Strongly minimal theories with recursive models

Uri Andrews Julia F. Knight

Abstract

We give effectiveness conditions on a strongly minimal theory T guaranteeing that all countable models have computable copies. In particular, we show that if T is strongly minimal and for all $n \geq 1$, $T \cap \exists_{n+2}$ is Δ^0_n, uniformly in n, then every countable model has a computable copy. A longstanding question of computable model theory asked whether for a strongly minimal theory with one computable model, every countable model has an arithmetical copy. Relativizing our main result, we get the fact that if there is one computable model, then every countable model has a Δ^0_4 copy.

1 Introduction

In computable model theory, we try to understand the algorithmic complexity of the various models of an elementary first order theory. For a theory with nice model-theoretic properties, it should be easier to understand the complexity of the models. We mention first some results for $\aleph_0$-categorical theories. Lerman and Schmerl [20] showed that for an $\aleph_0$-categorical theory T, if T is arithmetical and $T \cap \exists_{n+1}$ is Σ^0_n for each n, then T has a computable model. Knight [18] dropped the assumption that T is arithmetical, assuming that $T \cap \exists_{n+1}$ is Σ^0_n uniformly in n. At the time these results were proved, there was no known example of a non-arithmetical $\aleph_0$-categorical theory with a computable model. Khoussainov and Montalbán [14] gave an example, using an infinite language. Andrews [1] showed that in all tt degrees $\leq \mathbf{0}^{(\omega)}$, there are $\aleph_0$-categorical theories with computable models. Moreover, the theories are in a finite language.

For $\aleph_1$-categorical theories, Goncharov and Khoussainov [9] and Fokina [6] gave examples for which the theory has degree $\mathbf{0}^{(n)}$ and there are computable models. These

U. Andrews: University of Wisconsin, 480 Lincoln Dr., Madison, WI 53706;
e-mail: andrews@math.wisc.edu

J. F. Knight: University of Notre Dame, 255 Hurley, Notre Dame, IN 46556 ;
e-mail: Julia.F.Knight.1@nd.edu

Mathematics Subject Classification (2010): Primary 03C57; Secondary 03D45

examples used "Marker extensions", a method that does not produce strongly minimal theories. Andrews [1] showed that there are non-arithmetical strongly minimal theories with computable models. For strongly minimal theories, as for $\aleph_0$-categorical theories, in each tt-degree $\leq \mathbf{0}^{(\omega)}$, there is a theory T, in a finite language, such that all models of T have computable copies.

An important collection of results and questions in computable model theory involves using a bound on complexity of one model of a theory to give a uniform bound that serves for some copy of each model. A major open problem since the 90's has been to find such a uniform bound in the case of a strongly minimal theory with a computable model. Goncharov, Harizanov, Laskowski, Lempp, and McCoy [8] solved this problem in the case where the strongly minimal theory is *disintegrated* (or *geometrically trivial*). They showed that if $\mathcal{M}$ is a model of a disintegrated strongly minimal theory T, then the complete (or elementary) diagram, $D^c(\mathcal{M})$, is model complete. It follows that if $\mathcal{M}$ is X-computable, then T is computable in X''. In particular, if T is a disintegrated strongly minimal theory with a computable model, then T must be Δ^0_3. It then follows from a theorem of Harrington [10] and Khisamiev [12] that every model of T has a copy whose complete diagram is Δ^0_3. One might hope to drop the assumption of disintegration in the result of [8]. By a result of Andrews [1], there are non-arithmetical strongly minimal theories whose countable models are computable. This led some people to suspect that there would be no arithmetical set that computes copies of all models.

Here is our main result.

Main Theorem. If T is a strongly minimal theory such that $T \cap \exists_{n+2}$ is uniformly Δ^0_n, then every countable model of T has a computable copy.

The main result, relativized to $\emptyset^{(3)}$, gives the following corollary.

Main Corollary. If T is a strongly minimal theory with a recursive model, then every countable model of T has a Δ^0_4 copy.

Proof. If T has a recursive model, then $T \cap \exists_n$ is uniformly Σ^0_n. Thus, relative to $\emptyset^{(3)}$, $T \cap \exists_{n+2}$ is uniformly Δ^0_n. So, by the Main Theorem, relativized to $\emptyset^{(3)}$, all the other countable models have copies computable in $\emptyset^{(3)}$. □

Khoussainov, Laskowski, Lempp, and Solomon [13] showed that there is a disintegrated strongly minimal theory T such that the prime model has a computable copy, and the other models (with universe a subset of ω), not isomorphic to the prime model, all compute $\emptyset''$. This is the largest known gap. It remains open whether our Main Corollary is sharp.

Question 1. Is it true that if T is strongly minimal and has a recursive model, then all models of T have Δ^0_3 copies?

1.1 Model theoretic preliminaries

Definition 1.1. A complete elementary first order theory T is *strongly minimal* if for every model $\mathcal{M}$, and every formula $\varphi(\bar{a}, x)$ with parameters $\bar{a}$ in $\mathcal{M}$, $\varphi^{\mathcal{M}}(\bar{a}, x)$ is finite or co-finite.

Examples. The following are strongly minimal theories.

1. the theory of $\mathbb{Z}$ with the successor function
2. the theory of infinite $\mathbb{Q}$-vector spaces
3. the theory of the field $\mathbb{C}$ of complex numbers

These three canonical examples represent the three classes of the "Zilber trichotomy", which Zilber at one time conjectured to exhaust the strongly minimal theories. Hrushovski [11] showed that the Zilber Trichotomy Conjecture is not true by constructing exotic strongly minimal theories. These theories are combinatorial in nature and have no natural algebraic interpretation. Andrews [1], [2], [3] used variants of Hrushovski's construction to produce strongly minimal theories with interesting recursion-theoretic properties.

Definition 1.2 (Algebraic closure, independence)**.** Let T be a strongly minimal theory, let $\mathcal{M}$ be a model of T, and let X be a subset of $\mathcal{M}$.

- The *algebraic closure* of X in $\mathcal{M}$, denoted by $\mathrm{acl}_{\mathcal{M}}(X)$, is the union of the finite sets $\varphi^{\mathcal{M}}(\bar{c}, x)$ definable in $\mathcal{M}$ with parameters $\bar{c}$ in X.
- The set X is *algebraically independent* if for all $a \in X$, $a \notin \mathrm{acl}_{\mathcal{M}}(X \setminus \{a\})$.

When the model in question is clear, we write acl for $\mathrm{acl}_{\mathcal{M}}$. Algebraic closure gives a well-defined notion of dimension. The *dimension* of a set is the size of a maximal algebraically independent subset. For a strongly minimal theory T, each model is determined, up to isomorphism, by its dimension.

We say what dimension and algebraic closure mean in the three examples above. For the theory of $(\mathbb{Z}, S)$, each model consists of some number of $\mathbb{Z}$-chains. The algebraic closure of a set X is the union of the $\mathbb{Z}$-chains containing elements of X, and the dimension is the number of these $\mathbb{Z}$-chains. For the theory of non-trivial $\mathbb{Q}$-vector spaces, the algebraic closure of a set is the linear span, and dimension is vector space dimension. For the theory of $\mathbb{C}$ (the theory of algebraically closed fields of characteristic 0), algebraic closure is usual algebraic closure, and dimension is transcendence degree.

Definition 1.3 (Disintegration)**.** A strongly minimal theory is *disintegrated* if for all models $\mathcal{M}$ and $X \subseteq \mathcal{M}$, $\mathrm{acl}_{\mathcal{M}}(X) = \bigcup_{s \in X} \mathrm{acl}_{\mathcal{M}}(\{s\})$.

Of the three examples given above, only the first is disintegrated.

The assumptions in our main theorem involve fragments of the theory T of different quantifier complexities. In the proof, we classify formulas with free variables also by their quantifier complexity.

Definition 1.4. A formula is $\exists_n$ if it has the form $(\exists \bar{x}_1)(\forall \bar{x}_2)\cdots(Q\bar{x}_n)\varphi(\bar{y},\bar{x})$, where φ is quantifier-free and Q is either $\forall$ or $\exists$, depending on the parity of n. Note that a string of like quantifiers (all $\exists$, or all $\forall$) is counted as a single quantifier.

Definition 1.5 (B_n-formula, B_n-type, B_n-algebraicity)**.**

- A B_n-*formula* is a Boolean combination of $\exists_n$-formulas.
- A B_n-*type* is the set of B_n-formulas in a complete type.
- If b satisfies a B_n-formula $\varphi(\bar{a},x)$ such that $\varphi^{\mathcal{M}}(\bar{a},x)$ is finite, we say that b is in the B_n-*algebraic closure* of $\bar{a}$. In this case, we write $b \in \mathrm{acl}_{B_n}(\bar{a})$.

Notation: For $n \geq 1$, we write T_n for $T \cap \exists_n$.

Note that if $T \cap \exists_n$ is computable in X, so is $T \cap B_n$.

Definition 1.6 (n-saturation, bounded saturation)**.**

- $\mathcal{M}$ is n-*saturated* if for all $\bar{a} \in \mathcal{M}$, every B_n-type $p(\bar{a},x)$ consistent with the type of $\bar{a}$ is realized in $\mathcal{M}$.
- $\mathcal{M}$ is *boundedly saturated* if it is n-saturated for all n.

Every saturated structure is boundedly saturated. The familiar examples of strongly minimal theories have elimination of quantifiers down to B_n-formulas for some n, so the finite dimensional models are not boundedly saturated. However, Andrews [1] gave examples of strongly minimal theories whose finite dimensional models are boundedly saturated.

In the context of a strongly minimal theory, knowing that $\mathcal{M}$ is boundedly saturated is useful in building a copy. When we are building a copy of a model $\mathcal{M}$ that is boundedly saturated, we use the fact that it is always safe to add realizations of consistent B_n-types (as in Lemma 1.7 below). Knowing that $\mathcal{M}$ is *not* boundedly saturated is also useful for building a copy. If $\mathcal{M}$ is not n-saturated, there is some $\bar{c}$ in $\mathcal{M}$ such that every element of $\mathcal{M}$ is algebraic over $\bar{c}$ via a B_n-formula (see Lemma 5.1). It is always safe to add realizations of consistent types that contain such formulas.

The following lemma gives a hint as to how the condition of bounded saturation will be used.

Lemma 1.7. *Let $r(\bar{x})$ be a B_n-type.*

1. *Let $s(\bar{x}, y)$ be a B_n-type, extending $r(\bar{x})$, such that $s(\bar{x}, y)$ is generated by the formulas of $r(\bar{x})$ and $\exists_n$-formulas. Then for every extension of $r(\bar{x})$ to a complete type $r'(\bar{x})$, $r'(\bar{x})$ is consistent with $s(\bar{x}, y)$.*

2. *For any set Ψ of $\exists_n$-formulas in the variables $\bar{x}, y$, if $r(\bar{x}) \cup \Psi$ is consistent, then there is a B_n-type $s(\bar{x}, y)$, extending $r(\bar{x}) \cup \Psi$, such that $s(\bar{x}, y)$ is generated by the formulas of $r(\bar{x})$ and the $\exists_n$-formulas in $s(\bar{x}, y)$.*

3. *Suppose $\mathcal{M}$ is an n-saturated model of T. Let $p(\bar{x})$ be a B_n-type, and let $q(\bar{x}, \bar{y})$ be a B_{n-1}-type consistent with $p(\bar{x})$. Then every realization of $p(\bar{x})$ in $\mathcal{M}$ extends to a realization of $q(\bar{x}, \bar{y})$ in $\mathcal{M}$; i.e., the n-saturation condition gives a saturation condition for B_{n-1}-types of tuples.*

Proof. For Part 1, let $r'(\bar{x})$ be any type extending $r(\bar{x})$, and suppose that $s(\bar{x}, y)$ is generated by $r(\bar{x})$ and $\exists_n$-formulas. If $r'(\bar{x})$ and $s(\bar{x}, y)$ are inconsistent, then there is some $\exists_n$-formula $\psi(\bar{x}, y) \in s(\bar{x}, y)$ such that $r'(\bar{x}) \vdash \neg\psi(\bar{x}, y)$. Then $r'(\bar{x}) \vdash (\forall y)\neg\psi(\bar{x}, y)$. Since this is a $\forall_n$-formula, $r(\bar{x}) \vdash (\forall y)\neg\psi(\bar{x}, y)$, so $s(\bar{x}, y)$ is inconsistent. This is a contradiction.

For Part 2, let $r(\bar{x})$ be given, and fix an enumeration $(\varphi_j)_{j\in\omega}$ of all $\exists_n$-formulas in the variables $\bar{x}, y$. We can generate a B_n-type $s(\bar{x}, y)$ as follows. At stage 0, we set $\Phi_0 := \Psi$. At stage i, we have decided to put some subset Φ_{i-1} of $\Psi \cup \{\varphi_j \mid j < i\}$ into $s(\bar{x}, y)$. If $\Phi_{i-1} \cup \{\varphi_i(\bar{x}, y)\}$ is consistent with $r(\bar{x})$, then we let $\Phi_i = \Phi_{i-1} \cup \{\varphi_i\}$. If not, then it would be inconsistent to add φ_i, and we let $\Phi_i = \Phi_{i-1}$. This results in a set of $\exists_n$-formulas $\Phi = \bigcup_i \Phi_i$, which, along with $r(\bar{x})$, generates a complete B_n-type $s(\bar{x}, y)$.

For Part 3, we proceed by induction on the size of $\bar{y}$. If $\bar{y}$ has size 0, the claim is trivial. We are given a B_n-type $p(\bar{x})$, a B_{n-1}-type $q(\bar{x}, y, \bar{z})$, and a realization $\bar{a}$ of $p(\bar{x})$. Let Ψ be the set $\{(\exists\bar{z})\varphi(\bar{x}, y, \bar{z}) \mid \varphi \in q\}$. By Part 2 of the lemma, this can be extended to a type $s(\bar{x}, y)$ generated over $p(\bar{x})$ by its $\exists_n$-formulas, and by Part 1, $s(\bar{x}, y)$ is consistent with $p(\bar{x})$. By n-saturation, $s(\bar{x}, y)$ is realized by $\bar{a}$ and some b in $\mathcal{M}$. Now, the type $q(\bar{x}, y, \bar{z})$ is still consistent with the type of $\bar{a}, b$. By the inductive hypothesis, there is a realization of the type $q(\bar{a}, b, \bar{z})$, as required. □

The reason that this is pertinent is that during a construction, we will build B_{n-1}-types and will need to know that whatever we build, even if based on incorrect guesses at recursion-theoretic information, is realized in the model, provided that it is consistent with the theory.

In order to understand and enumerate the types in the theory, we use Morley rank, which we will often refer to simply as *rank*.

Definition 1.8 (Morley rank and degree)**.**

1. The *Morley rank* of a formula $\varphi(\bar{x})$ is the maximum dimension of a tuple (in any model) satisfying the formula. We write $\mathrm{MR}(\varphi(\bar{x}))$ for the Morley rank of $\varphi(\bar{x})$.

2. The *Morley rank of a type* is the minimum of the ranks of the formulas in the type.

3. The *Morley degree of a formula* $\varphi(\bar{x})$ is the maximal number of formulas $\psi_i(\bar{x})$ that are pairwise inconsistent, with $\mathrm{MR}(\varphi(\bar{x}) \wedge \psi_i(\bar{x})) = \mathrm{MR}(\varphi(\bar{x}))$. In every strongly minimal theory, the Morley degree of any formula is well-defined and finite.

4. A formula φ is said to have *minimal Morley rank/degree* inside a set S of formulas if φ has minimal Morley rank among all formulas in S, and among the formulas in S with this Morley rank, φ has minimal Morley degree.

Throughout what follows, we assume that T is a theory in a relational language. Given a theory T in a language L with function symbols, there is a natural theory T' in a relational language L' such that T' is inter-definable with T—simply consider functions as described by relation symbols instead of function symbols. Moreover, there is a uniform effective procedure for converting models of T' into models of T and vice versa. Since every term in L is uniformly both $\exists_1$- and $\forall_1$-definable in T', for each $n \geq 1$, every B_n-formula in T is expressed by a B_n-formula in T', and vice versa. Thus, for a theory T, if T_{n+2} is Δ^0_n uniformly in n, then the same is true for T'. We can then apply the Main Theorem to T' to see that every model of T' has a computable copy. This is enough to show that every model of T has a computable copy. Thus, working with theories in relational languages carries no loss of generality.

1.2 Recursion-theoretic preliminaries

The proofs of Theorems 6.1 and 7.1 each involve a construction by "workers". We build a sequence of structures $\mathcal{A}_n$ such that each $\mathcal{A}_n$ is Δ^0_n—we imagine Worker n as building the structure $\mathcal{A}_n$, using a Δ^0_n oracle. Each $\mathcal{A}_n$ is built to have certain relationships to the other structures $\mathcal{A}_i$ for $i \neq n$. Certainly, since $\mathcal{A}_{n+1}$ is not Δ^0_n, but rather only Δ^0_{n+1}, the construction of $\mathcal{A}_n$ cannot access direct information about $\mathcal{A}_{n+1}$. Rather, we will, in a Δ^0_n way, employ approximations at facts about $\mathcal{A}_{n+1}$ in the midst of the construction of $\mathcal{A}_n$. Below, we indicate how the Recursion Theorem allows us to engage in this construction to simultaneously build all the structures $\mathcal{A}_n$ for $n \in \omega$.

