Almost Weakly 2-Generic Sets*

Stephen A. Fenner†
Computer Science Department
University of Southern Maine

August 12, 1996

Abstract

There is a family of questions in relativized complexity theory—weak analogs of the Friedberg Jump-Inversion Theorem—that are resolved by 1-generic sets but which cannot be resolved by essentially any weaker notion of genericity. This paper defines *aw2-generic sets*, i.e., sets which meet every dense set of strings that is r.e. in some incomplete r.e. set. *Aw*2-generic sets are very close to 1-generic sets in strength, but are too weak to resolve these questions. In particular, it is shown that for any set X there is an *aw*2-generic set G such that $\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \not\subseteq \mathbf{P}^{G\oplus X}$. (On the other hand, if G is 1-generic, then $\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \subseteq \mathbf{P}^{G\oplus \mathrm{SAT}}$, where SAT is the **NP**-complete Satisfiability problem [6].) This result runs counter to the fact that most finite extension constructions in complexity theory can be made effective. These results imply that any finite extension construction that ensures any of the Friedberg analogs must be noneffective, even relative to an arbitrary incomplete r.e. set. It is then shown that the recursion theoretic properties of *aw*2-generic sets differ radically from those of 1-generic sets: every degree above $\mathbf{0}'$ contains an *aw*2-generic set; no *aw*2-generic set exists below any incomplete r.e. set; there is an *aw*2-generic set which is the join of two Turing equivalent *aw*2-generic sets. Finally, a result of Shore is presented [30] which states that every degree above $\mathbf{0}'$ is the jump of an *aw*2-generic degree.

1 Introduction

The Friedberg Completeness Criterion [14] states that any Turing degree above $\mathbf{0}'$ is relatively complete, i.e., it is the jump of another degree. One can prove this result by looking at 1-generic sets (i.e., sets Cohen generic for one-quantifier first order arithmetic) [20, 19, 31].

Journal of Symbolic Logic, 59(3):868–887, 1994. An earlier version appeared in Proceedings of the Sixth Annual IEEE Structure in Complexity Theory Conference, 1991

†Partially supported by NSF Grant CCR 92-09833. Some of these results also appeared in [Fen91a], [Fen91b], and [Fen91c], and were proved while the author was a graduate student in the University of Chicago Computer Science Department, with the support of a University of Chicago Fellowship. Email: fenner@usm.maine.edu

It is well-known [19, Lemma 2.6 (ii)] that for any 1-generic set G,

$$G' \leq_{\mathrm{T}} G \oplus K, \tag{1}$$

where K denotes the halting problem. Then, for any set S with $K \leq_{\mathrm{T}} S$, a 1-generic set G can be constructed such that

$$G \oplus K \equiv_{\mathrm{T}} S. \tag{2}$$

See [31, pp. 97–98] for a complete proof along these lines.

There are interesting complexity theoretic analogs of equation (1). In the current paper we look at equation (1) and its analogs to determine "how much" genericity is actually needed to prove them. We will show that 1-genericity is necessary in the following sense: essentially no weaker notion of genericity suffices. Later, we will also show that the same question about equation (2) yields the same answer.

A natural candidate for a complexity theoretic version of (1) would state that if G is 1-generic, then some $\mathbf{NP}^G$-complete set is polynomial-time Turing reducible to $G \oplus \mathrm{SAT}$, where SAT encodes the **NP**-complete Satisfiability problem. In other words,

$$\mathbf{NP}^G \subseteq \mathbf{P}^{G \oplus \mathrm{SAT}}. \tag{3}$$

This statement is unfortunately false, as can be shown by a straightforward forcing argument (see [28] for example). There are, however, a number of results in complexity theory which approximate equation (3) and whose proofs are similar to that of equation (1). The best known of these is due to Blum & Impagliazzo [6], which in essence states that for every 1-generic set G,

$$\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \subseteq \mathbf{P}^{G \oplus \mathrm{SAT}}. \tag{4}$$

See Appendix A for a list of the known results of this type.

