterations. This is exactly what we will do when composing module supports $\mathbb{S}_i = \langle \mathbb{P}_i, \mathbb{U}_i, \mathtt{parts}_i, \mathtt{init}_i, \mathtt{sat}_i \rangle$ for P_i for each $i \in \omega$, in order to obtain a compound module support $\mathbb{S} = \langle \mathbb{P}, \mathbb{U}, \mathtt{parts}, \mathtt{init}, \mathtt{sat} \rangle$ for $\bigwedge_{i \in \mathbb{N}} \mathsf{P}_i$. The domain of the partial order $\mathbb{P}$ is obtained by taking the disjoint union of the partial orders $\mathbb{P}_i$. Therefore $\mathbb{P} = \{\langle c, i \rangle : i \in \mathbb{N} \wedge c \in \mathbb{P}_i\}$. The order is defined accordingly: $\langle d, j \rangle \leq_{\mathbb{P}} \langle c, i \rangle$ if $i = j$ and $d \leq_{\mathbb{P}_i} c$. Similarly, $\mathbb{U} = \{\langle \nu, i \rangle : i < \mathbb{N} \wedge \nu \in \mathbb{U}_i\}$, $\mathtt{parts}(\langle c, i \rangle) = \{\langle \nu, i \rangle : \nu \in \mathtt{parts}_i(c)\}$ and $\mathtt{sat}(\langle \nu, i \rangle) = \mathtt{sat}_i(\nu)$.

The key element of the composition is the definition of $\mathtt{init}(n)$, which will return $\mathtt{init}_i(m)$ if n codes the pair (m, i). This way, infinitely many iterations are responsible for making $\mathcal{I}$ satisfy P_i for each $i \in \mathbb{N}$. The construction within the domination framework therefore follows the usual construction of a model satisfying two principles.

The property (i) in the definition of a module support for $\mathbb{S}$ inherits from the property (i) of $\mathbb{S}_i$ for each $i \in \mathbb{N}$. Indeed, if $\langle d, j \rangle \leq_{\mathbb{P}} \langle c, i \rangle$, then $j = i$ and $d \leq_{\mathbb{P}_i} c$. By the property (i) of $\mathbb{M}_i$, there is a function $f : \mathtt{parts}_i(d) \to \mathtt{parts}_i(c)$ such that $\mathtt{sat}_i(\nu) \subseteq \mathtt{sat}_i(f(\nu))$ for each $\nu \in \mathtt{parts}_i(d)$. Let $g : \mathtt{parts}(\langle d, i \rangle) \to \mathtt{parts}(\langle c, i \rangle)$ be defined by $g(\langle \nu, i \rangle) = \langle f(\nu), i \rangle$. $\mathtt{sat}(\langle \nu, i \rangle) = \mathtt{sat}_i(\nu) \subseteq \mathtt{sat}_i(f(\nu)) = \mathtt{sat}(g(\langle \nu, i \rangle))$.

Given a module $\mathbb{M}_i = \langle \mathbb{S}_i, \mathtt{box}_i, \mathtt{unbox}_i, \mathtt{prog}_i \rangle$ for P_i for each $i \in \mathbb{N}$, the definition of the compound module $\mathbb{M} = \langle \mathbb{S}, \mathtt{box}, \mathtt{unbox}, \mathtt{prog} \rangle$ for $\bigwedge_{i \in \mathbb{N}} \mathsf{P}_i$ does not contain any particular subtlety. Simply redirect $\mathtt{box}(\langle \nu, i \rangle, \varphi)$ to $\mathtt{box}_i(\nu, \varphi)$, $\mathtt{unbox}(\langle \nu, i \rangle, a)$ to $\mathtt{unbox}_i(\nu, a)$, and $\mathtt{prog}(\langle \nu, i \rangle)$ to $\mathtt{prog}_i(\nu)$. Again, the properties of a module support for $\mathbb{M}$ inherit the properties for $\mathbb{M}_i$.

6.2. A module for WKL

Weak König's lemma states for every infinite binary tree the existence of an infinite path through it. The usual effective construction of such a path follows the classical proof of König's lemma: we build the path by finite approximations and consider the infinite subtree below the finite path we constructed so far. The difficulty consists of finding which ones, among the finite extensions candidates, induce an infinite subtree.

First note that we do not share the same concerns as for the Erdős-Moser theorem about the choice of an instance, since WKL admits a universal instance which is the tree whose paths are completions of Peano arithmetics. Moreover, this universal instance is a primitive recursive tree functional.