Our construction proceeds as follows. From a computable sequence of Δ^0_i-indices, for $i \neq n$, we produce a Δ^0_n-index for a structure $\mathcal{A}_n$ (in fact, the construction will use only the indices i for $i < n$ or $i = n + 1$). Furthermore, our construction has the property that on any input whatsoever (for example, the indices we are given could be for partial

functions), we produce an index for a total Δ^0_n function that describes $\mathcal{A}_n$. In Section 3, we discuss exactly how $\mathcal{A}_n$ is to be described, but the details of this have no bearing on our present general discussion. We may consider the sequence of constructions of $\mathcal{A}_n$ for each $n \in \omega$ as follows. Given any computation $\Phi_i^{0^{(\omega)}}$, we produce a new computation $\Phi_{\sigma(i)}^{0^{(\omega)}}$ so that $\Phi_{\sigma(i)}^{0^{(\omega)}}(\langle n, -\rangle)$ is the function giving the description of $\mathcal{A}_n$. Further, we ensure that $\Phi_{\sigma(i)}^{0^{(\omega)}}$ is total and that only the Δ^0_n fragment of the oracle is ever used in the computation of $\Phi_{\sigma(i)}^{0^{(\omega)}}(\langle n, -\rangle)$. We also ensure that if Φ_i is a computation with the same property, namely that only the Δ^0_n fragment of the oracle is ever used in the computation of $\Phi_i^{0^{(\omega)}}(\langle n, -\rangle)$, then each computation $\Phi_{\sigma(i)}^{0^{(\omega)}}(\langle n, -\rangle)$ gives the required Δ^0_n description of $\mathcal{A}_n$. Now, we take a fixed-point i so that $\Phi_{\sigma(i)}^{0^{(\omega)}} = \Phi_i^{0^{(\omega)}}$, as guaranteed by the Recursion Theorem. Doing this ensures that we build a sequence of Δ^0_n descriptions for structures $\mathcal{A}_n$ for $n \in \omega$ with the property that each $\mathcal{A}_n$ relates to the other structures $\mathcal{A}_i$ for $i \neq n$ according to the given construction.

The first argument of this kind was a result of Harrington, saying that there is a non-standard model of first order Peano arithmetic that is Δ^0_2, but whose theory is not arithmetical. For an account of this result, see [15] or [5]. Ash gave a "meta-theorem", for transfinitely nested limit constructions, and this is what is used in [5] to prove Harrington's Theorem. Ash developed his meta-theorem independently of Harrington. Ash's original proof of the meta-theorem was an induction inspired by Martin's proof of Borel Determinacy, not using workers. The proof of Ash's meta-theorem in [5] uses workers. Ash's meta-theorem is relatively simple to use when the conditions are satisfied, but, unfortunately, they are not satisfied in our setting. There are other general descriptions of worker constructions in [16] (for finite levels) and [17] (for transfinite levels). Lerman [19] described some very general machinery. More recently, Montalbán [21] gave his own account of worker constructions, based on the notion of "ξ-true" stages. It would be possible to fit our constructions into the framework of Montalbán.

Here we will not use any meta-theorem or general framework. Instead, we will simply describe how each worker acts. We did not want the details of some necessarily complicated formal machinery to obscure the content of the proof. Like everyone else who has described worker constructions, from Harrington to Montalbán, we use the Recursion Theorem. The Recursion Theorem merely puts together the pieces of the construction. We must still say what each worker does at each step, and we must argue that all requirements are satisfied in the end.

1.3 Outline

Our Main Theorem says that for a strongly minimal theory T such that each $T \cap \exists_{n+2}$ is uniformly Δ^0_n, every model $\mathcal{M}$ has a computable copy. In the proof, we consider four

cases, depending on whether T is arithmetical, and on the saturation properties of the model $\mathcal{M}$.

1. T is Δ^0_N and $\mathcal{M}$ is N-saturated,
2. $\mathcal{M}$ is not N-saturated for some N—T may be arithmetical or not,
3. T is not arithmetical and $\mathcal{M}$ is saturated,
4. T is not arithmetical and $\mathcal{M}$ is boundedly saturated but of finite dimension,

In Section 2, we say how difficult it is to compute Morley ranks. In Section 3, we use computations of Morley ranks to arrive at an indexing of all B_n-types, for each n. In Sections 4–7, we give the proof for the four cases. Case 1 is simpler than the others, but it illustrates many of the core ideas. We begin with this case in Section 4. Case 3, considered in Section 6, and Case 4, considered in Section 7, involve workers constructions.

2 Computing Morley ranks

In this section, T is a fixed strongly minimal theory.

Lemma 2.1. *For each formula $\varphi(\bar{u}, x)$, there is a number k such that for all models $\mathcal{M}$ and all $\bar{a}$ in $\mathcal{M}$, if $\varphi^{\mathcal{M}}(\bar{a}, x)$ has at least k elements, then it is infinite. Moreover, if φ is a B_n-formula, then we can find k using T_{n+1}.*

Proof. If there is no such k, then by Compactness, we would have a model $\mathcal{M}$ with a tuple $\bar{a}$ such that $\varphi^{\mathcal{M}}(\bar{a}, x)$ and $\neg\varphi^{\mathcal{M}}(\bar{a}, x)$ are both infinite. Since T is strongly minimal, this is impossible. To find k, we search for the first k such that the sentence saying

$$(\forall \bar{u})\neg[(\exists^{\geq k}x)\varphi(\bar{u}, x) \ \wedge (\exists^{\geq k}x)\neg\varphi(\bar{u}, x)]$$

is in T. If φ is a B_n-formula, the sentence is $\forall_{n+1}$, so T_{n+1}-suffices. □

Lemma 2.2. *For a formula $\varphi(\bar{u}, x_1, \ldots, x_m)$, there exist $k_1, \ldots, k_m$ such that in any model $\mathcal{M}$, for any tuple $\bar{a}$,*

$$\mathcal{M} \models (\exists^{\geq k_1}x_1)\cdots(\exists^{\geq k_m}x_m)\varphi(\bar{a}, \bar{x}) \ \textit{iff} \ \mathcal{M} \models (\exists^{\infty}x_1)\cdots(\exists^{\infty}x_m)\varphi(\bar{a}, \bar{x})$$

Moreover, if φ is an $\exists_n$-formula, then we can find the numbers $k_1, \ldots, k_m$ using T_{n+1}.

Proof. Applying Lemma 2.1 to an $\exists_n$-formula $\varphi(\bar{u}, x_1, \ldots, x_m)$, letting x_m play the role of x, we can find k_m, using T_{n+1}. The formula $(\exists^{\geq k_m}x_m)\varphi(\bar{u}, x_1, \ldots, x_m)$ is $\exists_n$. Applying Lemma 2.1, we can find k_{m-1}, using T_{n+1}. The formula $(\exists^{\geq k_{m-1}}x_{m-1})(\exists^{\geq k_m}x_m)\varphi(\bar{u}, \bar{x})$ is $\exists_n$. Applying Lemma 2.1 again, we can find k_{m-2}. We continue in this way until we have k_1. □

We write $|\bar{x}|$ for the length of a tuple $\bar{x}$.

Lemma 2.3. *For any formula $\varphi(\bar{x})$ and any $m \leq |\bar{x}|$, there is a sentence saying that $\varphi(\bar{x})$ has rank at least m. Moreover, if φ is an $\exists_n$-formula, then the sentence saying that the rank is at least m is $\exists_n$, and we can find this sentence using T_{n+1}. In particular, the function that assigns to an $\exists_n$-formula its Morley rank is uniformly computable from T_{n+1}, and thus the function that assigns to a B_n-formula its Morley rank is uniformly computable from T_{n+2}.*

Proof. For each partition of the variables $\bar{x}$ into a tuple $\bar{z}$ of length m and remaining variables $\bar{y}$, we have a sentence saying that $(\exists^\infty z_1)\cdots(\exists^\infty z_m)(\exists \bar{y})\varphi(\bar{x})$. To say that there exist infinitely many z_i, we say that there are at least k_i, for appropriate finite numbers k_i. If φ is an $\exists_n$-formula, then $(\exists \bar{y})\varphi$ is also $\exists_n$, and we use T_{n+1} to find the numbers, as in Lemma 2.2. In this way, we arrive at a sentence saying that the variables $\bar{z}$ witness that the rank of φ is at least m. To say that φ has rank at least m, we take the disjunction of these sentences, over the possible choices of $\bar{z}$. This sentence is $\exists_n$. □

We will in several places need the following refinement of Lemma 2.3.

Lemma 2.4. *The set of pairs $(\varphi(\bar{x}), k)$ such that φ is an $\exists_n$-formula of Morley rank at least k is uniformly $\Pi^0_1(T_n)$.*

Proof. The Morley rank of $\varphi(x_1, \ldots x_l)$ is at least k if, for some permutation σ of $\{1, \ldots, l\}$, for all m, the sentence

$$(\exists^{\geq m} x_{\sigma(1)})(\exists^{\geq m} x_{\sigma(2)})\cdots(\exists^{\geq m} x_{\sigma(k)})(\exists x_{\sigma(k+1)})\cdots(\exists x_{\sigma(l)})\varphi(\bar{x})$$

is in T_n. This condition is thus uniformly $\Pi^0_1(T_n)$. □

2.1 The generic type of a k-tuple

Definition 2.5. If Γ is a set of formulas, then an n-tuple $\bar{a} \in \mathcal{M}^n$ has a *generic Γ-type* if for every $\varphi(x_1, \ldots, x_n) \in \Gamma$, if $\mathcal{M} \models \varphi(\bar{a})$, then $\mathrm{MR}(\varphi(\bar{x})) = n$.

In particular, we use this definition where Γ is the set of $\exists_n$, $\forall_n$ or B_n-formulas and refer to a generic $\exists_n$-type, generic $\forall_n$-type, or generic B_n-type. The generic B_n-types play a special role in our analysis. Using Lemma 2.4, we can give a simple way of computing these types.

Lemma 2.6. *The set of $\exists_n$-formulas satisfied by a generic tuple $\bar{a}$ is uniformly $\Pi^0_1(T_n)$.*

Proof. The formula $\varphi(x_1, \ldots x_k)$ is satisfied by the generic tuple if and only if its Morley rank is $\geq k$. By Lemma 2.4, this is a $\Pi^0_1(T_n)$ condition. □

In several key places, knowing that this condition is $\Pi^0_1(T_n)$ will allow us to combine universal quantifiers.

3 Enumerating types

In this section, T is a strongly minimal theory such that $T \cap \exists_{n+2}$ is uniformly Δ^0_n. For each n, we define an enumeration, or indexing, P^n of the B_n-types, and we verify several properties of this enumeration

Definition 3.1. We say that a pair $(\theta(\bar{x}), k)$ is a P^n-*index* for a B_n-type $p(\bar{x})$ if the following hold:

- θ is a B_n-formula of Morley rank k,
- the type $p(\bar{x})$ is the only B_n-type of rank k containing θ.

Definition 3.2. For any tuple $\bar{z} \subseteq \bar{x}$, we partition $\bar{x} = \bar{z}\bar{y}$, and we let $\mathrm{mult}(\theta, \bar{z}) = l$ if l is greatest so that $T \models (\exists^\infty z_1)(\exists^\infty z_2) \dots (\exists^\infty z_k)(\exists^l \bar{y})\ \theta(\bar{x})$.

Note that if $\mathrm{mult}(\theta, \bar{z}) = \infty$, then $\mathrm{MR}(\theta) > |\bar{z}|$. Also, if $\mathrm{mult}(\theta, \bar{z}) > 0$, then $\mathrm{MR}(\theta) \geq |\bar{z}|$.

Definition 3.3. We define $\mathrm{Mult}(\theta, k)$ to be the function that sends each tuple $\bar{z}$ of length k to $\mathrm{mult}(\theta, \bar{z})$. We write $\mathrm{Mult}(\theta, k) \leq \mathrm{Mult}(\psi, k)$ provided that for each $\bar{z}$ of length k, $\mathrm{mult}(\theta, \bar{z}) \leq \mathrm{mult}(\psi, \bar{z})$. We write $\mathrm{Mult}(\theta, k) < \mathrm{Mult}(\psi, k)$ if $\mathrm{Mult}(\theta, k) \leq \mathrm{Mult}(\psi, k)$ and $\mathrm{Mult}(\theta, k) \neq \mathrm{Mult}(\psi, k)$.

Lemma 3.4. *The pair (θ, k) is an index for a B_n-type if and only if $\theta \in B_n$, $\mathrm{MR}(\theta) = k$, and for every B_n formula ψ, either $\mathrm{Mult}(\theta \wedge \psi, k) = \mathrm{Mult}(\theta, k)$ or $\mathrm{Mult}(\theta \wedge \neg\psi, k) = \mathrm{Mult}(\theta, k)$.*

Proof. First, suppose that (θ, k) is an index for a B_n-type. Then there is only one B_n type of rank k that contains θ. Thus, either $MR(\theta \wedge \psi) < k$ or $MR(\theta \wedge \neg\psi) < k$. It follows that either $\mathrm{Mult}(\theta \wedge \psi, k) = 0$ or $\mathrm{Mult}(\theta \wedge \neg\psi, k) = 0$. We have

$$\mathrm{Mult}(\theta, k) = \mathrm{Mult}(\theta \wedge \psi, k) + \mathrm{Mult}(\theta \wedge \neg\psi, k)\ ,$$

where addition is component-wise. This is simply because the set of realizations of θ is the union of the set of realizations of $\theta \wedge \psi$ with the set of realizations of $\theta \wedge \neg\psi$. Thus, we have either $\mathrm{Mult}(\theta \wedge \psi, k) = \mathrm{Mult}(\theta, k)$ or $\mathrm{Mult}(\theta \wedge \neg\psi, k) = \mathrm{Mult}(\theta, k)$.

Now, suppose that for every B_n-formula $\psi(\bar{x})$, we have either $\mathrm{Mult}(\theta \wedge \psi, k) = \mathrm{Mult}(\theta, k)$ or $\mathrm{Mult}(\theta \wedge \neg\psi, k) = \mathrm{Mult}(\theta, k)$. Then, again using the fact that $\mathrm{Mult}(\theta, k) = \mathrm{Mult}(\theta \wedge \psi, k) + \mathrm{Mult}(\theta \wedge \neg\psi, k)$, we have that for every ψ, either $\mathrm{Mult}(\theta \wedge \psi, k) = 0$ or $\mathrm{Mult}(\theta \wedge \neg\psi, k) = 0$. Thus, $MR(\theta \wedge \psi) < k$ or $MR(\theta \wedge \neg\psi) < k$. Therefore, there can be only one B_n-type of rank k that contains θ. □

Lemma 3.5. *For a B_n-formula $\theta(\bar{x})$, the condition that $\mathrm{mult}(\theta, \bar{z}) \geq l$ is $\Pi^0_1(T_{n+1})$.*

Proof. By definition, $\mathrm{mult}(\theta, \bar{z}) \geq l$ if and only if, partitioning the variables $\bar{x}$ of θ into $\bar{z}$ and $\bar{y}$, we have

$$\forall m(\exists^{\geq m} z_1) \ldots (\exists^{\geq m} z_k)(\exists^{\geq l} \bar{y})\theta(\bar{x}) \in T_{\exists_{n+1}}) \ .$$

Thus, the condition is $\Pi^0_1(T \cap \exists_{n+1})$. □

Lemma 3.6. *The set of indices for B_n-types is computable in $(T_{n+1})'$.*

Proof. It is computable to check that $\theta \in B_n$, and, by Lemma 2.4, it is computable in $(T_{n+1})'$ to see that $MR(\theta) = k$. Now, to check that θ is an index, $(T_{n+1})'$ begins by computing $\mathrm{Mult}(\theta, k)$, which it can by Lemma 3.5. In particular, since $MR(\theta) = k$, it is impossible to have $\mathrm{mult}(\theta, \bar{z}) = \infty$ for any $\bar{z}$ of length k. Thus, $(T_{n+1})'$ can simply compute the function $\alpha = \mathrm{Mult}(\theta, k)$. Now, by Lemma 3.4, (θ, k) is an index if and only if for all $\psi(\bar{x}) \in B_n$ either

- $\forall \bar{z} \subseteq \bar{x}$ of length k, $\mathrm{mult}(\theta \wedge \psi, \bar{z}) \geq \alpha(\bar{z})$, or
- $\forall \bar{z} \subseteq \bar{x}$ of length k, $\mathrm{mult}(\theta \wedge \neg\psi, \bar{z}) \geq \alpha(\bar{z})$.

It follows from Lemma 3.5 that this condition is $\Pi^0_1(T_{n+1})$, so $(T_{n+1})'$ can check it. □

We now state some properties of this indexing of types.

Lemma 3.7.

1. *The set of indices for B_n-types is computable from $(T_{n+1})'$. Therefore, it is Δ^0_n if $n > 1$, and it is Δ^0_2 if $n = 1$.*
2. *Given an index (θ, k) for a B_n-type, we can compute the B_n-type with index (θ, k) using T_{n+1}. Thus, for $n > 1$, it is Δ^0_{n-1} to compute a B_n-type given its index, and it is Δ^0_1 to compute a B_1-type given its index.*

Proof. Statement (1) is immediate from Lemma 3.6. For Statement (2), let (θ, k) be an index for a B_n-type P. Note that ψ is in p if and only if $\mathrm{MR}(\theta \wedge \psi) \geq k$. By Lemma 2.4, this condition is $\Pi^0_1(T_{n+1})$. Since $\psi \in p$ if and only if $\neg\psi \notin p$, the condition is also $\Sigma^0_1(T_{n+1})$, so p is computable from T_{n+1}, uniformly in the pair (θ, k). □

During our constructions, for $n > 1$, the Δ^0_n worker will assign P^{n-1}-indices to tuples, based in part on guesses at the P^n-indices assigned by the Δ^0_{n+1}-worker. This worker needs to check the consistency of a B_n-type that seems to have index (θ, k) with the B_{n-1}-type that has index (χ, l). For $n = 1$, the Δ^0_1 worker will decide atomic facts about tuples based on guesses at P^1-indices. This worker needs to check the consistency of a B_1-type

that seems to have index (θ, k) with a finitary quantifier-free formula. It will sometimes happen that (θ, k) is not actually a P^1-index. In the next two results, we describe what Δ^0_n can do to check consistency, and when $n = 1$ we say what this means in the case where (θ, k) is not a P^1-index. We begin with the case where $n > 1$.