Since equation (4) relates to polynomial time-bounded computations, it is natural to ask if it holds for generic sets in some weaker, perhaps polynomial time-bounded sense. Several recursive and subrecursive notions of genericity have been studied and applied successfully in complexity theory [3, 13, 23, 25, 9]. For example, Ambos-Spies, Fleischhack, & Huwig introduced p-generic sets in [2, 3] as those having all properties enforcible by "p-standard diagonalizations," e.g., nonmembership in **P**, p-immunity, non-p-selectivity, etc. Building on earlier work of Mehlhorn and Lisagor in recursive Baire category, Lutz [23, 25] introduced the alternate notion of a Γ-generic set for any function class Γ, subrecursive or otherwise. He showed, for example, that initial segments of **PSPACE**-generic sets have high circuit size complexity and high space-bounded Kolmogorov complexity infinitely often [25].[1] The present author has shown [9] that for certain time-bounded complexity classes Γ, Γ-generic oracles suffice for proving many routine separation results in relativized complexity. For example, if we let **FP** denote the class of polynomial-time functions, then

$$(\forall\ \mathbf{FP}\text{-generic } G)\mathbf{P}^G \neq \mathbf{NP}^G \neq \text{co-}\mathbf{NP}^G. \tag{5}$$

[1] Lutz has also developed resource-bounded notions of Lebesgue measure and randomness [24, 25].

Stronger resource-bounded generic sets are also enough to separate the polynomial hierarchy [8].

One important reason for studying these weak, subrecursive notions of genericity is that there are recursive sets which fit the definitions. Resource-bounded genericity is closely tied to the notions of resource-bounded Baire category studied in [23, 9, 26]. They lead to satisfactory "almost all" theories of many common complexity classes—even though these classes are countable. We might call sets which are generic in some resource-bounded sense 'pseudogeneric'. Truly generic and n-generic sets, their relationship to Baire category, and their uses in recursion theory have been known for a long time (see [19, 29]).

Another important reason for studying pseudogeneric sets is that they provide a way to measure quantitatively the inherent difficulty of a finite extension argument in complexity theory. Generic and n-generic sets embody the method of diagonalization by finitely extending initial segments in recursion theory. As with truly generic sets, pseudogeneric sets also embody methods of diagonalization via finite extension, but only those using limited computational resources. Results such as equation (5) above express the fact that most finite extension constructions in complexity theory can be made effective and use limited resources—usually no more than an exponential blowup in the bounds used to define the complexity classes involved.

With the success pseudogeneric oracles have had in complexity theory, it is perhaps surprising that no recursive or subrecursive notion of genericity is strong enough to guarantee equation (4). In fact, we will argue that essentially no notion of genericity weaker than 1-genericity suffices to guarantee equation (4) or any of the results of similar type listed in Appendix A. To do this, we define a particular type of genericity, which we call *almost weakly 2-genericity* (or $aw2$-genericity for short), in a fashion similar to the weakly n-generic sets studied by Kurtz [22]. Given an arbitrary set X, we then construct an $aw2$-generic set G such that

$$\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \not\subseteq \mathbf{P}^{G \oplus X} \tag{6}$$

(Theorem 8), and our construction can easily be altered to defeat all of the other results listed in Appendix A simultaneously.

We define $aw2$-genericity precisely in Section 4. In short, an $aw2$-generic set must meet every dense set of strings which is r.e. in some incomplete r.e. degree. Restricting the requirement so that only dense r.e. sets of strings need be met yields exactly the weakly 1-generic sets. Expanding the requirement a little bit to include meeting dense sets r.e. in $\mathbf{0}'$ yields the weakly 2-generic sets, which are themselves all 1-generic (the hierarchy of weak n-generic sets interleaves with the hierarchy of n-generic sets [22]). Thus by merely adding $\mathbf{0}'$ to the requirement, we reverse the status of (4) from false to true.