It is natural to choose the infinite, computable binary tree functionals as our forcing conditions. A condition (tree) U *extends* T if $U^D \subseteq T^D$. A set G *satisfies* the condition T if G is an infinite path through T^D. Let us now see how we decide some Σ^0_1 query $\varphi(D, G)$. Consider the following question:

Q6: Is the set $T^D \cap \{\sigma \in 2^{<\omega} : \neg\varphi(D, \sigma)\}$ finite?

Let $\Gamma_\varphi^D = \{\sigma \in 2^{<\omega} : \neg\varphi(D,\sigma)\}$. Whenever $\varphi(D,\tau)$ holds and $\rho \succeq \tau$, $\varphi(D,\rho)$ holds, thus Γ_φ^D is a tree. At first sight, the question Q6 seems $\Sigma_2^{0,D}$. However, $T^D \cap \Gamma_\varphi^D$ is a tree, so the question can be formulated in a $\Sigma_1^{0,D}$ way as follows:

Q6': Is there some length n such that $T^D \cap \Gamma_\varphi^D$ has no string of length n?

If the answer is negative, the extension $T \cap \Gamma_\varphi$ is valid and forces $\varphi(D,G)$ not to hold. If the answer is positive, the condition T already forces $\varphi(D,G)$ to hold. Note that there is a hidden application of our moto "if you cannot choose, try every possibilities in parallel". Indeed, in many forcing arguments involving weak König's lemma, we $\emptyset'$-choose an extensible string $\sigma \in T$ such that $\varphi(D,\sigma)$ holds and $T^{D,[\sigma]}$ is infinite. However, we meet the same problem as in the Erdős-Moser case, that is, we are unable to decide which of the σ's will be extensible into an infinite subtree. To be more precise, for every $\sigma \in 2^n$, there may be some D such that the set $T^{D,[\sigma]}$ is finite. By taking T as our extension forcing $\varphi(D,G)$ to hold, we take in reality $\bigcup_{\sigma \in 2^n \cap T} T^{[\sigma]}$, that is, we take the union of the candidate extensions $T^{[\sigma]}$. We are now ready to define a module support $\mathbb{S}^{\mathsf{WKL}} = \langle \mathbb{P}, \mathbb{U}, \mathtt{parts}, \mathtt{init}, \mathtt{sat} \rangle$ for WKL.

The set $\mathbb{P}$ is the set of conditions as defined above. Each condition has only one part which can be identified as the condition itself, therefore $\mathbb{U} = \mathbb{P}$. Accordingly, $\mathtt{parts}(T) = \{T\}$. The function $\mathtt{init}(n)$ always returns the universal instance of WKL. Last, $\mathtt{sat}(T)$ is the collection of the infinite paths through T^D.

Lemma 6.2 $\mathbb{S}^{\mathsf{WKL}}$ is a module support.

Proof. We need to check the property (i) of a module support. Let $U \leq_{\mathbb{P}} T$ for some conditions T and $U \in \mathbb{P}$. Define $f : \mathtt{parts}(U) \to \mathtt{parts}(T)$ as the function $f(U) = T$. We claim that $\mathtt{sat}(U) \subseteq \mathtt{sat}(f(U))$ for each $U \in \mathtt{parts}(U)$. Since $\mathtt{parts}(U) = \{U\}$, we need to check that $\mathtt{sat}(U) \subseteq \mathtt{sat}(T)$, which is immediate since $U \subseteq T$. □

We now define the module $\mathbb{M}^{\mathsf{WKL}} = \langle \mathbb{S}^{\mathsf{WKL}}, \mathtt{box}, \mathtt{unbox}, \mathtt{prog} \rangle$ as follows. Given some tree T and some query $\varphi(D,G)$, $\mathtt{box}(T,G)$ is the formula $\psi(D) = (\exists n)[T^D \cap \Gamma_\varphi^D \cap 2^n = \emptyset]$. Recall that $\Gamma_\varphi^D = \{\sigma \in 2^{<\omega} : \neg\varphi(D,\sigma)\}$. If the answer to the question $\psi(D)$ is $\langle \mathtt{No} \rangle$, $\mathtt{unbox}(T, \langle \mathtt{No} \rangle)$ returns the tuple $\langle T \cap \Gamma_\varphi, f, b \rangle$ where $f : \mathtt{parts}(T \cap \Gamma_\varphi) \to \mathtt{parts}(T)$ is trivially defined by $f(T \cap \Gamma_\varphi) = T$ and b is the constant function returning $\langle \mathtt{No} \rangle$ everywhere. In the answer to the question $\psi(D)$ is $a = \langle \mathtt{Yes}, n, a' \rangle$, where n is the integer witnessing $T^D \cap \Gamma_\varphi^D \cap 2^n = \emptyset$ and a' witnesses the other existentials variables in $\varphi(D,G)$, $\mathtt{unbox}(T,a)$ returns the tuple $\langle T, id, b \rangle$ where id is the identify refinement function and b is the constant function returning $\langle \mathtt{Yes}, a' \rangle$ everywhere. No progress is needed for WKL. Therefore, $\mathtt{prog}(T)$ can be chosen to be any formula.