Lemma 3.8. *For every n, there is a $(T_{n+1})'$-computable procudure (if $n > 1$, this is Δ^0_n) for checking whether pairs (θ, k) and (χ, l) satisfy the following conditions:*

1. *(θ, k) is a P^n-index for a B_n-type $p(\bar{x})$*
2. *(χ, l) is a P^{n-1}-index of a B_{n-1}-type $q(\bar{x}, \bar{y})$,*
3. *$p(\bar{x}) \cup q(\bar{x}, \bar{y})$ is consistent.*

Proof. By Lemma 3.7, it is computable in $(T_{n+1})'$ to verify the first two conditions. We now argue that Condition (3) is equivalent to

$$(\forall \psi \in q)(\mathrm{MR}(\theta(\bar{x}) \wedge (\exists \bar{y})\psi(\bar{x}, \bar{y})) \geq k) \; .$$

Certainly, Condition (3) implies this, since each of the formulas $\theta(\bar{x}) \wedge (\exists \bar{y})\psi(\bar{x}, \bar{y})$ must be contained in $p(\bar{x})$ and $\mathrm{MR}(p) = k$. Suppose $p(\bar{x}) \cup q(\bar{x}, \bar{y})$ is inconsistent. Then there exist a formula $\rho(\bar{x}) \in p(\bar{x})$ and a formula $\xi(\bar{x}, \bar{y}) \in q(\bar{x}, \bar{y})$ such that $\rho(\bar{x}) \wedge \xi(\bar{x}, \bar{y})$ is inconsistent. Then $\rho(\bar{x}) \wedge (\exists \bar{y})\xi(\bar{x}, \bar{y})$ is inconsistent. Since $\mathrm{MR}(\theta(\bar{x}) \wedge \neg\rho(\bar{x})) < k$, and $\theta(\bar{x}) \wedge (\exists \bar{y})\xi(\bar{x}, \bar{y})$ implies $\theta(\bar{x}) \wedge \neg\rho(\bar{x})$, we get that $\mathrm{MR}(\theta(\bar{x}) \wedge (\exists \bar{y})\xi(\bar{x}, \bar{y})) < k$. By Lemma 2.4, this condition is $\Pi^0_1(T_{n+1})$. □

For $n = 1$, the Δ^0_1 worker will enumerate the atomic diagram of a structure, based on guesses at the P^1-indices assigned by the Δ^0_2 worker. Having enumerated a finite part of the atomic diagram with conjunction $\delta(\bar{b}, \bar{d})$, the Δ^0_1 worker will be checking the consistency of the type indexed by $(\theta(\bar{x}), k)$, where θ is a B_1-formula, with the quantifier-free formula $\delta(\bar{x}, \bar{y})$. Since the set of P^1-indices is not Δ^0_1, at some points in the construction, the Δ^0_1 worker will be attempting to verify consistency, but (θ, k) will fail to determine a type. We describe here what we do computably to check consistency, and we say what this means in the case where $(\theta(\bar{x}), k)$ is not a P^1-index.

Lemma 3.9. *Assuming that (θ, k) is a P^1-index for a type $p(\bar{x})$, it is computable to say whether $p(\bar{x})$ is consistent with a given quantifier-free formula $\delta(\bar{x}, \bar{y})$.*

Proof. By Lemma 2.3, we can computably check whether an existential formula has rank k, but we cannot do this for a B_1 formula. By Lemma 3.7, it is Δ^0_2 to say that (θ, k) is a P^1-index. By Lemma 2.4, it is Π^0_1 to say that a B_1 formula has rank at least k, so it is Σ^0_1 to say that a B_1 formula has rank less than k. Assuming that $(\theta(\bar{x}), k)$ is a P^1-index for

$p(\bar{x})$, exactly one of $\theta(\bar{x}) \wedge (\exists \bar{y})\delta(\bar{x}, \bar{y})$, $\theta(\bar{x}) \wedge \neg(\exists \bar{y})\delta(\bar{x}, \bar{y})$ will have rank less than k. In the first case, $\delta(\bar{x}, \bar{y})$ is inconsistent with $p(\bar{x})$, and in the second, $\delta(\bar{x}, \bar{y})$ is consistent with $p(\bar{x})$. □

Not knowing whether $(\theta(\bar{x}), k)$ is a P^1-index, we effectively search for one of the following:

1. Σ^0_1 evidence that $MR(\theta(\bar{x}) \wedge (\exists \bar{y})\delta(\bar{x}, \bar{y})) < k$,
2. Σ^0_1 evidence that $MR(\theta(\bar{x}) \wedge \neg(\exists \bar{y})\delta(\bar{x}, \bar{y})) < k$,
3. a change in our Δ^0_2 approximation indicating that $(\theta(\bar{x}), k)$ may not be a P^1-index.

Throughout what follows, the versions of consistency in Lemmas 3.8 and 3.9 will suffice for our purposes.

4 Boundedly saturated models of an arithmetical theory

In this section, we suppose that T is Δ^0_N, where for $1 \leq n < N$, T_{n+2} is Δ^0_n, and we consider a model $\mathcal{M}$ that is N-saturated. We show that $\mathcal{M}$ has a computable copy. Harrington [10] and Khisamiev [12] showed that for a decidable $\aleph_1$-categorical theory, all models have decidable copies. This result, relativized, gives the following.

Lemma 4.1 (Harrington, Khisamiev)**.** *Suppose T is a strongly minimal theory. If T is Δ^0_N, then every model of T has a copy whose complete diagram is Δ^0_N.*

Note. We could equally well obtain Lemma 4.1 from a result of Goncharov [7] and Peretyat'kin [22] on homogeneous structures with decidable copies.

If T is computable, then Lemma 4.1 gives computable copies of all models. We suppose $N > 1$. For our theory T, we have the enumerations P^n for the B_n-types. All of our models are assumed to have universe ω. We think of the natural numbers as constants. We will consider models with tuples of elements labeled by indices for types.

Definition 4.2 (P^n-labeling)**.** Let $\mathcal{A}$ be a model of T. A *P^n-labeling* of $\mathcal{A}$ is a function assigning to each tuple $\bar{a}$ from $\mathcal{A}$ a P^n-index for the B_n-type realized by $\bar{a}$.

In our constructions, for each $n > 1$, we produce a model with a Δ^0_n P^{n-1}-labeling. For $n = 1$, we produce a model whose atomic diagram is Δ^0_1. The next lemma gives a labeled model on top.

Lemma 4.3. *Suppose $\mathcal{A}$ is a model whose complete diagram is Δ^0_N, for some $N > 1$. Then $\mathcal{A}$ has a Δ^0_{N+1} P^N-labeling.*

Proof. Since the complete diagram of $\mathcal{A}$ is Δ^0_N and the set of P^N-indices is Δ^0_{N+1}, we assign P^N-indices to tuples of elements as follows. For each tuple $\bar{a} \in \mathcal{A}$, we take the first P^N-index (θ, k) for a type $p(\bar{x})$ such that $(\forall \psi(\bar{x}) \in B_N)\left(\psi \in p_{(\theta,k)} \text{ iff } \mathcal{A} \models \psi(\bar{a})\right)$. We can do this using Δ^0_{N+1}. □

In the lemma below, we say how (for $n > 1$) to pass from an n-saturated model with a Δ^0_{n+1} P^n-labeling to an isomorphic copy with a Δ^0_n P^{n-1}-labeling.

Lemma 4.4 (First Pull-Down Lemma)**.** *Suppose $n > 1$. Let $\mathcal{A}$ be a model of T that is n-saturated. If $\mathcal{A}$ has a Δ^0_{n+1} P^n-labeling, then there is an isomorphic copy $\mathcal{B}$ with a Δ^0_n P^{n-1}-labeling.*

Proof. Using Δ^0_n, we guess at the P^n-labeling of $\mathcal{A}$, and we produce an isomorphic copy $\mathcal{B}$ with a P^{n-1}-labeling. The isomorphism f from $\mathcal{B}$ to $\mathcal{A}$ will be Δ^0_{n+1}. In particular, we assign tentative values of f, based upon guesses at the Δ^0_{n+1} P^n-labeling of $\mathcal{A}$. When a guess at the P^n-labeling changes, some values of f that we had defined earlier may become undefined, to be defined again. To build a bijective function, we must ensure that for each $d \in \mathcal{B}$, from some stage onwards, $f(d)$ is assigned and does not change. Similarly, we must ensure that for each $c \in \mathcal{A}$, from some stage onwards, $f^{-1}(c)$ is assigned and does not change. Thus, we have the following requirements.

(a) Determine $f^{-1}(c)$ for $c \in \mathcal{A}$.

(b) Determine $f(d)$ for $d \in \mathcal{B}$.

We arrange the requirements in a natural list of length ω. As usual for a finite injury construction, when one of the guesses behind our action on a particular requirement changes, we change what we have done for this requirement (the requirement is injured), and we start over on later requirements (they are re-initialized). This distinction is to ensure that the k^{th} helper requirement for a (b)-requirement acts after the previous requirements have settled–the (b)-requirement is re-initialized, so its next injury is the first since re-initialization.

At each stage, as we assign values for f, we always choose the first possible image, or pre-image, preserving what we have done for earlier requirements, and we always maintain consistency of the B_{n-1}-type assigned to a tuple $\bar{b}, \bar{d}$ with the current approximation at the B_n-types assigned to $f(\bar{b})$ and its sub-tuples. By Lemma 3.8, this consistency check is Δ^0_n.

At each stage, we have a tuple $\bar{b}$ on which f is tentatively defined, say $f(\bar{b}) = \bar{a}$, and we have determined P^{n-1}-indices assigning B_{n-1}-types to $\bar{b}, \bar{d}$ and its sub-tuples. Say that $q(\bar{u}, \bar{v})$ is the B_{n-1}-type of $\bar{b}, \bar{d}$. We believe that we have correctly guessed P^n-indices for

the B_n-types of $\bar{a}$ and its sub-tuples. We have checked that $q(\bar{u}, \bar{v})$ is consistent with these B_n-types.

Suppose the next requirement has type (a), defining $f^{-1}(c)$. It is easy to satisfy this requirement, assuming that we have correctly guessed the P^n-indices for the B_n-types of $\bar{a}, c$ and its sub-tuples. We first check whether there is some $d_i \in \bar{d}$ such that we can take $f(d_i) = c$ (i.e., we check whether $q(\bar{u}, \bar{v}) \cup \{x = v_i\}$ is consistent with the B_n-types of $\bar{a}, c$ and its sub-tuples). If so, then for the first such d_i, we let $f(d_i) = c$. If not, then we create a new element e in $\mathcal{B}$, set $f(e) = c$ and assign to $\bar{b}, \bar{d}, e$ a B_{n-1}-type $q'(\bar{u}, \bar{v}, x)$ that extends $q(\bar{u}, \bar{v})$ and is consistent with the B_n-types of $\bar{a}, c$ and its sub-tuples.

Suppose the next requirement has type (b), defining $f(d)$. One possible strategy for satisfying this requirement would be to first determine a B_n-type for $\bar{a}, f(d)$, where this type is generated by formulas in the B_n-type of $\bar{a}$ and $\exists_n$-formulas, and then to search for an element that satisfies this type. Instead, our strategy for the requirement of type (b) is to try possible f-images for d until we find one that works. To try a possible image e, we guess the P^n-indices of the B_n-types of $\bar{a}, e$ and its sub-tuples, and we check consistency with these B_n-types of the B_{n-1}-type that we have currently assigned to $\bar{b}, d$ and further elements $\bar{d}$. It may turn out that one of our guesses is wrong, resulting in injury to the requirement of type (b). We introduce helper requirements, which have the effect of putting formulas into the $\exists_n$-type of $\bar{b}, d$. We satisfy one of these helper requirements each time the requirement of type (b) is injured. There will be only finitely many injuries, and we are happy to settle on a value for $f(d)$ before actually completing the B_n-type. Here is the family of "helper" requirements.

(c)$_k$ Suppose the type (b) requirement to define $f(d)$ is injured for the k^{th} time (since it was last initialized). Let $q(\bar{u}, x, \bar{v})$ be the B_{n-1}-type that we have currently assigned to $\bar{b}, d$ and a further tuple $\bar{d}$, and let $\varphi_k(\bar{u}, x)$ be the k^{th} $\exists_n$-formula (in variables $\bar{u}, x$) in order of Gödel number. If it is possible to make $\varphi_k(\bar{b}, d)$ true in $\mathcal{B}$, while maintaining consistency with the B_n-types of $\bar{a}$ and its sub-tuples, then we do this.

We say exactly how (after the k^{th} injury to the requirement of defining $f(d)$) to satisfy the helper requirement (c)$_k$. Say $\varphi_k(\bar{b}, d) = (\exists \bar{y})\alpha(\bar{b}, d, \bar{y})$, where $\alpha(\bar{u}, x, \bar{y})$ is B_{n-1}. We check whether $q(\bar{u}, x, \bar{v}) \cup \{\varphi_k(\bar{u}, x)\}$ is consistent with the B_n-types of $\bar{a}$ and its sub-tuples. Lemma 3.7 guarantees that we can do this. If we have consistency, then we look for a P^{n-1}-index for a B_{n-1} type

$$q'(\bar{u}, x, \bar{v}, \bar{y}) \supseteq q(\bar{u}, x, \bar{v}) \cup \{\alpha(\bar{u}, x, \bar{y})\}$$

that is consistent with the B_n-types of $\bar{a}$ and its sub-tuples. If there is no such P^{n-1}-index, it is because we have incorrectly guessed the P^n-index for some sub-tuple of $\bar{a}$, so we see injury to a higher priority requirement.

Claim 4.5. The type (b) requirement to define $f(d)$ cannot be injured infinitely often.

Proof of Claim. Suppose the requirement is injured infinitely often. Eventually, we correctly guess the P^n-indices for the B_n-types of $\bar{a}$ and its sub-tuples. Since we considered all $\exists_n$-formulas $\varphi_k(\bar{u}, x)$, the B_n-type $p(\bar{u})$ assigned to $\bar{a}$, together with those $\varphi_k(\bar{u}, x)$ for which we have provided witnesses generates a complete B_n-type $p'(\bar{u}, x)$. By Lemma 1.7, there is some e realizing $p'(\bar{a}, x)$. Consider a stage s large enough that all higher priority requirements have settled, and so have our approximations to the P^n-indices for B_n-types assigned to sub-tuples of the initial segment $\bar{a}, \bar{a}', e$ of ω containing $\bar{a}, e$. Since e is a possible choice for $f(d)$, some element of $\bar{a}, \bar{a}', e$ is assigned as $f(d)$, with no possibility of further injury. This contradicts the assumption that the (b) requirement is injured infinitely often. □

□

There is a special Pull-Down Lemma for the case where $n = 1$. Lemma 4.4 does not work in this case, since Δ^0_1 cannot check consistency as in Lemma 3.8. However, Δ^0_1 just needs to produce the atomic diagram. We shall check consistency as in Lemma 3.9.

Lemma 4.6 (Second Pull-Down Lemma)**.** *Let $\mathcal{A}$ be a model of T that is 1-saturated. If $\mathcal{A}$ has a Δ^0_2 P^1-labeling, then there is an isomorphic copy $\mathcal{B}$ with a Δ^0_1 atomic diagram.*

Proof. Guessing at the Δ^0_2 P^1-labeling of $\mathcal{A}$, we give a Δ^0_1 atomic diagram of an isomorphic copy $\mathcal{B}$. The isomorphism f from $\mathcal{B}$ to $\mathcal{A}$ will be Δ^0_2. At each stage, we have tentatively determined a finite part of f, say $f(\bar{b}) = \bar{a}$, based on guesses at the P^1-indices assigned to $\bar{a}$ and its sub-tuples. At each stage, we have permanently decided a finite part of the atomic diagram, say the conjunction is $\delta(\bar{b}, \bar{d})$. We check that $\delta(\bar{b}, \bar{d})$ is consistent with the B_1-types of $\bar{a}$ and its sub-tuples, using the consistency test from Lemma 3.9. Assuming that our guesses at the P^1-indices for these types are actually P^1-indices, we know that $\delta(\bar{b}, \bar{d})$ is consistent with the corresponding types.

As in the First Pull-Down Lemma, we have the following requirements.

(a) Determine $f^{-1}(c)$.

(b) Determine $f(d)$.

At each stage, the finite part of $f(\bar{b}) = \bar{a}$ that we have tentatively determined satisfies an initial set of requirements, based on guesses at the P^1-indices of types assigned to sub-tuples of the range $\bar{a}$. First, suppose that the next requirement is of type (a), defining $f^{-1}(c)$. Apart from the consistency check, this is satisfied as in the First Pull-Down Lemma. Once we have correctly guessed the P^1-indices of the B_1-types assigned to $\bar{a}, c$ and its sub-tuples, we can define $f^{-1}(c)$, once and for all.

Next, consider a requirement of type (b), defining $f(d)$. Again, to satisfy the requirement, we could first build a B_1-type and then look for $f(d)$ satisfying this type. Instead, we just try possible images of d, one after another, and each time the current one is shown not to work, we try to make a certain existential formula true of $\bar{b}, d$. As in the proof of the First Pull-Down Lemma, we have a family of "helper" requirements:

(c)$_k$ Suppose $f(\bar{b}) = \bar{a}$ for earlier requirements, and the requirement to define $f(d)$ is injured for the k^{th} time. Let $\varphi_k(\bar{u}, x)$ be the k^{th} existential formula in variables $\bar{u}, x$ corresponding to $\bar{b}, d$. Then add a witness to make $\varphi_k(\bar{b}, d)$ true in $\mathcal{B}$, if possible.

Say that $\varphi_k(\bar{u}, x) = (\exists \bar{v})\alpha(\bar{u}, x, \bar{v})$ and $\delta(\bar{b}, d, \bar{d})$ is the conjunction of the current part of the atomic diagram. We have already tested the consistency of $\delta(\bar{u}, x, \bar{y})$ with the B_1-types of $\bar{a}$ and its initial segments. There are finitely many ways to determine equality on the variables $\bar{v}, \bar{y}$ and decide the atomic sub-formulas of $\alpha(\bar{u}, x, \bar{v})$ to produce an extension $\delta'(\bar{u}, x, \bar{y}, \bar{v})$ of $\delta(\bar{u}, x, \bar{y})$ that implies $\alpha(\bar{u}, x, \bar{v})$. We see if one of these $\delta'(\bar{u}, x, \bar{y}, \bar{v})$ is consistent with the B_1-types of all sub-tuples of $\bar{a}$. If so, then (for the first we find), we put $\delta'(\bar{b}, d, \bar{d}, \bar{d}')$ into the atomic diagram of $\mathcal{B}$, where the elements of $\bar{d}'$ that are not in $\bar{b}, d, \bar{d}$ are the first few new elements of ω.