Equation (4) and the similar results in Appendix A thus stand far apart from other results in relativized complexity: the diagonalizations used to prove them are inherently noneffective, even relative to any incomplete r.e. set. By contrast, there are exponential-time computable $\mathbf{FP}$-generics, so those resources are all that are needed for equation (5). As a consequence, we may classify finite extension arguments in complexity theory as either

easy or *hard*, depending on whether or not recursive or subrecursive notions of genericity suffice to prove them.

Section 5 consists mainly of our proof that equation 4 does not hold for $aw2$-generic sets. This proof is the only part of the paper that is chiefly complexity theoretic, and nothing later in the paper depends on it. Thus, the proof may be safely skipped.

In Sections 6 and 7 we turn our attention to the purely recursion theoretic properties of $aw2$-generic sets. Comparing the degrees of $aw2$-generic sets with the 1-generic degrees in Section 6, we observe that the two notions are incomparable and very different: Whereas it is known that 1-generic degrees exist below every nonzero r.e. degree, we show that no $aw2$-generic degree exists below any incomplete r.e. degree. Above $\mathbf{0}'$ the tables are completely turned; no degree above $\mathbf{0}'$ is 1-generic, but *every* degree above $\mathbf{0}'$ is $aw2$-generic. Our proof bears an interesting contrast with the proof of equation (2) above: there, the set S is coded into the 1-generic set G, but K is needed to find where the set S is actually coded; in our construction, we make an $aw2$-generic $G \equiv_{\mathrm{T}} S$ by coding the bits of S *together* with the information needed to find them all into G itself. Thus S is computable from G without the use of K. All our proofs make crucial use of the Arslanov Completeness Criterion (see [31]). We obtain as a corollary that every degree $\mathbf{a} \geq \mathbf{0}'$ is hyperimmune with respect to every r.e. degree $\mathbf{b} < \mathbf{0}'$, by adapting a result in [22] (see [22] for definitions).

We further contrast $aw2$-generic sets with 1-generic sets in Section 7 by constructing an $aw2$-generic set G which can be computed in any nonrecursive $B \leq_m G$. This G is then the join of two Turing equivalent $aw2$-generic sets, and also the join of infinitely many $aw2$-generic sets, all Turing equivalent to G. Finally, we reproduce an unpublished proof by Shore that $aw2$-generics can also be used to prove the Friedberg Completeness Criterion, which immediately implies the existence of low $aw2$-generic sets below $\mathbf{0}'$.

There are several remaining open questions regarding $aw2$-generic sets which we pose in Section 8.

2 Preliminaries

We adopt more or less the notation of [31]. We let ω be the set of natural numbers, we let $2^{<\omega}$ denote the set of finite 0-1 sequences (binary strings), and we let 2^{ω} denote the set of infinite 0-1 sequences, which we identify with the power set of ω. We also identify strings with natural numbers by the usual binary representation. We normally denote strings using lower case Greek letters, except when we use them as inputs to computations (natural numbers), in which case they are usually denoted with lower case Roman letters. We further identify a set $A \subseteq \omega$ with its characteristic function $\chi_A : \omega \to \{0,1\}$. If σ is a string, we let $|\sigma|$ denote the length of σ (likewise, $|x|$ is the length of the binary representation for the natural number x), and we denote the empty (length 0) string by $\emptyset$. Note that if $x \leq y$, then $|x| \leq |y|$. If τ is a string, we write $\sigma\hat{}\tau$ to mean the concatenation of σ followed by τ. We use 0^n or 1^n to denote the concatenation of n 0's or n 1's, respectively. Note that 0^n represents the least natural number of length n. If $f \in 2^{<\omega} \cup 2^{\omega}$, we write $\sigma \preceq f$ to mean

that σ is extended by (is a prefix of) f, and we write $\sigma \prec f$ to mean that $\sigma \preceq f$ and $\sigma \neq f$.