Lemma 6.3 $\mathbb{M}^{\mathsf{WKL}}$ is a module.

Proof. We need to ensure that, given the unique part T of the condition T and some answer a to a Σ_1^0 formula $\psi(D) = \mathtt{box}(T,\varphi)$ where $\varphi(D,G)$ is a Σ_1^0 formula, $\mathtt{unbox}(T,a)$ outputs a tuple $\langle U, f, b \rangle$ where U is an extension of T, $f : \mathtt{parts}(U) \to \mathtt{parts}(T)$ is defined by $f(U) = T$ and for every set $G \in \mathtt{sat}(U)$, $b(U)$ is an answer to $\varphi(D,G)$.

Suppose that $a = \langle \mathtt{No} \rangle$. By definition of $\mathtt{unbox}(T, \langle \mathtt{No} \rangle)$, $U = T \cap \Gamma_\varphi$ and b is the constant function $\langle \mathtt{No} \rangle$. By definition of $\mathtt{sat}(U)$, G is an infinite path through $T^D \cap \Gamma_\varphi^D$. Let σ be any initial segment of G. In particular, $\sigma \in \Gamma_\varphi^D$. Unfolding the definition of Γ_φ^Dn $\varphi(D,\sigma)$ does not hold. Therefore $\varphi(D,G)$ does not hold.

Suppose now that $a = \langle \mathtt{Yes}, n, a' \rangle$, where a' witnesses the existential variables of $\varphi(D,G)$. Again, by definition of $\mathtt{unbox}(T,a)$, $U = T$, f is the identify function and b is the constant function returning $\langle \mathtt{Yes}, a' \rangle$ everywhere. By definition of $\mathtt{sat}(U)$, G is an infinite path through T^D. Let σ be an initial segment of G of length n. Since $T^D \cap \Gamma_\varphi^D \cap 2^n = \emptyset$, $\varphi(D,\sigma)$ holds, so $\varphi(D,G)$ holds. □

Finally, we explain how to extract a solution to the universal instance of WKL below some set D, given the infinite decreasing sequence of stage trees constructed in subsection 4.3. Given the sequence $T_0 \geq T_1 \geq \dots$ whose roots are $T_0 \geq T_1 \geq \dots$, there is no much choice since each condition T_s has only one part, that is, the tree T_s itself. By compactness, $\bigcap_s T_s^D$ is infinite. Take any infinite path G through $\bigcap_s T_s^D$. This completes the proof of Theorem 6.1.

6.3. Beyond the atomic model theorem

We conclude this section by a discussion on the generality of the domination framework and its key properties.

Ramsey-type theorems satisfy one common core combinatorial property: given an instance I of a principle P, for every infinite set $X \subseteq \mathbb{N}$, there is a solution of $Y \subseteq X$ of I. This property makes Ramsey-type principles combinatorially weak. Indeed, Solovay [30] proved that the sets computable by every solution to a given instance I of P are precisely the hyperarithmetical ones. Moreover, Groszek and Slaman [10] proved that the hyperarithmetical sets are precisely the sets S admitting a modulus, namely, a function f such that every function dominating f computes S. These results put together can be interpreted as stating that the coding power of Ramsey-type principles comes from the sparsity of their solutions. If an instance can force its solutions $H = \{x_0 < x_1 < \dots\}$ to have arbitrarily large gaps, then the principal function p_H defined by $p_H(n) = x_n$ will be fast-growing, and contain some computational power.

The strength of many principles in reverse mathematics can be explained in terms of the ability to ensure gaps in the solutions. ACA has instances whose solutions are everywhere-sparse, in that the principal function of the solutions dominated the modulus function of $\emptyset'$. Some principles such as COH, AMT or FIP imply the existence of hyperimmune sets, which are sets sparse enough so that their principal function is not dominated by any computable function. These sets have infinitely many gaps, but their repartition cannot be controlled.