As in Claim 4.5, the (c)$_k$ requirements ensure that the associated type (b) requirement is satisfied. □

Combining the lemmas above, we obtain the following theorem.

Theorem 4.7. *Let T be a Δ^0_N strongly minimal theory such that for $1 \leq n < N$, T_{n+2} is Δ^0_n. Then every N-saturated model of T has a recursive copy.*

Proof. By Lemma 4.1, each model has a copy whose complete diagram is Δ^0_N. Applying Lemma 4.3, we get a Δ^0_{N+1} P^N-labeling. Now, we work our way down. Given a model $\mathcal{A}_{n+1}$ with a Δ^0_{n+1} P^n-labeling, we apply Lemma 4.4 to get an isomorphic copy $\mathcal{A}_n$ with a Δ^0_n P^{n-1} labeling. Eventually, we come to a copy $\mathcal{A}_2$ with a Δ^0_2 P^1-labeling. Then we apply Lemma 4.6 to get a recursive copy $\mathcal{A}_1$. □

5 Models that are not boundedly-saturated

Let T be a strongly minimal theory with a model $\mathcal{M}$ that is not boundedly saturated. Let n be minimal such that $\mathcal{M}$ is not n-saturated. This means that there is a tuple $\bar{c}$ and a B_n-type $p(\bar{c}, x)$ that is consistent with the type of $\bar{c}$, but is not realized in $\mathcal{M}$. The goal of this section is to construct a copy of $\mathcal{M}$ with a Δ^0_n P^{n-1}-labeling. Since $\mathcal{M}$ is $(n-1)$-saturated, we can then apply Lemmas 4.4 and 4.6 to get a computable copy of $\mathcal{M}$.

Lemma 5.1. *If $p(\bar{c}, x)$ is a B_n-type consistent with the type of $\bar{c}$ and omitted in $\mathcal{M}$, then it can only be the generic B_n-type over $\bar{c}$. That is, for each B_n-formula $\psi(\bar{c}, x) \in p$, the formulas guaranteeing that $(\exists^{\infty} x)\psi(\bar{c}, x)$ holds are in the type of $\bar{c}$.*

Proof. Suppose $p(\bar{c}, x)$ is not the generic B_n-type over $\bar{c}$, say $\psi(\bar{c}, x) \in p(\bar{c}, x)$, where $(\exists^{=k} x)\psi(\bar{c}, x)$ is in the type of $\bar{c}$. Let $a_1, \ldots, a_k$ be the k elements of $\mathcal{M}$ satisfying $\psi(\bar{c}, x)$. We claim that some a_i satisfies all of $p(\bar{c}, x)$. Suppose not. Then each a_i satisfies some B_n-formula $\varphi_i(\bar{c}, x)$ that is not in $p(\bar{c}, x)$. It follows that $\mathcal{M} \models (\forall x)[\psi(\bar{c}, x) \rightarrow \bigvee_i \varphi_i(\bar{c}, x)]$, so the type of $\bar{c}$ includes the formula $(\forall x)[\psi(\bar{u}, x) \rightarrow \bigvee_i \varphi_i(\bar{c}, x)]$. However, $\bigvee_i \varphi_i(\bar{c}, x)$ is inconsistent with $p(\bar{c}, x)$. Thus, $p(\bar{c}, x)$ is inconsistent with the type of $\bar{c}$, a contradiction. We have shown that if $p(\bar{c}, x)$ is an algebraic B_n-type over $\bar{c}$, then it must be realized in $\mathcal{M}$. □

We assumed that $\mathcal{M}$ omits some B_n-type $p(\bar{c}, x)$ that is consistent with the type of $\bar{c}$. By Lemma 5.1, $p(\bar{c}, x)$ must be the generic B_n-type over $\bar{c}$. Since the generic type is omitted, every element of $\mathcal{M}$ is algebraic over $\bar{c}$ by a B_n-formula. As we mentioned earlier, this is every bit as useful as n-saturation.

Lemma 5.2. *If every element of $\mathcal{M}$ is algebraic over $\bar{c}$ by a B_n-formula, then $\mathcal{M}$ has a copy $\mathcal{A}$ whose B_n-diagram is recursive in $\mathrm{tp}_{B_{n+1}}(\bar{c})$.*

Proof. We produce the copy $\mathcal{A}$, determining the truth value of B_n-sentences $\varphi(\bar{c}, \bar{b})$ for larger and larger tuples $\bar{b}$. At each stage s, we will have committed to a finite set of B_n-sentences $\Phi_s := \{\varphi_i \mid i < l\}$. We ensure that for every element b mentioned, there is some $\varphi_j(\bar{c}, b)$ such that for some integer N, $(\exists^{=N} x)\varphi_j(\bar{c}, x) \in \mathrm{tp}_{B_{n+1}}(\bar{c})$. We also ensure that $(\exists \bar{x}) \wedge_{i<l} \varphi_i(\bar{c}, x_i) \in \mathrm{tp}_{B_{n+1}}(\bar{c})$. For each B_n-sentence $\varphi(\bar{c}, \bar{b})$, at some stage s, we add either $\varphi(\bar{c}, \bar{b})$ or its negation to Φ_s.

Finally, we have Henkin requirements, witnessing $\exists_{n+1}$-formulas in the type of $\bar{c}$. If $\psi(\bar{x}, \bar{y})$ is B_n and $(\exists \bar{y})\psi(\bar{x}, \bar{y}) \in \mathrm{tp}_{\exists_{n+1}}(\bar{c})$, then for some tuple $\bar{d}$ and some s, we put $\psi(\bar{c}, \bar{d})$ into Φ_s. Since in $\mathcal{M}$, every element is B_n-algebraic over $\bar{c}$, each of these requirements can be satisfied. Moreover, we can proceed recursively in $\mathrm{tp}_{B_{n+1}}(\bar{c})$.

Claim: $\Phi := \cup_s \Phi_s$ is the B_n-diagram of a structure $\mathcal{A}$. In particular, if Φ includes a B_n-sentence $(\exists \bar{x})\psi(\bar{c}, \bar{b}, \bar{x})$, then it also contains the sentence $\psi(\bar{c}, \bar{b}, \bar{d})$, for some $\bar{d}$.

Proof of Claim. We first show that for any B_n formula $\varphi(\bar{c}, \bar{y})$ and any m, if the formula $(\exists^{=m} \bar{y})\varphi(\bar{u}, \bar{y})$ is in $\mathrm{tp}_{B_{n+1}}(\bar{c})$, then there are exactly m distinct tuples $\bar{y}$ from $\mathcal{A}$ such that $\varphi(\bar{c}, \bar{y}) \in \Phi$. There is an $(\exists_{n+1})$-formula $\varphi^*(\bar{u})$ saying that there exist distinct tuples $\bar{y}_1, \ldots, \bar{y}_m$ satisfying $\varphi(\bar{u}, \bar{y})$. This formula is in the type of $\bar{c}$. The Henkin requirements guarantee that there exist distinct tuples $\bar{d}_1, \ldots, \bar{d}_m$ with $\varphi(\bar{c}, \bar{d}_i) \in \Phi$. Consistency of Φ with the B_{n+1}-type of $\bar{c}$ guarantees that there cannot be more.

Now, consider the B_n-sentence $(\exists \bar{x})\psi(\bar{c}, \bar{b}, \bar{x})$ in Φ. Then $(\exists \bar{y})(\exists \bar{x})\psi(\bar{c}, \bar{y}, \bar{x})$ is true of $\bar{c}$, so it is satisfied in $\mathcal{M}$. Adding conjuncts to ψ, if necessary, we may suppose that $\psi(\bar{c}, \bar{y}, \bar{x})$ is satisfied by just m distinct tuples. The previous paragraph shows that there are exactly m tuples $\bar{b}_i, \bar{d}_i$ such that $\psi(\bar{c}, \bar{b}_i, \bar{d}_i) \in \Phi$. Now, $\bar{b}$ must be one of the $\bar{b}_i$, as otherwise $\{\psi(\bar{c}, \bar{b}_i, \bar{d}_i) \mid i < m\} \cup \{(\exists \bar{x})\psi(\bar{c}, \bar{b}, \bar{x})\}$ would be a subset of Φ that is inconsistent with the B_{n+1}-type of $\bar{c}$. Thus, we have $\psi(\bar{c}, \bar{b}, \bar{d}_i) \in \Phi$, where $\bar{b} = \bar{b}_i$. □

It remains to show that the structure $\mathcal{A}$ that we have built is isomorphic to the given $\mathcal{M}$. For this, we use a standard König's Lemma argument. We build a tree whose paths will be isomorphisms between $(\mathcal{A}, \bar{c})$ and $(\mathcal{M}, \bar{c})$. We construct a tree of viable partial functions from $\mathcal{A}$ into $\mathcal{M}$, putting a finite function p with $\bar{c} \subset \mathrm{dom}(p)$ into the tree if the B_n-type of $\bar{b} := \mathrm{dom}(p)$ is the same as that of $p(\bar{b})$. At level n, the functions are defined on $\bar{c}$ and the first n elements of $\mathcal{A}$ (in the ω-ordering).

The tree is finitely branching, since the elements of $\mathcal{A}$ are all algebraic over $\bar{c}$ via B_n-formulas. The tree is infinite. For any tuple $\bar{b}$ in $\mathcal{A}$, the B_n-type of $\bar{b}$ over $\bar{c}$ in $\mathcal{A}$ is consistent with the type of $\bar{c}$. Since the B_n-type of $\bar{b}$ is algebraic over $\bar{c}$, it must be realized in $(\mathcal{M}, \bar{c})$, by Lemma 5.1. Therefore, we have a viable function p defined on $\bar{b}$. Now, König's Lemma yields a path through the tree. This path gives an embedding $f : \mathcal{A} \to \mathcal{M}$. We now show that the function f is onto. Suppose $a \in \mathcal{M}$ is one of N elements satisfying the B_n-formula $\varphi(\bar{c}, x)$. The Henkin requirements guarantee that $\mathcal{A} \models (\exists^{\geq N} x)\varphi(\bar{c}, x)$. Now, f maps elements satisfying $\varphi(\bar{c}, x)$ to elements satisfying $\varphi(\bar{c}, x)$, so a is in the range. Therefore, f must be onto, and it is an isomorphism. □

If $n = 0$ or $n = 1$, then Lemma 5.2 suffices to give a recursive copy of $\mathcal{M}$, since $\mathrm{tp}_{B_2}(\bar{c})$ is Δ^0_1, by Lemma 3.7. Thus, we may assume $n > 1$. It may at first seem that having a Δ^0_n B_n-diagram should be enough to give a Δ^0_n P^{n-1}-labeling, but this seems not to suffice. We want a construction like the one in Lemma 5.2 except that we assign B_{n-1}-types in addition to B_n-formulas. The only obstruction to carrying out the construction directly, as in Lemma 5.2, is that we need a Δ^0_n way to check whether the type with a certain P^{n-1} index is consistent with a given B_n-formula over $\bar{c}$. The remainder of the current section is devoted to this. In fact, we will prove a stronger result, yielding a copy of $\mathcal{M}$ with a Δ^0_n P^n-labeling. By Lemma 3.8, it is Δ^0_n to turn this into a P^{n-1}-labeling.

Before we begin, we isolate one step in the argument, since it reappears in later constructions. Essentially, the remainder of the argument is focused on extending this result from generic tuples $\bar{h}$ to our tuple $\bar{c}$.

Lemma 5.3. *For $n > 1$, if $\bar{h}$ is a generic tuple, let S be the set of pairs $(\varphi(\bar{h}, \bar{x}), q(\bar{h}, \bar{y}))$ such that $\varphi(\bar{u}, \bar{x})$ is an $\exists_{n+1}$-formula, $q(\bar{u}, \bar{y})$ is either a B_n-type or a B_{n-1}-type (given by its P^n or P^{n-1}-index), and $\{\varphi(\bar{h}, \bar{x})\} \cup q(\bar{h}, \bar{y})$ is consistent with the type of $\bar{h}$. Then S is computable in $(T_{n+1})'$, so it is Δ^0_n, uniformly in n and the length of $\bar{h}$.*

Note that in Lemma 5.3, $\bar{x}$ and $\bar{y}$ need not be disjoint sets of variables.

Proof. By Lemma 2.6, since $\bar{h}$ is generic, its $\exists_{n+1}$-type is $\Pi^0_1(T_{n+1})$. A B_n- or B_{n-1}-type $q(\bar{h}, \bar{y})$ is consistent with $\varphi(\bar{h}, \bar{x})$ over $\bar{h}$ iff

$$(\forall \psi \in q)\left((\exists \bar{x}, \bar{y})(\varphi(\bar{h}, \bar{x}) \wedge \psi(\bar{h}, \bar{y})) \in \mathrm{tp}_{\exists_{n+1}}(\bar{h})\right).$$

This is $\Pi^0_1(T_{n+1})$, so it can be computed from $(T_{n+1})'$. □

The following is a variant of Lemma 5.3, which applies to any tuple $\bar{c}$, but only applies to types for tuples $\bar{x}$ over $\bar{c}$ provided that each x_i is B_n-algebraic over $\bar{c}$. This will let us use essentially the same proof as in Lemma 5.2 to produce a copy of $\mathcal{M}$ with a Δ^0_n P^n-labeling.

Lemma 5.4. *Let $\bar{c}$ be any tuple. Let S be the set of pairs*

$$(\bigwedge_{i<|\bar{x}|} \varphi_i(\bar{c}; x_i) \wedge \varphi(\bar{c}, \bar{x}), r(\bar{c}, \bar{x}))$$

such that φ is an $\exists_{n+1}$-formula, each φ_i is a B_n-formula that is algebraic over $\bar{c}$, r is a B_n-type (given by its P^n-index), and $r(\bar{c}, \bar{x}) \cup \{\bigwedge_{i<|\bar{x}|} \varphi_i(\bar{c}; x_i) \wedge \varphi(\bar{c}, \bar{x})\}$ is consistent with the type of $\bar{c}$. Then S is computable from $(T_{n+1})' \oplus T_{n+2}$, so it is Δ^0_n.

Note that Lemma 5.4 also holds for B_{n-1}-types r. We can show this either by the same proof as for B_n-types (given below), or by considering the consistent B_n-types and taking their restrictions to B_{n-1}-types.

Proof. Our first step is to partition $\bar{c}$ into two pieces, $\bar{g}$ and $\bar{b}$.

Claim 5.5. There exists a partition of $\bar{c}$ into two pieces $\bar{g}$ and $\bar{b}$ such that the following conditions hold.

1. $\bar{g}$ is a maximal sub-tuple of $\bar{c}$ realizing the generic $\forall_{n+1}$-type—it need not satisfy the generic B_{n+1}-type,
2. $\bar{b}$ is a tuple algebraic over $\bar{g}$ by an $\exists_{n+1}$-formula $\Theta(\bar{g}, \bar{x})$,
3. For a generic tuple $\bar{h}$, there are only finitely many tuples satisfying $\Theta(\bar{h}, \bar{x})$.

Proof. [Proof of Claim 5.5] Let $\bar{g}$ be a maximal sub-tuple of $\bar{c}$ realizing the generic $\forall_{n+1}$-type. Let $\bar{b}$ consist of the remaining elements of $\bar{c}$. We will show that this partition satisfies the three conditions. Given an element c_i of $\bar{c} \smallsetminus \bar{g}$, we show that c_i is algebraic over $\bar{g}$ by an $\exists_{n+1}$-formula. Since the type of $\bar{g}, c_i$ does not contain all of the formulas of the generic $\forall_{n+1}$-type, there is some $\exists_{n+1}$-formula true of $\bar{g}, c_i$ and not true of a generic tuple.

Sub-claim: If $\bar{h}$ is generic of the same length as $\bar{g}$, and $\varphi(\bar{h}, x)$ is algebraic, where $\varphi(\bar{u}, x)$ is $\exists_{n+1}$, then $\varphi(\bar{g}, x)$ is also algebraic.

Proof of Sub-claim. The generic tuple $\bar{h}$ must satisfy $\neg(\exists^{\geq N}x)\varphi(\bar{u},x)$, for some N. The formula $\neg(\exists^{\geq N}x)\varphi(\bar{u},x)$ is logically equivalent to a $\forall_{n+1}$-formula, so it satisfied by $\bar{g}$ as well. □

The conjunction of algebraic $\exists_{n+1}$-formulas $\varphi_i(\bar{g},x_i)$ as in the Sub-claim, one for each $c_i \in \bar{c} \smallsetminus \bar{g}$, gives the formula $\Theta(\bar{g},\bar{x})$ required for Conditions (2) and (3) in Claim 5.5. □

Having proved Claim 5.5, we return to the proof of Lemma 5.4. Taking the partition $\bar{g},\bar{b}$ from the Claim, we will say how Δ^0_n can determine consistency of a formula $\varphi(\bar{g},\bar{b},\bar{x}) \wedge \wedge_{i<|\bar{x}|}\varphi_i(\bar{g},\bar{b},x_i)$, where $\varphi(\bar{g},\bar{b},\bar{x})$ is an $\exists_{n+1}$-formula, the formulas $\varphi_i(\bar{g},\bar{b},x_i)$, one for each x_i in $\bar{x}$, are algebraic B_n-formulas, and $r(\bar{g},\bar{b},\bar{x})$ is a B_{n-1}-type, given by its P^{n-1}-index. Let $\bar{z}$ correspond to $\bar{g}$, and let $\bar{y}$ correspond to $\bar{b}$. Consider the formula

$$\xi(\bar{z};\bar{y},\bar{x}) := \bigwedge_{i<|\bar{x}|} \varphi_i(\bar{z};\bar{y},x_i) \wedge \varphi(\bar{z};\bar{y},\bar{x}) \wedge \Theta(\bar{z};\bar{y}) \wedge \bigwedge_{i<|\bar{x}|} (\exists^{<\infty}u_i)\varphi_i(\bar{z},\bar{y},u_i).$$

We can replace $(\exists^{<\infty}u_i)\varphi_i(\bar{z},\bar{y},u_i)$ by $(\exists^{\geq N}u_i)\neg\varphi_i(\bar{z},\bar{y},u_i)$, for an appropriate N. By Lemma 2.1, T_{n+1} can identify ξ with an equivalent $\exists_{n+1}$-formula.