If ψ is a (partial) function, we denote the domain and range of ψ by $\mathrm{dom}(\psi)$ and $\mathrm{range}(\psi)$, respectively. We write $\psi(x)\downarrow$ or $\psi(x)\uparrow$ to mean that x is or is not in $\mathrm{dom}(\psi)$ respectively. We extend this notation to strings by identifying them with functions with domain a finite initial segment of ω and range $\{0,1\}$. We define the join $A \oplus B$ of two sets $A, B \subseteq \omega$ as $\{2x \mid x \in A\} \cup \{2x+1 \mid x \in B\}$. We will also have occasion to join a string σ with a set X, as in $\sigma \oplus X$. In this case, we always interpret σ as the finite set $\{x \in \omega \mid \sigma(x)\downarrow = 1\}$.

We let $\varphi_0, \varphi_1, \varphi_2, \ldots$ and $\{0\}, \{1\}, \{2\}, \ldots$ be acceptable numberings of partial recursive functions and relativized partial recursive functions, respectively. As is customary, we define $W_e \stackrel{\mathrm{df}}{=} \mathrm{dom}(\varphi_e)$. We let $\{e\}^A$ denote the eth function partial recursive in the set $A \subseteq \omega$, and we let $A\restriction k$ denote the set $\{x \in \omega \mid x \in A \ \ \& \ \ x < k\}$. We fix a recursive enumeration $K_0, K_1, K_2, \ldots$ of K. Finally, we fix a one-to-one pairing function $\langle \cdot, \cdot \rangle$ from $\omega \times \omega$ onto ω which is recursive and recursively invertible; the one defined in [29] will do. Other recursion theoretic concepts and definitions used can be found in [29] or [31].

Our complexity theoretic notation is standard. We use the usual Turing machine model for resource-bounded computation, with input, intermediate calculation, and output all in binary. We say a machine *accepts* its input if the output is nonzero; otherwise it *rejects*. A machine *recognizes* a set S if it accepts an input x if $x \in S$ and rejects x otherwise. For relativized computation, we assume the machine explicitly writes its queries in binary on a separate tape; the query is then replaced by the oracle answer in one step. We define $\mathbf{P}$ ($\mathbf{NP}$) as the class of all sets recognized in deterministic (nondeterministic) polynomial time. The class co-$\mathbf{NP}$ consists of the complements of $\mathbf{NP}$ sets. For a set $A \subseteq \omega$, $\mathbf{P}^A$ is the class of sets recognizable in polynomial time with oracle A. $\mathbf{NP}^A$ and co-$\mathbf{NP}^A$ are defined similarly. See [17] for more details.

3 n-Generic and Weakly n-Generic Sets

A set $S \subseteq 2^{<\omega}$ is *dense* if every string has an extension in S, i.e.,

$$(\forall \sigma \in 2^{<\omega})(\exists \tau \in S)\sigma \preceq \tau.$$

If S is a set of strings and $A \in 2^\omega$, we say that A *meets* S if there is a $\sigma \in S$ such that $\sigma \prec A$. We say that A *strongly avoids* S if there is a $\sigma \prec A$ such that for all $\tau \succeq \sigma$, $\tau \notin S$. We now give recursion theoretic definitions of two types of genericity. The first was originally defined in [16], and further studied in [19].

Definition 1 (Hinman) *For $n \geq 1$, a set G is n-generic if G either meets or strongly avoids every Σ^0_n set of strings. In particular, a 1-generic set meets or strongly avoids every r.e. set of strings.*

The second type of genericity was defined and studied in [22].

Definition 2 (Kurtz) *For $n \geq 1$, a set G is weakly n-generic if G meets every dense Σ^0_n set of strings. In particular, a weakly 1-generic set meets every dense r.e. set of strings.*

Kurtz [22] showed that the weakly 1-generic degrees are exactly the hyperimmune degrees, and also that the notions of n-genericity and weak n-genericity strictly interleave in strength.