Another important aspect of the hole-based analysis is their definitional complexity. For example, AMT has the ability to ensures Δ^0_2 gaps, which gives it more computational power than COH or EM which can only have Δ^0_1 gaps. This is the main feature used by the domination framework to prove that $\mathsf{COH} \wedge \mathsf{EM}$ does not imply AMT. This framework was designed to exploit this weakness of the principles, and is therefore relatively specific to the atomic model theorem. However, some weakenings of AMT, such as the finite intersection property, share some similar properties, in that they can also be purely characterized in terms of hyperimmunity properties. The author leaves open the following question:

Question 6.4 Does COH imply FIP in RCA_0?

7. Evolution of the local zoo

In this last section, we give a short history of the zoo related to the Erdős-Moser theorem. In Figure 6, we present the various implications proven between the principles EM, STS^2, SADS and AMT. An arrow denotes an implication over RCA_0. A dotted arrow from a principle P to a principle Q denotes the existence of an ω-model of P which is not a model of Q.

Justification of the arrows:

(1) Hirschfeldt, Shore and Slaman [15] proved that AMT is a consequence of SADS over RCA_0.
(2) The author proved in [25] that $\mathsf{STS}(2)$ implies AMT over RCA_0 using a similar argument.
(3) Lerman, Solomon and Towsner [19] separated EM from SADS using an iterated forcing construction.
(4) The author noticed in [22] that the forcing of Lerman, Solomon and Towsner could be adapted to separate EM from $\mathsf{STS}(2)$ over RCA_0.
(5) Wang [31] used the notion of preservation of Δ^0_2 definitions to separate $\mathsf{COH} + \mathsf{EM} + \mathsf{WKL}$ from SADS and $\mathsf{STS}(2)$ over RCA_0.
(6) This is the main result of the current paper.

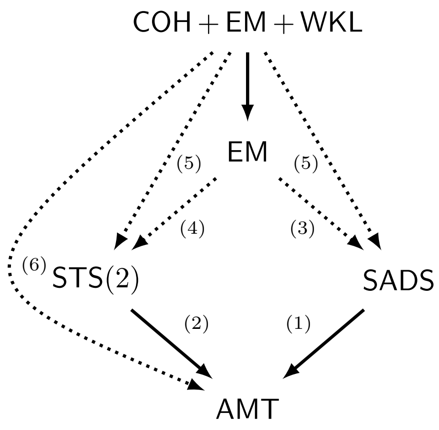

Figure 6. Evolution of the zoo

Acknowledgements. The author is thankful to his PhD advisor Laurent Bienvenu for useful comments and discussions, and to the anonymous referee for his numerous comments and improvement suggestions. The author is funded by the John Templeton Foundation ('Structure and Randomness in the Theory of Computation' project). The opinions expressed in this publication are those of the author(s) and do not necessarily reflect the views of the John Templeton Foundation.

References

[1] Andrey Bovykin and Andreas Weiermann. The strength of infinitary Ramseyan principles can be accessed by their densities. *Annals of Pure and Applied Logic*, page 4, 2005. To appear.

[2] Peter Cholak, Rod Downey, and Greg Igusa. Any FIP real computes a 1-generic. *arXiv preprint arXiv:1502.03785*, 2015.

[3] Peter A. Cholak, Carl G. Jockusch, and Theodore A. Slaman. On the strength of Ramsey's theorem for pairs. *Journal of Symbolic Logic*, 66(01):1–55, 2001.

[4] Chitat Chong, Theodore Slaman, and Yue Yang. The metamathematics of stable Ramsey's theorem for pairs. *Journal of the American Mathematical Society*, 27(3):863–892, 2014.

[5] Chris J. Conidis. Classifying model-theoretic properties. *Journal of Symbolic Logic*, 73(03):885–905, 2008.

[6] S. Barry Cooper. Minimal degrees and the jump operator. *Journal of Symbolic Logic*, 38(02):249–271, 1973.

[7] Rod Downey, David Diamondstone, Noam Greenberg, and Daniel Turetsky. The finite intersection principle and genericity. to appear in the Mathematical Proceedings of the Cambridge Philosophical Society. Available at `http://homepages.ecs.vuw.ac.nz/~downey/publications/FIP_paper.pdf`, 2012.

[8] Damir D. Dzhafarov. Strong reductions between combinatorial principles. In preparation.

[9] Damir D. Dzhafarov and Carl Mummert. On the strength of the finite intersection principle. *Israel J. Math.*, 196(1):345–361, 2013.