We note that for a generic tuple $\bar{h}$, the formula $\xi(\bar{h};\bar{y},\bar{x})$ must be algebraic. In fact, we see that $\Theta(\bar{h};\bar{y}) \wedge \varphi_i(\bar{h};\bar{y},x_i) \wedge \bigwedge_i(\exists^{<\infty}u_i)\varphi_i(\bar{h},\bar{y},u_i)$ is algebraic for each i. This is because $\Theta(\bar{h};\bar{y})$ is an algebraic formula in the variables $\bar{y}$ and there are only finitely many realizations of the φ_i over any $\bar{y}$ satisfying $\bigwedge_i(\exists^{<\infty}u_i)\varphi_i(\bar{h},\bar{y},u_i)$. By the sub-claim above, $\xi(\bar{g};\bar{y},\bar{x})$ is also an algebraic formula.

Since $\xi(\bar{h};\bar{y},\bar{x})$ is a $\exists_{n+1}$-formula, Lemma 2.6 shows that it is computable from $(T_{n+1})'$ to check whether the $\exists_{n+1}$-formulas $\exists^{\geq k}(\bar{y},\bar{x})\xi(\bar{h};\bar{y},\bar{x})$ are true. Let K be the number of distinct tuples $\bar{y},\bar{x}$ satisfying the formula $\xi(\bar{h};\bar{y},\bar{x})$. We consider the formula

$$\chi(\bar{h},\bar{y}_1,\bar{x}_1,\ldots,\bar{y}_K,\bar{x}_K) := \bigwedge_{i\leq K} \xi(\bar{h},\bar{y}_i,\bar{x}_i) \wedge \bigwedge_{i<j\leq K} (\bar{x}_i\bar{y}_i \neq \bar{x}_j\bar{y}_j)\ .$$

This is an $\exists_{n+1}$-formula that is satisfied. Now, we use Lemma 5.3 to find an index of a B_n-type $q(\bar{h};\bar{y}_1,\bar{x}_1,\ldots,\bar{y}_K,\bar{x}_K)$ consistent with χ over $\bar{h}$.

Since $\mathrm{tp}_{\exists_{n+1}}(\bar{g}) \subseteq \mathrm{tp}_{\exists_{n+1}}(\bar{h})$, if $s(\bar{w},\bar{y},\bar{x})$ is a B_n-type, and $s(\bar{g},\bar{y},\bar{x}) \cup \{\xi(\bar{g};\bar{y},\bar{x})\}$ is consistent, then $s(\bar{h},\bar{y},\bar{x}) \cup \{\xi(\bar{h};\bar{y},\bar{x}\}$ is also consistent. Thus, any B_n-type $r(\bar{w},\bar{y},\bar{x})$ such that $r(\bar{g},\bar{y},\bar{x}) \cup \{\xi(\bar{g};\bar{y},\bar{x})\}$ is consistent is a restriction of $q(\bar{w};\bar{y}_1,\bar{x}_1,\ldots,\bar{y}_K,\bar{x}_K)$ to the variables $\bar{w},\bar{y}_i,\bar{x}_i$ for some $i \leq K$. There are only finitely many such distinct B_n-types, which we index as $q_j(\bar{w},\bar{y},\bar{x})$. Further, Lemma 3.7 shows that it is computable from $(T_{n+1})'$ to find P^n-indices for these types and to determine whether $q_j = q_i$ for each pair $j \neq i$. We do this and remove any repetitions in our list.

We claim that it is computable from $(T_{n+1})'$ to say that $q_j(\bar{g},\bar{b},\bar{x}) \cup \{\xi(\bar{g};\bar{b},\bar{x})\}$ is consistent. For each of the B_n-types $q_j(\bar{w},\bar{y},\bar{x})$, we find a formula $\alpha_j(\bar{w},\bar{y},\bar{x})$ that is in

the type q_j but not in any of the others. Now, by Lemma 3.7, it is computable from T_{n+2} to determine, for each j, whether $\{\alpha_j(\bar{g}, \bar{b}, \bar{x}), \xi(\bar{g}; \bar{b}, \bar{x})\}$ is consistent, since we only need to check whether a single formula is in the $\exists_{n+1}$-type of $\bar{c}$. If $\{\alpha_j(\bar{g}, \bar{b}, \bar{x}), \xi(\bar{g}; \bar{b}, \bar{x})\}$ is consistent, then $q_j(\bar{g}, \bar{b}, \bar{x}) \cup \{\xi(\bar{g}; \bar{b}, \bar{x})\}$ is consistent as well, since some B_{n-1}-type must extend $\{\alpha_j(\bar{g}, \bar{b}, \bar{x}), \xi(\bar{g}, \bar{b}, \bar{x})\}$ consistently, and the only possibility is q_j.

We have computed a finite list of P^n-indices for types q_j that are consistent with $\bigwedge_{i<|\bar{x}|} \varphi_i(\bar{c}; x_i) \wedge \varphi(\bar{c}, \bar{x})$ and the type of $\bar{c}$, and our finite list represents all such types. Given any P^n-index, it is computable from $(T_{n+1})'$ to check whether it gives another index for one of these types. If so, then the given index is consistent, and if not, it is not. □

Next, we show how an argument like that in Lemma 5.2 suffices to yield a computable copy of $\mathcal{M}$.

Lemma 5.6. *Let T be a strongly minimal theory, and let $\mathcal{M}$ be a model that is not n-saturated. Then there is a copy of $\mathcal{M}$ with a P^n-labeling computable in $(T_{n+1})' \oplus T_{n+2}$.*

Proof. As in Lemma 5.2, we fix a tuple $\bar{c}$ so that every element of $\mathcal{M}$ is algebraic over $\bar{c}$ via a B_n-formula. We produce a copy of $\mathcal{M}$ with a Δ^0_n P^n-labeling. We use the fact that $\mathrm{tp}_{B_{n+1}}(\bar{c})$ is computable in T_{n+2}, by Lemma 3.7. At each stage s, we will have a finite tuple $M_s = \bar{c}, \bar{y}$ with a P^n-labeling that is consistent with the type of $\bar{c}$. Let $p(\bar{c}, \bar{y})$ be the B_n-type given by the P^n-labeling. Further, for each $y_i \in \bar{y}$, we have a specified B_n-formula $\rho_i(\bar{c}, x)$ that is algebraic over $\bar{c}$ and, according to the P^n-labeling, true of y_i. We then consider the s^{th} B_n-formula $\psi(\bar{c}, x)$. We first search $\mathrm{tp}_{B_{n+1}}(\bar{c})$ for a K such that $(\exists^{=K} x)\psi(\bar{c}, x)$ or $(\exists^{=K})\neg\psi(\bar{c}, x)$ is in the B_{n+1}-type of $\bar{c}$. Without loss of generality, we assume the former. Next, we let l be the number of elements in M_s satisfying $\psi(\bar{c}, x)$. For a tuple of variables $\bar{z}$ of length $K - l$, let $\delta(\bar{c}, \bar{y}, \bar{z})$ be a quantifier-free formula saying that for $z_j \in \bar{z}$, $z_j \notin M_s$. We search for a P^n-index for a type $r(\bar{c}, \bar{y}, \bar{z})$, such that the following hold:

- the formula $\bigwedge_{y_i \in \bar{y}} \rho_i(\bar{c}, y_i) \wedge \delta(\bar{c}, \bar{y}, \bar{z}) \wedge \bigwedge_{z_j \in \bar{z}} \psi(\bar{c}, z_j)$ is consistent with $r(\bar{c}, \bar{y}, \bar{z})$ over the type of $\bar{c}$.
- The restriction of $r(\bar{c}, \bar{y}, \bar{z})$ to the variables $\bar{c}, \bar{y}$ is equal to $p(\bar{c}, \bar{y})$.

There must be such a type. To see this, recall that, by Lemma 5.1, since $p(\bar{c}, \bar{u})$ is algebraic, it must be realized by some $\bar{y}$ inside $\mathcal{M}$. Since there are only l elements that satisfy $\psi(\bar{c}, x)$ inside the tuple $\bar{c}, \bar{y}$ in $\mathcal{M}$, there must be $K - l$ other elements that satisfy $\psi(\bar{c}, x)$. If $\bar{y}$ realizes $p(\bar{c}, \bar{u})$ in $\mathcal{M}$ and $\bar{z}$ consists of the $K - l$ elements outside $\bar{c}, \bar{y}$ for which $\psi(\bar{c}, x)$ holds in $\mathcal{M}$, then the B_n-type of $\bar{c}, \bar{y}, \bar{z}$ satisfies the conditions. By Lemma 5.4, it is computable in $(T_{n+1})' \oplus T_{n+2}$ to find a P^n index for the type. We assign

this index to $\bar{c}, \bar{y}, \bar{z}$, determine $\rho_i(\bar{c}, z_i) = \psi(\bar{c}, z_i)$ for each $z_i \in \bar{z}$, assign consistent P^n-types to sub-tuples, and let this be M_{s+1}. This builds a P^n-labeling of a structure isomorphic to $\mathcal{M}$, exactly as in the proof of Lemma 5.2. □

Theorem 5.7. *Let T be a strongly minimal theory such that T_{k+2} is Δ^0_k uniformly in k. Let $\mathcal{M}$ be a model of T that is not boundedly saturated. Then there is a computable copy of $\mathcal{M}$.*

Proof. Let n be least so that $\mathcal{M}$ is not n-saturated. If $n = 0$ or $n = 1$, then the result follows from Lemma 5.2, so we suppose $n > 1$. In Lemma 5.6, we showed that there is a P^n-labeling of a copy of $\mathcal{M}$ that is computable in $(T_{n+1})' \oplus T_{n+2}$, so it is Δ^0_n. By Lemma 3.8, it is Δ^0_n to determine the consistency of P^n-types with P^{n-1}-types. Thus, it is Δ^0_n to turn this into a Δ^0_n P^{n-1}-labeling. Since $\mathcal{M}$ is m-saturated for every $m < n$, we can use Lemmas 4.4 and 4.6 to build a computable copy of $\mathcal{M}$. □

6 Saturated models of non-arithmetical theories

The cases that remain are where the theory T is not arithmetical, and the given model $\mathcal{M}$ is either saturated or else boundedly saturated but of finite dimension. Since the theory T is not arithmetical, we do not have a "top" model as we did in Section 4. To produce a computable copy of $\mathcal{M}$, we will use a worker construction. The Δ^0_n worker produces a structure $\mathcal{A}_n$, based on guesses at the structure $\mathcal{A}_{n+1}$ produced by the Δ^0_{n+1} worker. We want an isomorphism f_n from $\mathcal{A}_n$ onto $\mathcal{A}_{n+1}$. At each level n, the construction of $\mathcal{A}_n$ will involve requirements of type (a), putting c into $\mathrm{ran}(f_n)$, and type (b), putting d into $\mathrm{dom}(f_n)$. We will need to be more careful with the type (b) requirements than we were when the theory was arithmetical. The Δ^0_{n+1} worker will have extra saturation requirements, so that the Δ^0_n worker can satisfy the type (b) requirements.

If we are building a copy of the saturated model, it is always fine to add more generics. When we are building a finite dimensional model, we will need to ensure that every element is algebraic over a fixed basis. That will take more work. In this section, our goal is to build a computable copy of the countable saturated model. Since we have Lemmas 4.4 and 4.6, it is enough to prove the following.

Theorem 6.1. *Let T be a strongly minimal theory such that T_{n+2} is Δ^0_n uniformly in n. Then there is a copy of the countable saturated model with a Δ^0_3 P^2-labeling.*

Proof. Throughout the construction, for each $n \geq 3$, the Δ^0_n worker builds a structure $\mathcal{A}_n$, with a P^{n-1} labeling. The universe of each $\mathcal{A}_n$ is ω. The Δ^0_n worker guesses at the P^n-labeling of $\mathcal{A}_{n+1}$ and attempts to make $\mathcal{A}_n$ isomorphic to $\mathcal{A}_{n+1}$. We will show that all

$\mathcal{A}_n$ are isomorphic, and that the common isomorphism type is that of the saturated model of T.

The Δ^0_n worker assigns B_{n-1}-types to tuples, while guessing at the sequence of B_n-types assigned by the Δ^0_{n+1} worker. The goal is to produce an isomorphism f_n from $\mathcal{A}_n$ to $\mathcal{A}_{n+1}$. We have the following requirements.

(a) Find an f_n-pre-image for some $c \in \mathcal{A}_{n+1}$.

(b) Find an f_n-image for some $d \in \mathcal{A}_n$.

(c) If $r(\bar{x})$ is the B_{n-1}-type assigned to $\bar{e}$, and $s(\bar{x}, y)$ is a B_{n-1}-type generated by $r(\bar{x})$ and further $\exists_{n-1}$-formulas, then realize $s(\bar{e}, \bar{y})$.

(d) Contribute towards ensuring that $\mathcal{A}_n$ has dimension at least n.

We begin the construction with requirement (d). The Δ^0_n worker assigns to the first n elements (in the usual ordering of ω) the generic B_{n-1}-type. This type has P^{n-1}-index $(\bigwedge_{i<n} x_i = x_i, n)$. Further, we set $f_n(i) = i$ for each $i < n$. It is guessed that the B_n-type of $f_n(0, \ldots, n-1)$ has P^n-index $(\bigwedge_{i<n} x_i = x_i, n)$. Since the Δ^0_{n+1} worker assigns to the first $n+1$ elements the generic B_n-type, this guess is correct. The rest of the Δ^0_n construction proceeds over this permanent assignment of $f_n(0, \ldots, n-1)$.

Note that P^{n-1}-indices for types r and s give rise to a (c)-requirement if and only if $r \subset s$ and for all $\exists_{n-1}$-formulas $\varphi(\bar{x}, y)$,

$$\varphi \in s \text{ OR } (\exists \psi(\bar{x}, y) \in (s \cap \exists_{n-1}))\, ((\forall y)\, (\psi(\bar{x}, y) \to \neg\varphi(\bar{x}, y)) \in r)\,.$$

The displayed condition is $\Sigma^0_1(r \oplus s)$. This is $\Sigma^0_1(\Delta^0_{n-2})$, so it is Σ^0_{n-2}. Thus, saying that the condition holds for all $\exists_{n-1}$-formulas $\varphi(\bar{x}, y)$ is Π^0_{n-1}. So, the set of pairs of P^{n-1}-indices for (c)-requirements is Δ^0_n.

Note: Our reason for starting with the Δ^0_3 worker is to ensure that $\Pi^0_2(\Delta^0_{n-2})$ is Π^0_{n-1}.

The role of the Δ^0_n (c)-requirements is to ensure that the Δ^0_{n-1} (b)-requirements can be satisfied. This is explained in more detail below. At each stage in the construction, we have f_n mapping a tuple $\bar{b}$ in $\mathcal{A}_n$ (which we are building) to a tuple $\bar{a} \in \mathcal{A}_{n+1}$ that we believe is assigned P^n-type $p(\bar{u})$, and we have assigned the P^{n-1}-index of a type $q(\bar{u}, x, \bar{v})$ to a tuple $\bar{b}, d, \bar{d}$ in $\mathcal{A}_n$. We believe, based on guesses at the P^n-indices, that the B_{n-1}-type q is consistent with the B_n-types assigned to $f_n(\bar{b}')$ for all sub-tuples $\bar{b}'$ of $\bar{b}$.

For a requirement of type (a), we need to find a pre-image for c, where c is the least constant not in $\bar{a}$. We guess the B_n-type $p'(\bar{u}, x)$ of $\bar{a}, c$. We either map some $e \in d, \bar{d}$ to c, or else choose an appropriate type $q'(\bar{u}, x, \bar{v}, y)$ for the first new constant e over $\bar{b}, \bar{d}$,

and map e to c. We take the first possible pre-image for c, preserving what we have done for earlier requirements (always maintaining consistency of the B_{n-1}-types that we are assigning with the B_n-types that the isomorphism f_n gives to the sub-tuples of $\bar{b}, c$).

For a requirement of type (b), we need an f_n-image for an element d. As in the case where T is arithmetical, we try possible images e_i, guessing the P^n-index of the type assigned to $\bar{a}, e_i$ and checking that this type is consistent with the current B_{n-1}-type that we have assigned to $\bar{b}, d$ and the current $\bar{d}$. To handle the injury to the (b) requirement, we add the family of "helper" requirements.

(e)$_k$ If the type (b) requirement to define $f_n(d)$ is injured for the k^{th} time, where for earlier requirements, f_n assigns the B_n-type $p(\bar{u})$ to $\bar{b}$, and $\varphi_k(\bar{u}, x)$ is the k^{th} $\exists_n$-formula in variables $\bar{u}, x$, then make $\varphi(\bar{b}, d)$ true in $\mathcal{A}_n$, if possible. That is, if $q(\bar{u}, x, \bar{v})$ is the B_{n-1}-type that we have currently assigned to $\bar{b}, d$ and further constants $\bar{d}$, and $p(\bar{u}) \cup q(\bar{u}, x, \bar{v}) \cup \{\varphi_k(\bar{u}, x)\}$ is consistent, then assign a B_{n-1}-type $q'(\bar{u}, x, \bar{v}, \bar{v}')$ (extending q) to a larger tuple $\bar{b}, d, \bar{d}, \bar{d}'$ so as to witness that $\varphi_k(\bar{b}, d)$ is true.

After finitely many steps, we settle on an image for $f_n(d)$, and we find a P^n-index. In Lemma 6.3 below, we argue that the actions for requirements (e)$_k$ ensure that the requirements of type (b) are satisfied, assuming that those of type (c) at level $n + 1$ are satisfied.