We are mainly interested in the following theorem (equation (4) above), which was essentially proven in [6]:

Theorem 3 (Blum, Impagliazzo) *If G is a 1-generic set, then*

$$\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \subseteq \mathbf{P}^{G \oplus \text{SAT}},$$

where SAT *is the* **NP***-complete set of satisfiable Boolean formulae.*

As was mentioned above, their proof technique is similar to that used to prove (1), and has been used to prove several other results with the same flavor (see Appendix A). One of our aims is to show that 1-genericity is *necessary* to prove Theorem 3, i.e., that a reasonable, slightly weaker notion of genericity does not suffice to prove Theorem 3 or any result similar to it. To do this, we will define $aw2$-generic sets (Definition 7), then show (Theorem 8) that there is an $aw2$-generic set G which fails to satisfy the conclusion of Theorem 3. To do this, we will first generalize the above notions of genericity by defining a Γ-generic set for an arbitrary class of partial functions Γ.

4 Γ-Generic and $Aw2$-Generic Sets

Definition 4 *A partial genericity requirement is a partial function $h: 2^{<\omega} \to 2^{<\omega}$ such that $\sigma \preceq h(\sigma)$ for all $\sigma \in 2^{<\omega}$ such that $h(\sigma)\!\downarrow$. A genericity requirement is a partial genericity requirement that is total. A string τ meets a partial genericity requirement h if there is a $\sigma \preceq \tau$ such that either $h(\sigma)\!\uparrow$ or $h(\sigma) \preceq \tau$. If $A \in 2^\omega$, we say that A meets h if some $\tau \prec A$ meets h.*

Note that A meets the partial genericity requirement h if and only if A meets or strongly avoids range(h) as a set of strings.

Definition 5 *Let Γ be an arbitrary countable class of partial functions. A set $G \in 2^\omega$ is Γ-generic if G meets every partial genericity requirement in Γ.*

This definition originates out of the work of Lutz in resource-bounded Baire category [23, 25]. It is useful to define Γ-genericity in terms of meeting functions rather than sets of strings, as this definition works particularly well for subrecursive classes Γ. Despite the change of emphasis, Γ-genericity includes the other definitions of genericity given above: For $n \geq 1$ let Γ_n and $\tilde{\Gamma}_n$ be the $\emptyset^{(n-1)}$-partial recursive and $\emptyset^{(n-1)}$-total recursive functions,

respectively. It is easy to show that the Γ_n-generic sets are exactly the n-generic sets, and the $\tilde{\Gamma}_n$-generic sets are exactly the weakly n-generic sets.

We now define $aw2$-genericity.

Definition 6 *If* $\mathbf{d}$ *is a Turing degree, define the class of total functions*

$$\mathrm{rec}(\mathbf{d}) \stackrel{\mathrm{df}}{=} \{f \mid (\exists A \in \mathbf{d}) f \leq_{\mathrm{T}} A\}.$$

Definition 7 *A set* G *is* $aw2$*-generic if* G *is* $\mathrm{rec}(\mathbf{d})$*-generic for every r.e. degree* $\mathbf{d} < \mathbf{0}'$.

If we included $\mathbf{0}'$ itself in Definition 7, we would define a $\mathrm{rec}(\mathbf{0}')$-generic set, which is the same as a weakly 2-generic set; for this reason, we call $aw2$-generic sets 'almost weakly 2-generic'.[2] All weakly 2-generic sets are 1-generic (not conversely [22]), so Theorem 3 holds for them. It is interesting that merely adding this last r.e. degree produces enormous differences in the properties of the resulting generic sets. We will look at more of these differences in Sections 6 and 7.

We will restrict our attention to the r.e. degrees only, and not concern ourselves with the properties of sets which are $\mathrm{rec}(\mathbf{d})$-generic for every incomplete Δ^0_2 degree $\mathbf{d}$. We currently know little about such sets, and we pose as an open question whether they are the same as weakly 2-generic sets.

5 Theorem 3 Fails for $Aw2$-Generic Sets

We now state one of our two main results, which says that $aw2$-genericity is not sufficient for proving Theorem 3. In fact, our result generalizes for an arbitrary set joined with G, not necessarily SAT. The proof can be modified easily to construct an $aw2$-generic G to cause all the results listed in Appendix A to fail simultaneously.