[10] Marcia J Groszek and Theodore A Slaman. Moduli of computation (talk). *Buenos Aires, Argentina*, 2007.

[11] Denis R. Hirschfeldt. *Slicing the truth*, volume 28 of *Lecture Notes Series. Institute for Mathematical Sciences. National University of Singapore*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2015. On the computable and reverse mathematics of combinatorial principles, Edited and with a foreword by Chitat Chong, Qi Feng, Theodore A. Slaman, W. Hugh Woodin and Yue Yang.

[12] Denis R. Hirschfeldt and Carl G. Jockusch. On notions of computability-theoretic reduction between Π^1_2 principles. *J. Math. Log.*, 16(1):1650002, 59, 2016.

[13] Denis R. Hirschfeldt, Carl G. Jockusch, Bjørn Kjos-Hanssen, Steffen Lempp, and Theodore A. Slaman. The strength of some combinatorial principles related to Ramsey's theorem for pairs. *Computational Prospects of Infinity, Part II: Presented Talks, World Scientific Press, Singapore*, pages 143–161, 2008.

[14] Denis R. Hirschfeldt and Richard A. Shore. Combinatorial principles weaker than Ramsey's theorem for pairs. *Journal of Symbolic Logic*, 72(1):171–206, 2007.

[15] Denis R. Hirschfeldt, Richard A. Shore, and Theodore A. Slaman. The atomic model theorem and type omitting. *Transactions of the American Mathematical Society*, 361(11):5805–5837, 2009.

[16] Carl G. Jockusch and Frank Stephan. A cohesive set which is not high. *Mathematical Logic Quarterly*, 39(1):515–530, 1993.

[17] Laurie Kirby and Jeff Paris. Accessible independence results for peano arithmetic. In *Bulletin of the London Mathematical Society*, volume 14, pages 285–293. Oxford University Press, 1982.
[18] Alexander P. Kreuzer. Primitive recursion and the chain antichain principle. *Notre Dame Journal of Formal Logic*, 53(2):245–265, 2012.
[19] Manuel Lerman, Reed Solomon, and Henry Towsner. Separating principles below Ramsey's theorem for pairs. *Journal of Mathematical Logic*, 13(02):1350007, 2013.
[20] Joseph Roy Mileti. *Partition theorems and computability theory*. ProQuest LLC, Ann Arbor, MI, 2004. Thesis (Ph.D.)–University of Illinois at Urbana-Champaign.
[21] Antonio Montalbán. Open questions in reverse mathematics. *Bulletin of Symbolic Logic*, 17(03):431–454, 2011.
[22] Ludovic Patey. A note on "Separating principles below Ramsey's theorem for pairs". Unpublished. Available at `http://ludovicpatey.com/media/research/note-em-sts.pdf`, 2013.
[23] Ludovic Patey. Degrees bounding principles and universal instances in reverse mathematics. *Annals of Pure and Applied Logic*, 166(11):1165–1185, 2015.
[24] Ludovic Patey. Iterative forcing and hyperimmunity in reverse mathematics. In Arnold Beckmann, Victor Mitrana, and Mariya Soskova, editors, *CiE. Evolving Computability*, volume 9136 of *Lecture Notes in Computer Science*, pages 291–301. Springer International Publishing, 2015.
[25] Ludovic Patey. Somewhere over the rainbow Ramsey theorem for pairs. Submitted. Available at `http://arxiv.org/abs/1501.07424`, 2015.
[26] Ludovic Patey. The weakness of being cohesive, thin or free in reverse mathematics. *Israel J. Math.*, 216(2):905–955, 2016.
[27] David Seetapun and Theodore A. Slaman. On the strength of Ramsey's theorem. *Notre Dame Journal of Formal Logic*, 36(4):570–582, 1995.
[28] Joseph R. Shoenfield. On degrees of unsolvability. *Annals of Mathematics*, 69(03):644–653, May 1959.
[29] Robert I. Soare. *Turing computability*. Theory and Applications of Computability. Springer-Verlag, Berlin, 2016. Theory and applications.
[30] Robert M. Solovay. Hyperarithmetically encodable sets. *Trans. Amer. Math. Soc.*, 239:99–122, 1978.
[31] Wei Wang. The definability strength of combinatorial principles, 2014. To appear. Available at http://arxiv.org/abs/1408.1465.
[32] Liang Yu. Lowness for genericity. *Archive for Mathematical Logic*, 45(2):233–238, 2006.