To see that the type (c) requirements are satisfied at every level, note that by Lemma 1.7, if a tuple $\bar{e}$ is assigned the B_{n-1}-type $r(\bar{x})$, and s is a B_{n-1}-type generated by formulas in $r(\bar{x})$ and $\exists_{n-1}$-formulas, then at any stage, regardless of which B_n-type $r'(\bar{x})$ is assigned to $f(\bar{e})$, $r'(\bar{x}) \cup s(\bar{x}, y)$ is consistent. Thus, in a saturated model, every realization of $r'(\bar{x})$ extends to a realization of $r'(\bar{x}) \cup s(\bar{x}, y)$. This shows that, regardless of what other consistent commitments we make, we can always find a consistent way to add a realization of $s(\bar{e}, y)$.

Lemma 6.2. *The P^{n-1}-labeling created by the Δ^0_n worker gives a well-defined structure $\mathcal{A}_n$ that has this P^{n-1}-labeling.*

Proof. Recall that $n \geq 3$. Let $\mathcal{A}_n$ be the structure such that for each relation symbol R and tuple $\bar{c}$, $\mathcal{A}_n \models R(\bar{c})$ if and only if the formula $R(\bar{x})$ is in the B_{n-1}-type assigned to $\bar{c}$. We show that for all formulas $\psi(\bar{x})$ that are $\exists_k$ or $\forall_k$ for some $k < n$ and for all appropriate tuples $\bar{c}$, $\mathcal{A}_n \models \psi(\bar{c})$ if and only if $\psi(\bar{x})$ is in the B_{n-1}-type assigned to $\bar{c}$. We suppose that $\psi(\bar{x})$ is in prenex normal form, and we proceed by induction on the total number of quantifiers.

If ψ is atomic, $\mathcal{A}_n \models \psi(\bar{c})$ if and only if $\psi(\bar{x})$ is in the B_{n-1}-type assigned to $\bar{c}$, by definition. We assume the statement is true for all B_{n-1}-formulas ψ with fewer than l

quantifiers. Suppose the B_{n-1}-formula $(\forall x)\psi(\bar{u}, x)$ is in the type assigned to $\bar{c}$. To show that $\mathcal{A}_n \models (\forall x)\psi(\bar{c}, x)$, we note that if the negation held, then $\neg\psi(\bar{c}, x)$ would be satisfied by some element a. By the inductive hypothesis, the B_{n-1}-type assigned to $\bar{c}, a$ includes the formula $\neg\psi(\bar{u}, x)$. This implies that the assigned types are inconsistent. Similarly, suppose the B_{n-1}-formula $(\exists x)\psi(\bar{u}, x)$ is in the type assigned to $\bar{c}$. To show that $\mathcal{A}_n \models (\exists x)\psi(\bar{c}, x)$, we use the fact that the construction succeeds in all requirements of type (c). Lemma 1.7 shows that some type $q(\bar{u}, x)$ containing $\psi(\bar{u}, x)$ is assigned to $\bar{c}$ and some a. By the inductive hypothesis, $\mathcal{A}_n \models \psi(\bar{c}, a)$, so $\mathcal{A}_n \models (\exists x)\psi(\bar{c}, x)$. □

Now that the model $\mathcal{A}_n$ is well-defined, it makes sense, for example, to refer to the B_n-type of a tuple in $\mathcal{A}_n$, even though the type is not assigned by the Δ^0_n worker.

Lemma 6.3. *For all n, all requirements at level n are satisfied. Hence, for all n, $\mathcal{A}_n$ is isomorphic to $\mathcal{A}_{n+1}$.*

Proof. Suppose that at level n, some requirement is not satisfied. The first one must have type (b), to find an f_n-image for d, since the requirements of other types are easily seen to be satisfiable once the earlier requirements have been satisfied. Having failed to satisfy the requirement of type (b) for the element d, we may also fail to satisfy later requirements of type (a) or (b), but we still satisfy all requirements of type (c). Say that for earlier requirements, we have $f_n(\bar{b}) = \bar{a}$. By the (e)$_k$ requirements, the B_n-type of d over $\bar{b}$ in $\mathcal{A}_n$ is generated by the B_n-type of $\bar{b}$ and the $\exists_n$-formulas in the type of d over $\bar{b}$. Since we know that Δ^0_{n+1} will satisfy all of its type (c) requirements, there will be a realization e of this B_n-type over $\bar{a}$, and we can eventually satisfy the (b)-requirement by sending d to this e. □

We have shown that the structures $\mathcal{A}_n$ built by the different workers are all isomorphic to some fixed structure $\mathcal{A}$.

Lemma 6.4. *The structure $\mathcal{A}$ is a model of T.*

Proof. By Lemma 6.2, every P^n-type assigned to a tuple $\bar{b}$ in $\mathcal{A}_{n+1}$ represents only true statements about the tuple. However, every B_n-sentence $\varphi \in T$ is in every P^n-type. Thus, φ is true in $\mathcal{A}_{n+1}$, so φ is true in $\mathcal{A}$. Since this holds for every n, $\mathcal{A} \models T$. □

Lemma 6.5. *The model $\mathcal{A}$ is the saturated model of T.*

Proof. We show that for each n, $\dim(\mathcal{A}) \geq n$. The tuple $(0, \ldots, n-1)$ is assigned the generic B_{n-1}-type in $\mathcal{A}_n$. For all $k \geq n$, $f_k : \mathcal{A}_k \to \mathcal{A}_{k+1}$ is the identity function on $\{0, \ldots, n-1\}$, and each $\mathcal{A}_k$ assigns the n-tuple $(0, \ldots, n-1)$ to be B_{k-1}-generic. Then by Lemma 6.2, this tuple satisfies the full generic type in $\mathcal{A}_n$. Thus, the tuple has dimension n. □

We have a Δ^0_3 P^2-labeling of $\mathcal{A}_3$, where this is a saturated model of T. By Lemmas 4.4 and 4.6, we get a recursive copy. □

7 Boundedly saturated models of finite dimension

In this section, we consider Case 4, where $\mathcal{M}$ has finite dimension and is boundedly saturated. By Lemmas 4.4 and 4.6, to show that $\mathcal{M}$ has a computable copy, it is enough to prove the following.

Theorem 7.1. *Let T be a strongly minimal theory such that T_{n+2} is Δ^0_n uniformly in n, and suppose $\mathcal{M}$ is a boundedly saturated model of dimension k. Then for some N there is a copy of $\mathcal{M}$ with a Δ^0_N P^{N-1}-labeling.*

We will split the proof into two sub-cases, and we will say, in each sub-case, what is N. For each $n \geq N$, we will produce a structure $\mathcal{A}_n$ with a Δ^0_n P^{n-1}-labeling. We will also produce an isomorphism $f_n : \mathcal{A}_n \to \mathcal{A}_{n+1}$. We will say more later about the complexity of f_n. We fix in advance a tuple $\bar{c}$ of k independent elements. The tuple $\bar{c}$ is known at all levels $n \geq N$. It will form a basis for all $\mathcal{A}_n$, and it will be fixed by all of the isomorphisms f_n. The Δ^0_n worker has access to T_{n+2}, and thus computes the B_{n+1}-type of $\bar{c}$ and enumerates the $\forall_{n+2}$-type of $\bar{c}$, by Lemma 2.6. For each n, $\mathcal{A}_n$ will have infinitely many further B_n-independent elements, but each of these will become algebraic over $\bar{c}$ at some level.

Here is the first sub-case.

Sub-case 4 (a): There is some K such that for every $n \geq K$, there is an $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ that is algebraic and consistent with $\mathrm{tp}(\bar{c})$ and the generic B_{n-1}-type $p(\bar{c}, x)$.

We may suppose that $K > 2$. We produce a copy of $\mathcal{M}$ with a Δ^0_K P^{K-1}-labeling. Thus, we may let $N = K$. The difficulty in the construction is ensuring that each element is algebraic over $\bar{c}$. Towards this aim, for each even $n \geq K$, the pair of workers Δ^0_n and Δ^0_{n+1} will work together. In particular, Δ^0_{n+1} will create an element a, assign it a generic B_n-type over $\bar{c}$ and send it to an element satisfying an algebraic $\exists_{n+2}$-formula in $\mathcal{A}_{n+2}$. The Δ^0_n worker will identify the highest priority element to make algebraic, not satisfying any algebraic B_{n-1}-formula, and will send this element to a, if it does not satisfy an algebraic $\exists_n$-formula.

In the course of the construction, we build the sequence of functions f_n, for $n \geq K$, where f_n is an isomorphism from $\mathcal{A}_n$ to $\mathcal{A}_{n+1}$, such that f_n is Δ^0_{n+1}, uniformly in n. Thus, all of the composite maps $f_{n-1} \circ \ldots \circ f_K$ are Δ^0_n, and the Δ^0_n worker can compute the images in $\mathcal{A}_n$ of elements in $\mathcal{A}_K$. This gives a Δ^0_n ordering $<$ of $\mathcal{A}_n$ induced by the usual

ordering $<_\omega$ on the universe of $\mathcal{A}_K$. We will define f_n using this order, and we will have to argue that each f_n is in fact a bijection, so that the order is well-defined.

By Lemma 2.6, Δ^0_{n-2} can enumerate the $\forall_n$-type of $\bar{c}$. Then Δ^0_{n-2} can enumerate the algebraic $\exists_n$-formulas $\varphi(\bar{c}, x)$. It follows that Δ^0_{n-1} can decide which $\exists_n$-formulas $\varphi(\bar{c}, x)$ are algebraic. By Lemma 3.2, given a P^n-index for a type $p(\bar{c}, x)$, Δ^0_{n-1} can compute the type. Then Δ^0_n can determine whether $p(\bar{c}, x)$ contains an algebraic $\exists_n$-formula. Recall that Δ^0_n assigns B_{n-1}-types to tuples in $\mathcal{A}_n$. Thus, Δ^0_n can easily find the $<$-first element $b \in \mathcal{A}_n$ that does not satisfy any algebraic B_{n-1}-formula.

Note that Δ^0_{n+1} could find the $<$-least element of $\mathcal{A}_n$ for which the $\exists_n$-type over $\bar{c}$ is generic. Of course, the construction is dynamic, and the workers will collaborate to make sure that all elements are eventually algebraic over $\bar{c}$. For each $n \geq K$, Δ^0_n arranges that the element a_n that is right after $\bar{c}$ in (the ω-ordering of) $\mathcal{A}_n$ is assigned the generic B_{n-1}-type. For odd n, the Δ^0_n worker finds an algebraic $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ that is consistent with the generic B_{n-1}-type $p(\bar{c}, x)$ and the generic type of $\bar{c}$. By Lemma 5.3, it is Δ^0_n to identify such a $\varphi_n(\bar{c}, x)$. Say that $\varphi_n(\bar{c}, x) = (\exists \bar{y})\psi(\bar{c}, x, \bar{y})$, where ψ is $\forall_n$. In addition to the other requirements, the Δ^0_n worker has the goal of making the f_n-image of a_n satisfy $\varphi_n(\bar{c}, x)$. For even $n \geq K$, the Δ^0_n worker acts to ensure that if e_n is the $<$-first element of $\mathcal{A}_n$ that does not satisfy an algebraic B_{n-1}-formula, then either it satisfies an algebraic $\exists_n$-formula or else $f_n(e_n) = a_{n+1}$.

Here are the requirements:

(a) Find an f_n-pre-image for some $c \in \mathcal{A}_{n+1}$.

(b) Find an f_n-image for some $d \in \mathcal{A}_n$.

(c) If $r(\bar{x})$ is the B_{n-1}-type assigned to a tuple $\bar{e}$, and $s(\bar{x}, y)$ is a B_{n-1}-type generated by $r(\bar{x})$ and the collection of $\exists_{n-1}$-formulas in $s(\bar{x}, y)$, then realize $s(\bar{e}, y)$.

(d) (If n is odd): Ensure that $f_n(\bar{c}, a_n)$ satisfies $\varphi_n(\bar{c}, x)$.

(e) (If n is even): Ensure that if e_n is the $<$-first element of $\mathcal{A}_n$ whose B_{n-1}-type is generic over $\bar{c}$, then either e_n satisfies some algebraic $\exists_n$-formula $\psi(\bar{c}, x)$ or else $f_n(e_n) = a_{n+1}$.

At level n, the single requirement of type (d) or (e) has highest priority. If n is odd, we give the type (b) requirement for a_n highest priority among the remaining requirements. We start by saying how the (d) requirement is satisfied. Note that it is Δ^0_n to find $\varphi_n(\bar{c}, x)$ and to determine the number of realizations of $\varphi_n(\bar{c}, x)$. For an element of $\mathcal{A}_{n+1}$ to satisfy an $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ would normally be a Σ^0_{n+1} condition, but knowing the number of such elements makes it Δ^0_{n+1}. Thus, in a finite-injury fashion, we can change

the f_n-image of a_n finitely often before finding an image that satisfies $\varphi_n(\bar{c}, x)$. The only remaining issue is that, when we assign B_{n-1}-types, we must verify that the types are consistent with $\varphi_n(\bar{c}, a_n)$ and the generic type of $\bar{c}$. This can be done, by Lemma 5.3.

Now, we describe how (e) requirements are satisfied. Let e_n be the $<$-first element of $\mathcal{A}_n$ that does not satisfy an algebraic B_{n-1}-formula. This element is easily identified by Δ^0_n. We can map it to a_{n+1} unless we have assigned a B_{n-1}-type $q(\bar{c}, x, \bar{y})$ to e_n and a further tuple $\bar{d}$, such that $q(\bar{c}, x, \bar{y})$ is inconsistent with the generic B_n-type $p(\bar{c}, x)$. This would mean that e_n satisfies some algebraic $\exists_n$-formula $\psi(\bar{c}, x)$. Either way, requirement (e) is satisfied.

The other requirements are handled exactly as in previous arguments, modulo the finite injury incurred due to action on behalf of the type (d) or (e) requirements. Again, for a requirement of type (b), we will have a family of "helping requirements" $(e)_k$, so that when the requirement of finding an image for d is injured for the k^{th} time, we make $\bar{b}, d$ satisfy the k^{th} $\exists_n$-formula, if this is consistent.

Lemma 7.2. *For all n, the Δ^0_n worker gives the P^{n-1}-labeling of a structure, and every requirement is satisfied. Therefore, the structures $\mathcal{A}_n$ are all isomorphic. Moreover, each $\mathcal{A}_n$ is a model of T.*

Proof. Note that, on each level, every (c) requirement is satisfied (since it is not subject to injury). Thus, at each level, we do build a structure $\mathcal{A}_n$. As before, this guarantees that the B_{n-1}-type assigned to a given tuple is actually realized by the tuple. Let n be least such that some requirement at level n fails to be satisfied. Once the (d) or (e) requirement and the (b) requirement for a_n are satisfied, the rest is proved as in Lemmas 6.2, 6.3, and 6.4. For odd n, to see that the (d)-requirement is satisfied, recall that $\varphi_n(\bar{c}, x)$ is an algebraic $\exists_{n+1}$-formula, so it has the form $(\exists \bar{y})\psi(\bar{c}, x, \bar{y})$, where ψ is a $\forall_n$-formula. Also note that on each level, the tuple $\bar{c}$ is generic. Using these facts, we can show the following.

Claim: There is a tuple $\bar{e}$ in $\mathcal{A}_{n+1}$ containing all realizations of $\varphi_n(\bar{c}, x)$, as well as witnesses for each realization.

Proof of Claim. Let M be greatest such that $(\exists^{\geq M} x)\varphi_n(\bar{c}, x)$ is in the $\exists_{n+1}$-type of $\bar{c}$. Consider the B_n-type $q(\bar{c}, x_1, \ldots, x_M, \bar{y}_1, \ldots, \bar{y}_M)$ that is generated by the B_n-formulas true of $\bar{c}$, the formulas $\psi(\bar{c}, x_i, \bar{y}_i)$, for $1 \leq i \leq M$, and $x_i \neq x_j$, for $1 \leq i < j \leq M$, and further $\exists_n$-formulas. The type (c) requirements at level $n+1$ guarantee that q is realized by some tuple $\bar{e}$ in $\mathcal{A}_{n+1}$. This is the tuple we want. □

Take $\bar{e}$ as in the Claim. Consider a stage large enough that the P^n-index has settled for the tuple $\bar{e}$ in $\mathcal{A}_{n+1}$. At this stage s, we can assign an image for a_n, and there will be no further injury. To see that the (e) requirement is satisfied at an even level n, note that each f_k, for $k < n$, is total. This is because we are supposing n to be least so that some

requirement at level n fails to be satisfied. Then e_n is well-defined. Once the Δ^0_n worker identifies e_n, we satisfy the (e) requirement. It follows that all $\mathcal{A}_n$ are isomorphic, and all are models of T. □

Lemma 7.3. *For odd n, let $\varphi_n(\bar{c}, x)$ be the special formula found by the Δ^0_n worker. Then $f_n(a_n)$ satisfies $\varphi_n(\bar{c}, x)$. Thus, a_n is in the $\exists_{n+1}$-algebraic closure of $\bar{c}$.*

Proof. This is immediate from the success of requirement (d) and Lemma 7.2. □

Lemma 7.4. *Let d be any element in $\mathcal{A}_K$. Then $d \in \mathrm{acl}(\bar{c})$.*

Proof. Suppose, toward a contradiction, that d is the least element of $\mathcal{A}_K$ that is not in $\mathrm{acl}(\bar{c})$. Let n be even and sufficiently large that each constant less than d is mapped by the composite function $f_{n-1} \circ \ldots \circ f_{K+1} \circ f_K$ to an element of $\mathrm{acl}_{B_n}(\bar{c})$. Then the image of d in $\mathcal{A}_n$ is e_n. Now, either d satisfies an algebraic $\exists_n$-formula $\varphi(\bar{c}, x)$ or else $f_n(e_n) = a_{n+1}$. By the previous lemma, a_{n+1} is algebraic over $\bar{c}$. Therefore, d is algebraic over $\bar{c}$. □

In Sub-case 4 (a), we assumed that for all $n \geq K$, there is an $\exists_{n+1}$-formula that can be used to make an element algebraic. We showed that the structure $\mathcal{A}_K$ is a model of T with a generic tuple $\bar{c}$ such that every $d \in \mathcal{A}_K$ is algebraic over $\bar{c}$. This $\mathcal{A}_K$ is a copy of $\mathcal{M}$ with a Δ^0_K P^{K-1}-labeling. We now turn to the other sub-case.