Theorem 8 *For every set* $X \subseteq \omega$*, there exists an* $aw2$*-generic set* G *such that*

$$\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \not\subseteq \mathbf{P}^{G \oplus X}.$$

Corollary 9 *There exists an* $aw2$*-generic set* G *such that*

$$\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \not\subseteq \mathbf{P}^{G \oplus \mathrm{SAT}}.$$

We will prove Theorem 8 by an initial segment construction with no injury. To do this, we will need Lemma 12, which itself depends on two crucial facts regarding functions recursive in r.e. sets. The first of these is a generalization of the recursion theorem called the Arslanov Completeness Criterion (see [31, page 88]).

Theorem 10 (Arslanov) *An r.e. set* A *is complete if and only if there is a function* $f \leq_{\mathrm{T}} A$ *such that* $W_{f(x)} \neq W_x$ *for all* x.

[2] These sets were originally called *inc*-generic in [10].

The second fact is a result of Jockusch relating Arslanov's criterion with the existence of diagonally nonrecursive functions (see [31, page 90, exercise V.5.8]).

Theorem 11 (Jockusch) *If A is an arbitrary set, then*

$$(\exists f \leq_{\mathrm{T}} A)(\forall e)[W_e \neq W_{f(e)}] \Longleftrightarrow (\exists h \leq_{\mathrm{T}} A)(\forall e)[h(e) \neq \varphi_e(e)].$$

The two facts together immediately imply the following lemma:

Lemma 12 *If $\mathbf{d}$ is an incomplete r.e. degree and $f \in \mathrm{rec}(\mathbf{d})$, then there exist infinitely many e such that $f(e) = \varphi_e(e)$.*

Proof: Suppose there are only finitely many such e. Then there exists an $\tilde{f}$ differing from f on only finitely many values such that $(\forall e)[\tilde{f}(e) \neq \varphi_e(e)]$. Let $A \in \mathbf{d}$ be an r.e. set. Since $\tilde{f} \leq_{\mathrm{T}} A$, A must be complete. $\Box$

Proof of Theorem 8: Fix an arbitrary set $X \subseteq \omega$. The basic idea is that we build G by alternating between meeting genericity requirements and diagonalizing against polynomial-time deterministic oracle Turing machines. We diagonalize by a standard **NP** hiding trick played in infinitely many *coding regions*. We meet genericity requirements between the coding regions. For these two tasks not to conflict with each other, we must be able to tell in polynomial time whether or not we are inside a coding region. We ensure this by meeting each genericity requirement h only on an input e where $h(e) = \varphi_e(e)$ (we also encode e into the oracle). A polynomial-time oracle machine can then recognize the coding regions (without knowing h) by computing $\varphi_e(e)$.

Let $\{g_0, g_1, g_2, \ldots\}$ be the set of all genericity requirements g such that there exists an r.e. degree $\mathbf{d} < \mathbf{0}'$ with $g \in \mathrm{rec}(\mathbf{d})$. (The particular enumeration we choose for this set is not important, since at this point we do not seek to control the complexity of G. In the proof of Theorem 14, however, we do need to control the complexity of G, so there we will proceed more carefully.) For any $\sigma \in 2^{<\omega}$ let $\ell(\sigma)$ be the least n such that $\sigma(x)\uparrow$ for all x with $|x| \geq n$. It will be convenient in this proof to define a function $\mathcal{Z}$ so that for any string σ and number n, $\mathcal{Z}(\sigma, n) = \sigma$ if $n < \ell(\sigma)$, and otherwise $\mathcal{Z}(\sigma, n)$ is σ extended with just enough 0's so as to be defined on exactly those numbers of length strictly less than n. Let $\{C_e\}_{e \in \omega}$ be a set of partial recursive functions such that

1. $(\forall e, x \in \omega)\ \varphi_e(x)\downarrow \Longleftrightarrow C_e(x)\downarrow$,
2. $(\forall e, x \in \omega)\ \varphi_e(x)\downarrow \Longrightarrow \varphi_e(x) \leq C_e(x)$, and
3. the predicate "$C_e(u)\downarrow \leq v$" is computable in time bounded by a polynomial in $|e|$, u, and v.

The set $\{C_e\}_{e\in\omega}$ corresponds to a Blum complexity measure [5] with certain additional restrictions on its values and running time. Such C_e clearly exist; for example, we may define $C_e(x) \stackrel{\text{df}}{=} 0^{\rho(e,x)+1}$, where $\rho(e,x)$ is the running time of the eth Turing machine on input x (we assume the input and output of the machine are both in binary). Also let $P_0, P_1, P_2, \ldots$ be an enumeration of all polynomial-time deterministic oracle Turing machines (see [17] for example), each P_i running in time $p_i(n) \stackrel{\text{df}}{=} n^i + i$ for all oracles. Finally, for all i let m_i be least such that $p_i(n) < 2^{n-1}$ for all $n \geq m_i$.

We construct G as a limit of binary strings γ_i,

$$\emptyset = \gamma_0 \prec \gamma_1 \prec \gamma_2 \prec \ldots \prec G.$$

We build $\gamma_0, \gamma_1, \ldots$ so that for all i, γ_{i+1} meets g_i (and thus G meets g_i, which makes G $aw2$-generic). Also, γ_{i+1} ensures that the set L defined below is not recognized by $P_i^{G\oplus X}$.

$$L \stackrel{\text{df}}{=} \{0^d \in D \mid (\exists x)[|x| = d-1 \ \&\ x\hat{}0 \in G]\},$$

where $D \stackrel{\text{df}}{=} \{0^{d_0}, 0^{d_1}, 0^{d_2}, \ldots\}$, and the d_i are defined below in the construction of G. We will show that $D \in \mathbf{P}^G$, thus evidently $L \in \mathbf{NP}^G$. We will also ensure that for all $0^d \in D$,

$$(\exists x)[|x| = d-1 \ \ \&\ \ x\hat{}0 \in G] \iff \neg(\exists y)[|y| = d-1 \ \ \&\ \ y\hat{}1 \in G], \tag{7}$$

so $L \in$ co-$\mathbf{NP}^G$ as well. We keep L out of $\mathbf{P}^{G\oplus X}$ by the explicit diagonalization mentioned above, which will prove the theorem.

Let $\gamma_0 \stackrel{\text{df}}{=} \emptyset$. Given γ_i, we define γ_{i+1} as follows:

1. Define $f_i \stackrel{\text{df}}{=} \lambda n.\ell(g_i(\mathcal{Z}(\gamma_i, n)\hat{}1))$. Note that $f_i \leq_{\mathrm{T}} g_i$, so f_i satisfies the hypothesis of Lemma 12.

2. Let $n_i \stackrel{\text{df}}{=} (\mu n)[n \geq m_i \ \ \&\ \ n \geq \ell(\gamma_i) \ \ \&\ \ f_i(n) = \varphi_n(n)]$. The number n_i exists by Lemma 12.

3. Let $\gamma \stackrel{\text{df}}{=} \mathcal{Z}(\gamma_i, n_i)\hat{}1$. Notice that $\gamma_i \prec \gamma$ and $\gamma(0^{n_i}) = 1$.

4. Let $\gamma' \stackrel{\text{df}}{=} g_i(\gamma)$. Note that $\gamma \preceq \gamma'$ and γ' meets g_i. In step 8, we will define γ_{i+1} as an extension of γ', so γ_{i+1} meets g_i as well. Notice also that

   $$\ell(\gamma') = f_i(n_i) = \varphi_{n_i}(n_i) \leq C_{n_i}(n_i)\!\downarrow$$

   by our assumptions about C_{n_i} above.

5. Let $d_i \stackrel{\text{df}}{=} C_{n_i}(n_i)$. By the remark in step 4, γ' is undefined on all numbers of length d_i, and since $d_i > n_i \geq m_i$, we have $p_i(d_i) < 2^{d_i-1}$. (The ith coding region consists of all numbers of length d_i.)