Sub-case 4 (b): For infinitely many n, there is no $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ that is algebraic and consistent with $\mathrm{tp}(\bar{c})$ and the generic B_{n-1}-type for $\bar{c}, x$.

In this sub-case, we will show that there is a copy of $\mathcal{M}$ with a Δ^0_3 P^2-labeling. (Thus, we have the conclusion of Theorem 7.1 with $N = 3$.) The construction will have one key feature that did not appear in the constructions in earlier sections or in Sub-case 4 (a). Earlier, the Δ^0_n worker constructed an isomorphism f_n via a finite injury construction, so f_n was Δ^0_{n+1}, uniformly in n. In Sub-case 4 (b), f_n will be constructed via an infinite injury construction. As a consequence, we cannot say that f_n is Δ^0_{n+1} *uniformly*, although the argument will show that each f_n is Δ^0_{n+1}.

In general, when we begin the construction of $\mathcal{A}_n$ using a Δ^0_n oracle, we will not know whether there is a suitable $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$, algebraic and consistent with $\mathrm{tp}(\bar{c})$ and the generic B_{n-1}-type for $\bar{c}, x$. Thus, we cannot begin by finding it. Instead, we search for such a $\varphi_n(\bar{c}, x)$, and we begin the construction of $\mathcal{A}_n$ and f_n, assuming that no such φ_n exists. Of course, if we find an appropriate $\exists_{n+1}$-formula φ_n after we have already committed to a formula that implies $\neg\varphi_n(\bar{c}, a)$, then we cannot use $\varphi_n(\bar{c}, x)$ to make the element a algebraic over $\bar{c}$. Instead, we will make some new element a' algebraic via φ_n, and we hope that the lower worker will succeed in mapping the first element that needs to be made algebraic to this new a'.

In the course of the construction, we build a sequence of functions f_n each of which is Δ^0_{n+1}, whose indices are uniformly Δ^0_{n+2} for $n > 2$. The maps $f_3, \ldots, f_{n-1}$ are all known by Δ^0_{n+1}. Thus, for each $\mathcal{A}_n$, we can consider the isomorphism $F := f_{n-1} \circ \cdots \circ f_3$ from $\mathcal{A}_3$ to $\mathcal{A}_n$. We consider $\mathcal{A}_n$ to be ordered by $<$, where this is the image under F of the usual order $<_\omega$ on ω (the universe of $\mathcal{A}_3$). Recall that Δ^0_n can determine whether the B_n-type $p(\bar{c}, x)$ with a given index is algebraic, and can check whether a given $\exists_{n+1}$-formula $\varphi(\bar{c}, x)$ is algebraic. As in Subcase 4 (a), Δ^0_n assigns B_{n-1}-types. Once Δ^0_n approximates sufficiently much of the isomorphisms $f_3, \ldots, f_{n-1}$ (in particular, f_{n-1}), Δ^0_n can locate the $<$-least element of B_n that does not satisfy any algebraic B_{n-1}-formula $\psi(\bar{c}, x)$. We call this element e_n.

We define inductively the "purpose" of each level n. There are three possible purposes: A, B or C. The bottom level is an A level. If level n is an A level that does not find a formula φ_n, then level $n+1$ is a B level if and only if there is an $\exists_{n+2}$-formula $\varphi_{n+1}(\bar{c}, x)$ that is algebraic and consistent with $\mathrm{tp}(\bar{c})$ and the generic B_n-type $p(\bar{c}, x)$. Otherwise, level $n+1$ is also an A level. If level n is a B level, then level $n+1$ is automatically a C level. If level n is a C level, then level $n+1$ is automatically an A level. The idea is that pairs where n is an A level and $n+1$ is a B level will ensure that one more element becomes algebraic. The purpose of C levels is to allow one level where nothing special happens, which will guarantee enough saturation of the appropriate structures to ensure that B levels succeed.

Throughout the construction, if level $n-1$ is an A level that fails to find a formula φ_{n-1}, the Δ^0_n worker will search for an $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ that is algebraic and consistent with the B_{n+1}-type of $\bar{c}$ and the generic B_{n-1}-type $p(\bar{c}, x)$. Note that for φ_n to have these properties is Δ^0_n, by Lemma 5.3. If such a φ_n is found, then the Δ^0_n-worker will re-start the task of building f_n, canceling $f_n(y)$ for every element $y \notin \bar{c}$. In other words, the Δ^0_n worker realizes that he is a B level, and from then on, he can work with that knowledge.

The Δ^0_n worker will add a new element a_n, assign to $\bar{c}, a_n$ the generic B_{n-1}-type $p(\bar{c}, x)$, and begin building f_n again. Now, defining $f_n(a_n)$, and ensuring that it satisfies $\varphi_n(\bar{c}, x)$, has the top priority. The element a_n is labeled "algebraizable". In this event, Δ^0_n, knowing now that he is a B level worker and not an A-level worker, switches from attempting to satisfy one requirement (called (f) on the list below) to a different requirement (called (e) below). In particular, it is the role of a Δ^0_n B level worker to label an element a_n algebraizable and ensure that a_n is mapped to an element in $\mathcal{A}_{n+1}$ that is algebraic over $\bar{c}$. The task of the Δ^0_{n-1} worker below, an A level worker, is to ensure that the first element that does not satisfy an algebraic B_{n-2}-formula either satisfies an algebraic $\exists_n$-formula $\psi(\bar{c}, x)$ or is mapped to the element a_n labeled algebraizable.

Note that what we have described above, finding the formula $\varphi_n(\bar{c}, x)$ and starting over on building f_n, happens at most once. The requirements are as follows:

(a) Find an f_n-pre-image for some $c \in \mathcal{A}_{n+1}$.

(b) Find an f_n-image for some $d \in \mathcal{A}_n$.

(c)$_k$ If the type (b) requirement to define $f(d)$ is injured for the k^{th} time, and for the current B_n-type $p(\bar{u})$ and B_{n-1}-type $q(\bar{u}, x, \bar{v})$, $p(\bar{u}) \cup q(\bar{u}, x, \bar{v})$ is consistent with the k^{th} $\exists_n$-formula $\psi_k(\bar{u}, x)$ (in variables $\bar{u}, x$), then add a witness $\bar{d}'$ to make $\psi_k(\bar{b}, d)$ true in $\mathcal{A}_n$.

(d) If the tuple $\bar{e}$ is assigned the B_{n-1}-type $r(\bar{x})$, and $s(\bar{x}, y)$ is a type generated by $r(\bar{x})$ and the collection of $\exists_{n-1}$-formulas in $s(\bar{x}, y)$, then realize $s(\bar{e}, y)$.

The following two requirements depend on the nature of the layer n.

(e) If level n is a B level, then create a new element a_n, label it algebraizable, and ensure that $f_n(a_n)$ satisfies the algebraic $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$.

(f) Suppose that level n is an A level and that there is no $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ that is algebraic and consistent with the generic B_{n-1}-type $q(\bar{c}, x)$ and the B_{n+1}-type of $\bar{c}$. Let e_n be the $<$-first element of $\mathcal{A}_n$ that does not satisfy any algebraic B_{n-1}-formula. If some element a_{n+1} of $\mathcal{A}_{n+1}$ is labeled algebraizable, then ensure that either e_n satisfies some algebraic $\exists_n$-formula or else $f_n(e_n) = a_{n+1}$.

If n is a C layer, then there is no (e) or (f) requirement.

The priorities are as follows: (e) has highest priority, (f) has second highest priority, and (a)-(d) are all of lower priority with the (b)-requirement for e_n having highest priority among these. The actions for requirements (a)-(d) are essentially as in the saturated case. Note that if level $(n-1)$ is an A level, it is not clear whether the Δ^0_n worker should satisfy requirement (e) or (f). Being a B level is then a $\Sigma^0_1(\Delta^0_n)$ event, so, if a special formula φ_n is found, then the (e) requirement combines with the (a)-(d) requirements exactly as in Sub-case 4 (a), and they all succeed. The main difficulty will be to show that despite injury from the (f) requirement, the requirements of types (a) and (b) can still be satisfied.

Note first that if, in attempting to satisfy the (f)-requirement, we find a formula φ_n, then the (f)-requirement is permanently satisfied. This removes any restraints for the requirement and injures all lower-priority requirements (i.e., f_n becomes undefined for all elements except for $\bar{c}$). This can happen at most once. The Δ^0_{n+1} worker searches for an $\exists_{n+2}$ algebraic formula φ_{n+1}, and upon finding one, enumerates a single pair (φ_{n+1}, a_{n+1}). Thus, the pair (φ_{n+1}, a_{n+1}) where φ_{n+1} is the formula found, and a_{n+1} is the labeled element, forms a $\Sigma^0_2(\Delta^0_n)$ singleton. So, the Δ^0_n worker can use Σ^0_2-approximation to approximate the element a_{n+1}. This means that at each stage, the Δ^0_n worker has a guess as

to the identity of a_{n+1}, which is allowed to be "undefined". If, in fact a_{n+1} is defined, then from some stage onwards, the Δ^0_n worker's guess will be correct. If, on the other hand, the Δ^0_{n+1} worker never defines an element to be a_{n+1}, then the Δ^0_n worker will infinitely often guess "undefined", but may also infinitely often guess various elements of $\mathcal{A}_{n+1}$.

As a consequence, f_n will be defined by an infinite injury process. That is, as stages go by, f_n may send an element x to infinitely many different images based on different guesses as to the identity of a_{n+1}, but then the correct image is the one based on the guess that a_{n+1} is "undefined". As discussed above, after finite injury, the Δ^0_n worker can find e_n, where this is the $<$-first element of $\mathcal{A}_n$ not satisfying any algebraic B_{n-1}-formula $\psi(\bar{c}, x)$. We can map e_n to the special element a_{n+1} unless it satisfies some algebraic $\exists_n$-formula, witnessed in the course of assigning B_{n-1}-types.

An "(f)-true stage" is a stage at which the approximation correctly determines whether φ_n exists, and, in addition, either identifies the element a_{n+1} labeled as algebraizable, or correctly guesses that a_{n+1} is "undefined"; i.e., that the Δ^0_{n+1}-worker never defines an element as a_{n+1}. A more precise formulation of the type (b) requirement for d says that for all but finitely many (f)-true stages, d is in the domain of f_n and the value of $f_n(d)$ is the same on all of these stages. The (a)-requirements are made precise in the same way. Then our function f_n will be the limit of the Δ^0_n approximations to f_n along the (f)-true stages.

The main difficulty in this construction is as follows. Having made progress towards building f_n, we have a tuple $e_n, \bar{b}$ on which we have defined f_n, say $f_n(e_n, \bar{b}) = e', \bar{b}'$, and we have a further tuple $\bar{d}$ on which we have not defined f_n. Later, via our $\Sigma^0_2(\Delta^0_n)$-approximation, we come to believe that a_{n+1} is defined and is equal to k. We now attempt to make $f_n(e_n) = k$, if e_n has not become algebraic over $\bar{c}$ via some $\exists_n$-formula, witnessed by our assignment of a B_{n-1}-type to $\bar{c}, e_n, \bar{b}, \bar{d}$. While believing that k is labeled algebraizable, we need to continue building f_n. We may later stop believing that k was labeled algebraizable, and in this case, we want to be able to make f_n return to $f_n(e_n) = e'$ and $f_n(\bar{b}) = \bar{b}'$.

It appears that the actions made to extend the construction at stages where we assign $f_n(e_n) = k$ may make it impossible to revert to the earlier f_n with $f_n(e_n, \bar{b}) = e', \bar{b}'$. There would be no problem if, in fact, some element is labeled algebraizable, for then this would be a standard finite-injury phenomenon. In particular, once the approximation settles down on the correct a_{n+1}, the construction would continue with no further injury from the (f)-requirement. However, in the case where no element is labeled algebraizable by the Δ^0_{n+1} worker, we need to ensure that each later requirement of type (a) or (b) can succeed on the (f)-true stages.

It will turn out that our $(c)_k$-requirements will suffice to ensure that we satisfy (b)-requirements, but in order to ensure we satisfy (a)-requirements, we will need to protect

the image of f_n. For some tuples $a, \bar{e}$ in $\mathcal{A}_{n+1}$, we will engage in the process explained below to "certify" this tuple. If we do so, then we will attempt to preserve $f_n(e_n, \bar{b}) = a, \bar{e}$ on the (f)-true stages. That is, when we change f_n in order to send e_n to our current guess d at a_{n+1}, we will ensure that the B_n-type of $f_n(e_n, \bar{b})$ is the same as the B_n type of $a, \bar{e}$. That is, we look for a tuple $\bar{d}$ so that $d, \bar{d}$ is assigned the same B_n-type to be the new image of $e_n, \bar{b}$. If we set $f_n(e_n, \bar{b}) = d, \bar{d}$, then any of our later B_{n-1}-commitments will still be consistent with the B_n-type of $a, \bar{e}$, since when we make the extension, we check consistency with the B_n-type of the current image and these images have the same B_n-type. We now describe the process of certifying a tuple $a, \bar{e}$.

Lemma 7.5. *Let $p(\bar{c}, a, \bar{e})$ be a given guess at the B_n-type of $\bar{c}, a, \bar{e}$. For a sub-tuple $\bar{g}$ of $\bar{e}$, it is Δ^0_n to check if*

$$\{(\exists \bar{y})\psi(\bar{c}, e_n, \bar{g}, \bar{y}) \mid \psi \in p(\bar{c}, a, \bar{e}))\} \textit{ is contained in the generic } \exists_{n+1}\textit{-type.}$$

Proof. Since $\{(\exists \bar{y})\psi(\bar{c}, e_n, \bar{g}, \bar{y}) \mid \psi \in p(\bar{c}, a, \bar{e}))\}$ is a subset of the generic $\exists_{n+1}$-type if and only if

$$(\forall \psi \in p(\bar{c}, a, \bar{e}))\ ((\exists \bar{y})\, \psi(\bar{c}, e_n, \bar{g}, \bar{y}) \text{ is in the generic } \exists_{n+1}\text{-type})\,,$$

and by Lemma 2.6, the generic $\exists_{n+1}$-type is $\Pi^0_1(T_{n+1})$, the whole condition is Π^0_1 relative to $T_{n+1} \oplus p(\bar{c}, a, \bar{e}))$. Thus, it is Π^0_{n-1} to verify that

$$\{(\exists \bar{y})\psi(\bar{c}, e_n, \bar{g}, \bar{y}) \mid \psi \in p(\bar{c}, a, \bar{e})\}$$

is contained in the generic $\exists_{n+1}$-type. □

If the empty tuple $\bar{g}$ does not satisfy this condition, then we will not certify the tuple $a, \bar{e}$. If the empty tuple satisfies the condition, then we find a maximal $\bar{g} \subseteq \bar{e}$ that satisfies the condition. Let $\bar{k}$ be $\bar{b} \smallsetminus \bar{g}$. Having the partition of $\bar{b}$ into $\bar{g}, \bar{k}$, we find, for each $k_i \in \bar{k}$, and for $\bar{k}^*_i = \bar{k} - k_i$, a $\forall_n$-formula $\psi_i(\bar{c}, e_n, \bar{g}, y, \bar{z})$ such that $\psi(\bar{c}, a, \bar{g}, k_i, \bar{k}^*_i)$ is in $p(\bar{c}, a, \bar{g}, \bar{k})$ and $(\exists \bar{z})\psi_i(\bar{c}, e_n, \bar{g}, y, \bar{z})$ is not in the generic $\exists_{n+1}$-type. (These formulas witness the maximality of $\bar{g}$.)

Let $\Psi = \wedge_{k_i \in \bar{k}} \psi_i(\bar{c}, e_n, \bar{g}, k_i, \bar{k}^*_i)$. Say that $\bar{c}, a, \bar{g}$ has length m and let $\bar{h}$ be a generic tuple of length m. Over $\bar{h}$, there are only finitely many tuples $\bar{z}$ satisfying $\Psi(\bar{h}, \bar{z})$. Let K be the number of such tuples. Note that

$$\chi(\bar{c}, x) := \neg(\exists^\infty u_1)(\exists^\infty u_2) \ldots (\exists^\infty u_m)(\exists^{\geq K} \bar{z})\Psi(\bar{c}, x, \bar{u}, \bar{y})$$

is equivalent to a $\forall_{n+1}$-formula, and by Lemma 2.1, it is Δ^0_n to figure out how to replace $(\exists^\infty u_i)$ here by $(\exists^{\geq l_i} u_i)$ for an appropriate l_i. Since for a generic tuple $\bar{h}$, there are K realizations of $\Psi(\bar{h}, \bar{z})$, if $\chi(\bar{c}, a)$ were true, then a would be algebraic over $\bar{c}$ via a $\forall_{n+1}$-formula.

The condition

$$\mathcal{Q}_{a,\bar{e}} := (\forall \psi(\bar{c}, x) \in \text{the generic } B_n\text{-type})((\exists x)(\chi(\bar{c}, x) \wedge \psi(\bar{c}, x)) \in \text{tp}_{\exists_{n+2}}(\bar{c}))$$

is Π^0_1 over a positive instance of $\text{tp}_{\exists_{n+2}}(\bar{c})$ along with the generic B_n-type, which is Π^0_n, so condition $\mathcal{Q}_{a,\bar{e}}$ is Π^0_n. Condition $\mathcal{Q}_{a,\bar{e}}$ declares that it is possible to have the generic B_n-type over $\bar{c}$ and also be algebraic via χ. In particular, if condition $\mathcal{Q}_{a,\bar{e}}$ is true, then the Δ^0_{n+1} worker can use χ as φ_{n+1}, and we know that the Δ^0_{n+1} worker will identify some element as a_{n+1}. When the Δ^0_n-worker sees $\neg\mathcal{Q}_{a,\bar{e}}$, then we say it certifies the tuple $a, \bar{e}$. If the approximation to the B_n-type assigned to $a, \bar{e}$ changes, then the certification is removed.

Now, we describe how this certification determines how we define f_n when Δ^0_n changes its guess about the identity of a_{n+1} from "undefined" to d for some d. At this moment, we have defined $f_n(\bar{c}, e_n, \bar{b}) = \bar{c}, a, \bar{e}$. The first thing we do is choose a maximal initial sub-tuple $\bar{e}_0$ of $\bar{e}$ so that $a, \bar{e}_0$ has been certified. We un-define f on some values of $\bar{b}$, if necessary, so that we have $\bar{e}_0 = \bar{e}$. Now, we need to re-define f_n so that $f_n(e_n) = d$ instead of $f_n(e_n) = a$. Before doing so, we insist on finding a tuple $\bar{d}$ so that the B_n-type assigned to $\bar{c}, d, \bar{d}$ is the same as that assigned to $\bar{c}, a, \bar{e}$. Until we find such a $\bar{d}$, we continue the construction with $f_n(\bar{c}, e_n, \bar{b}) = f_n(\bar{c}, a, \bar{e})$, ignoring the new approximation that $a_{n+1} = d$. If we find a $\bar{d}$ as needed, then we re-define $f_n(\bar{c}, e_n, \bar{b}) = \bar{c}, d, \bar{d}$. Note that if the approximation to the B_n-type of $\bar{c}, d, \bar{d}$ changes, we revert to $f_n(\bar{c}, e_n, \bar{b}) = f_n(\bar{c}, a, \bar{e})$ until we find a new $\bar{d}$ with the correct type. This is the privilege that $a, \bar{e}$ has gained by being certified. We must argue below that if in fact $a_{n+1} = d$, then a tuple $\bar{d}$ as needed will be found.

The $(c)_k$-actions are exactly as usual on stages where (f) sees no element labeled algebraizable. In particular, if s is the k^{th} stage where (f) sees no element labeled algebraizable, and $f_n(b)$ is undefined, although $f_n(b)$ was defined at the previous stage where (f) saw no element labeled algebraizable, then we attempt to make the k^{th} $\exists_n$-formula true of b over the higher priority tuple.

Similarly, every time Δ^0_n guesses some element d is a_{n+1}, and f_n is changed accordingly, we have a $(c)_k$-strategy which works assuming that $f_n(e_n) = d$. In particular, on the k^{th} stage where $f_n(b)$ is undefined and $f_n(e_n) = d$, we attempt to make the k^{th} $\exists_n$-formula true of b over the higher-priority tuple.

Lemma 7.6. *Every requirement succeeds. Thus, for all n, $\mathcal{A}_n$ is a structure with a Δ^0_n P^{n-1}-labeling and $\mathcal{A}_n$ is isomorphic to $\mathcal{A}_{n+1}$.*

Proof. If level n is a B or C level, we need only verify the success of requirements (a)-(e) or (a)-(d). The argument is exactly as in Lemma 7.2 of Sub-case 4 (a). So, we suppose that n is an A level, and that all requirements at levels below n are satisfied, and we

show that all requirements at level n are satisfied. We must show that the (a)-(d) and (f)-requirements all succeed. If the Δ^0_n worker finds an algebraic $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$, then the (f)-requirement is fulfilled in that way at a finite stage, and the other requirements are satisfied as in Lemma 7.2 of Sub-case 4 (a).

Suppose there is no algebraic $\exists_{n+1}$-formula $\varphi_n(\bar{c}, x)$ consistent with the generic B_{n-1}-type over $\bar{c}$. We consider the (f) requirement at level n. First, suppose that Δ^0_n correctly believes an element d is labeled algebraizable by the Δ^0_{n+1} worker. Recall that e_n is the $<$-first element of $\mathcal{A}_n$ not satisfying any algebraic B_{n-1}-formula. The (f)-requirement is to map e_n to d unless e_n satisfies some algebraic $\exists_n$-formula. The difficulty is that some tuple $a, \bar{e}$ may have been certified, where $f_n(\bar{c}, e_n, \bar{b}) = \bar{c}, a, \bar{e}$. Then we insist that the full B_n-type of $\bar{c}, e_n, \bar{b}$ is preserved by f_n. Thus, we must argue that there is a way to map e_n to d while preserving this full B_n-type. Since d is labeled algebraizable, the $(n+1)$-level is a B level and the $(n+2)$-level is a C-level. Thus, all requirements succeed on levels $n+1$ and $n+2$. It follows that $\mathcal{A}_{n+1} \cong \mathcal{A}_{n+3}$.

The (d)-requirements for $\mathcal{A}_{n+3}$ are satisfied, and Lemma 1.7 implies that $\mathcal{A}_{n+1}$ realizes all consistent B_{n+1}-types. Recall our partition $\bar{e} = \bar{g}, \bar{k}$. Let $\bar{h}$ be a tuple in $\mathcal{A}_{n+1}$, of the same length as $\bar{g}$, such that $\bar{h}$ is B_{n+1}-generic over $\bar{c}, d$. By $\neg\mathcal{Q}_{a,\bar{e}}$, there are at least K realizations $\bar{z}$ of Ψ over $\bar{c}, d, \bar{h}$. If there were infinitely many $\bar{z}$ satisfying $\Psi(\bar{c}, d, \bar{h}, \bar{z})$, then d would satisfy

$$(\exists^\infty u_1) \dots (\exists^\infty u_m)(\exists^\infty \bar{z})\Psi(\bar{c}, x, \bar{u}, \bar{z}).$$

This would be equivalent to an algebraic $\exists_{n+1}$-formula $\varphi(\bar{c}, x)$ that is consistent with the generic B_n-type, contradicting our assumption that no $\varphi_n(\bar{c}, x)$ is found. We consider two cases.

Case 1: Suppose that for some $\bar{z}$ for which $\Psi(\bar{c}, d, \bar{h}, \bar{z})$ holds in $\mathcal{A}_{n+1}$, the B_n-type of $d, \bar{h}, \bar{z}$ over $\bar{c}$ is the same as that of $a, \bar{g}, \bar{k}$. In this case, we may take $d, \bar{h}, \bar{z}$ as our next f_n-image for $e_n, \bar{b}$.

Case 2: Suppose that for all $\bar{z}$ satisfying $\Psi(\bar{c}, d, \bar{h}, \bar{z})$ in $\mathcal{A}_{n+1}$, the B_n-type of $d, \bar{h}, \bar{z}$ over $\bar{c}$ is *not* the same as that of $a, \bar{g}, \bar{k}$. Since there are only finitely many $\bar{z}$ such that $\Psi(\bar{c}, d, \bar{h}, \bar{z})$ holds, we can let $\rho(\bar{c}, x, \bar{u}, \bar{z})$ be a B_n formula that is satisfied by $d, \bar{h}, \bar{z}$ for all of these $\bar{z}$, and is not satisfied by $a, \bar{g}, \bar{k}$. Since the $\exists_{n+1}$-type of $\bar{c}, a, \bar{g}$ is contained in the generic $\exists_{n+1}$-type, a generic tuple must satisfy $(\exists\bar{z})(\Psi(\bar{c}, x, \bar{u}, \bar{z}) \wedge \neg\rho(\bar{c}, x, \bar{u}, \bar{z}))$. However, $\bar{c}, d, \bar{h}$ does not satisfy this formula.

Since $a, \bar{e}$ was certified, we know $\neg\mathcal{Q}_{a,\bar{e}}$. Thus, we have $(\exists^{\geq K}\bar{z})\Psi(\bar{c}, d, \bar{h}, \bar{z})$. Since none of the tuples $d, \bar{h}, \bar{z}$ satisfy $\rho(\bar{c}, x, \bar{u}, \bar{z})$, we see that:

$$\alpha(\bar{c}, d) := (\exists^\infty u_1) \dots (\exists^\infty u_m)(\exists^K \bar{z})(\Psi(\bar{c}, d, \bar{u}, \bar{z}) \wedge \rho(\bar{c}, d, \bar{u}, \bar{z}))$$

is true in $\mathcal{A}_{n+1}$. But for a generic tuple $x, \bar{u}$, we know that there are exactly K realizations of $\Psi(\bar{c}, x, \bar{u}, \bar{z})$ and one of them satisfies $\neg\rho(\bar{c}, x, \bar{u}, \bar{z})$. Thus $\alpha(\bar{c}, d)$ is an algebraic $\exists_{n+1}$-formula that is consistent with the generic B_n-type over $\bar{c}$, since d is labeled algebraizable, so $\bar{c}, d$ is given the generic B_n-type. Thus, there would be a special formula φ_n found on level n.

We have seen that if there is no special formula φ_n, and the f_n-image of $e_n, \bar{b}$ is certified, then it is possible to change the f_n-image of $e_n, \bar{b}$, when we guess that a_{n+1} is defined, in such a way that we can return to the old f_n-image if we later believe that a_{n+1} is not defined. If there actually is an element $a_{n+1} \in \mathcal{A}_{n+1}$ that Δ^0_{n+1} has labeled algebraizable, then, after finitely many steps, Δ^0_n will see this. In this case, after we assign $f_n(\bar{c}, e_n) = \bar{c}, d$, the other level n requirements are satisfied just as in the saturated case.

It remains to see that (a) and (b) requirements can succeed if Δ^0_n, working on requirement (f), infinitely often falsely believes that some element of $\mathcal{A}_{n+1}$ is labeled algebraizable. Suppose the first requirement to fail is a (b) requirement, finding an f_n-image for d. There are infinitely many stages where the Δ^0_n-worker correctly guesses that no element is labeled algebraizable on level $n+1$. Moreover, all of the $(c)_k$ requirements associated with d will act on these (f)-true stages. It follows that the B_n-type of d over the previous constants is generated by the $\exists_n$-formulas. Let $p(\bar{c}, \bar{b}, d)$ be this B_n-type. By minimality of n, the functions f_k are defined for all $k < n$, so Δ^0_n eventually knows the element e_n.

Consider an (f)-true stage s late enough that e_n is known (so, by its higher priority, $e_n \in \bar{b}$ or $e_n = d$), $f_n(\bar{c}, \bar{b})$ will not change on any later (f)-true stage, and the P^{n+1}-indices in $\mathcal{A}_{n+1}$ have settled down on a tuple large enough to contain a realization m of $p(\bar{x}, \bar{y}, z)$ over $f_n(\bar{c}, \bar{b})$. Suppose further that s is large enough that for no element d' before m does $f_n(\bar{c}, \bar{b}), d'$ satisfy the fragment of the $\exists_n$-type realized already by $\bar{c}, \bar{b}, d$. Then at any (f)-true stage $s' > s$, f_n will send d to m. This means that the (b) requirement is satisfied after all.

Suppose that the first requirement to fail is an (a) requirement, to find a pre-image for c. Let $\bar{b}$ be the tuple of constants $\leq c$ and the part of $ran(f_n)$ determined for higher priority (b)-requirements. Let a be the image of e_n on all large enough (f)-true stages. Let $\bar{c}, a, \bar{e}$ be the initial segment of ω that contains $\bar{c}, a$, and $\bar{b}$. Then at some stage s, the approximation to the B_n-type of $\bar{c}, a, \bar{e}$ and its sub-tuples will settle. Further, $a, \bar{e}$ will be certified. To see this, first note that the $\exists_{n+1}$-type of $\bar{c}, a$ is contained in the generic $\exists_{n+1}$-type. Otherwise, a satisfies an algebraic $\exists_{n+1}$-formula that is consistent with the generic B_{n-1}-type, so a formula φ_n will be found, contrary to assumption. If $\mathcal{Q}_{a,\bar{e}}$ were true, this would imply that there is an element a_{n+1} labeled algebraizable, and we are considering the case where there is no such element. Thus $\neg\mathcal{Q}_{a,\bar{e}}$ must hold, so we will certify $a, \bar{e}$ at some stage $s' > s$. Let $t > s'$ be the next (f)-true stage at which we determine an f_n-pre-image for c, say $f_n(\bar{d}) = \bar{c}, a, \bar{e}$. Then at every non-(f)-true stage, we insist on keeping the

same B_n-type of $f_n(\bar{d})$ so that we can return to $f_n(\bar{d}) = \bar{c}, a, \bar{e}$ at every later (f)-true stage. Thus, the requirement (a) is satisfied. □

Lemma 7.7. *If the Δ^0_n worker is a B level worker, and this worker labels an element a_n algebraizable with a formula $\varphi_n(\bar{c}, x)$, then a_n is algebraic over $\bar{c}$ via $\varphi_n(\bar{c}, x)$.*

Proof. This follows from the satisfaction of the (e) requirement for B level workers and the success of the other requirements ensuring that $\mathcal{A}_n \cong \mathcal{A}_{n+1}$. □

Lemma 7.8. *Let e_n be the $<$-least element of $\mathcal{A}_n$ that does not satisfy any algebraic B_{n-1}-formula. Then $e_n \in \mathrm{acl}(\bar{c})$.*

Proof. Suppose, towards a contradiction, that $e_n \notin \mathrm{acl}(\bar{c})$. Let $m > n$ be an A level. Since we are in Sub-case 4 (b), let $k \geq m$ be least so that no φ_k is found. This k is also an A level. Let $l + 1 > k$ be the least B level; thus, level l is an A level that finds no formula φ_l. Since e_l is the image of e_n in $\mathcal{A}_l$ under the composition of $f_n, \ldots f_{l-1}$, the (f)-requirement ensures that either $e_l \in \mathrm{acl}(\bar{c})$ or the f_l-image of e_l in $\mathcal{A}_{l+1}$ is labeled as algebraizable. Then by Lemma 7.7, e_l satisfies an algebraic formula over $\bar{c}$. □

Lemma 7.9. *Every element of $\mathcal{A}_3$ is algebraic over $\bar{c}$.*

Proof. Suppose, toward a contradiction, that d is the $<$-least element not in $\mathrm{acl}(\bar{c})$. Let n be large enough that each element $<$-before d is in $\mathrm{acl}_{B_{n-1}}(\bar{c})$. Then the image of d under the composition of $f_3, ..., f_{n-1}$ is e_n. By Lemma 7.8, d is in $\mathrm{acl}(\bar{c})$ after all. □

Thus, $\mathcal{A}_3$ is a model of T in which $\bar{c}$ satisfies the generic type (Lemma 7.6) and every element is algebraic over $\bar{c}$. Therefore, it is a copy of $\mathcal{M}$.

Acknowledgments. The first author's research was partially supported by NSF grant DMS-1201338 and by NSF Grant No. 0932078000 while he was in residence at the Mathematical Sciences Research Institute in Berkeley, California, during the Spring 2014 semester.

References

[1] U. Andrews, "The degrees of categorical theories with recursive models", *Proc. Amer. Math. Soc.*, vol. 141(2013), pp. 2501–2514.

[2] U. Andrews, "A new spectrum of recursive models using an amalgamation construction", *J. Symbolic Logic*, vol. 76(2011), pp. 883–896.

[3] U. Andrews, "New spectra of strongly minimal theories in finite languages", *Ann. Pure Appl. Logic*, vol. 162(2011), pp. 367–372.

[4] U. Andrews and J. F. Knight, "Presenting non-1-saturated strongly minimal structures", in preparation.

[5] C. J. Ash and J. F. Knight, *Computable Structures and the Hyperarithmetical Hierarchy*, Elsevier, 2000.

[6] E. Fokina, "Arithmetic Turing degrees and categorical theories of computable models", *Mathematical logic in Asia*, World Sci. Publ., Hackensack, NJ, 2006, pp. 58–69.

[7] S. S. Goncharov, "Strong constructivizability of homogeneous models", *Algebra and Logic*, vol. 17(1978), pp. 363–388.

[8] S. S. Goncharov, V. S. Harizanov, M. C. Laskowski, S. Lempp, and C. McCoy, "Trivial, strongly minimal theoreis are model complete after naming constants", *Proc. Amer. Math. Soc*, vol. 131(2003), pp. 3901–3912.

[9] S. S. Goncharov and B. Khoussainov, "Complexity of categorical theories with computable models", *Algebra and Logic*, vol. 43(2004), pp. 650–665.

[10] L. Harrington, "Recursively presentable prime models", *J. Symb. Logic*, vol. 39(1974), pp. 305–309.

[11] E. Hrushovski, "A new strongly minimal set", *Ann. Pure Appl. Logic*, vol. 62(1993), pp. 147–166

[12] N. G. Khisamiev, "On strongly constructive models of decidable theories", *Izvestiya Akademii Nauk Kazakhskoĭ, SSR. Seriya Fiziko-Matematicheskaya*, (1974), pp. 83–84, 94.

[13] B. Khoussainov, M. C. Laskowski, S. Lempp, D. R. Solomon, "On the computability-theoretic complexity of trivial, strongly minimal models", *Proc. Amer. Math. Soc.*, vol. 135(2007), pp. 3711–3721

[14] B. Khoussainov and A. Montalbán, "A computable $\aleph_0$-categorical structure whose theory computes True Arithmetic", *J. Symb. Logic*, vol. 75(2010), pp. 728–740.

[15] J. F. Knight, "Degrees of models with prescribed Scott set", in *Classification: Proc. of Joint U.S.-Israel Workshop*, ed. by Baldwin, 1987, pp. 182–191.

[16] J. F. Knight, "A metatheorem for constructions by finitely many workers", *J. Symb. Logic*, vol. 55(1990), pp. 787–804.

[17] J. F. Knight, "Constructions by transfinitely many workers", *Annals of Pure and Appl. Logic*, vol. 46(1990), pp. 211–234.

[18] J. F. Knight, "Non-arithmetical $\aleph_0$-categorical theories with recursive models", *J. Symb. Logic*, vol. 59(1994), pp. 106–112.

[19] M. Lerman, *A Framework for Priority Arguments*, ASL, 2010.

[20] M. Lerman and J. H. Schmerl, "Theories with recursive models", *J. Symb. Logic*, vol. 44(1979), pp. 59–76.

[21] A. Montalbán, "Priority arguments via true stages", *J. Symb. Logic*, vol. 79(2014), pp. 1315–1355.

[22] M. G. Peretyat'kin, "A criterion of strong constructivizability of a homogeneous model", *Algebra and Logic*, vol. 17(1978), pp. 290–301.