6. Let $\gamma'' \stackrel{\text{df}}{=} \mathcal{Z}(\gamma', p_i(d_i) + 1)$.

7. Let x and y be the least numbers of length $d_i - 1$ such that the machine $P_i^{\gamma'' \oplus X}$ on input 0^{d_i} queries neither $x\hat{}0$ nor $y\hat{}1$. (See the Section 2 regarding the meaning of $\gamma'' \oplus X$.) Such x and y exist by the previous remark about d_i.

8. If $P_i^{\gamma'' \oplus X}(0^{d_i})$ accepts, let γ_{i+1} be the same as γ'' except that $\gamma_{i+1}(y\hat{}1) = 1$. If $P_i^{\gamma'' \oplus X}(0^{d_i})$ rejects, let γ_{i+1} be the same as γ'' except that $\gamma_{i+1}(x\hat{}0) = 1$. Here we diagonalize against P_i, preserving its erroneous computation. Note that $\gamma' \preceq \gamma_{i+1}$, and that γ_{i+1} preserves all queries made by $P_i^{\gamma'' \oplus X}(0^{d_i})$.

The set $G \stackrel{\text{df}}{=} \bigcup_{i \in \omega} \gamma_i$ meets all the g_i by step 4, thus G is $\text{rec}(\mathbf{d})$-generic for all r.e. $\mathbf{d} < \mathbf{0}'$, and thus G is $aw2$-generic. Recall that $D \stackrel{\text{df}}{=} \{0^{d_0}, 0^{d_1}, 0^{d_2}, \ldots\}$ from the construction above, and

$$L = \{0^d \in D \mid (\exists x)[|x| = d - 1 \ \&\ x\hat{}0 \in G]\}.$$

By the diagonalization step 8, we have $L(0^{d_i}) \neq P_i^{G \oplus X}(0^{d_i})$ for every i, thus $L \notin \mathbf{P}^{G \oplus X}$. Also by step 8, equation (7) above is maintained for all $0^d \in D$. This implies that $L \in \mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G$, provided D is easy to compute from G. We complete the proof by showing that $D \in \mathbf{P}^G$.

First, notice that $n_0 < d_0 < n_1 < d_1 < \ldots$. Fix an input 0^d. We reconstruct k and $d_0, \ldots, d_k$ such that $d_k \leq d < d_{k+1}$. Then $0^d \in D$ if and only if $d = d_k$. The algorithm is given in the next paragraph. Since we coded a 1 into G at each point 0^{n_i} for $i \leq k$, we can tell exactly where G meets each g_i—at the γ defined in step 3. Since by step 4, $g_i(\gamma)$ is defined only on strings strictly shorter than $\ell(C_{n_i}(n_i))$, we can then "skip over" $g_i(\gamma)$ by computing the value $d_i = C_{n_i}(n_i)$ and ignoring the oracle between length n_i and d_i. This computation runs in time polynomial in n_i by our assumptions about $\{C_e\}_{e \in \omega}$. [For this to work, it was crucial in the construction of G that we could choose n_i so that $\ell(g_i(\gamma))$ was bounded by $C_{n_i}(n_i)$. This in turn relied on Lemma 12.]

We start with $n \stackrel{\text{df}}{=} 0$ and let n increase throughout the algorithm up to d. To find d_0, we query G on $\emptyset, 0, 00, \ldots, 0^n, \ldots$ until either $n = d$ or query 0^n returns 1. If $n = d$, reject; otherwise $n = n_0$ by step 3, and so $d_0 = C_{n_0}(n_0) = C_n(n)$ from step 5. If $C_n(n) > d$, reject; if $C_n(n) = d$, accept. Otherwise compute the actual value of $d_0 = C_n(n)$ by linear search using the predicate $P(y) \stackrel{\text{df}}{=} [C_n(n) \leq y]$. Now to find d_1, query G on $0^{d_0+1}, 0^{d_0+2}, \ldots, 0^n, \ldots$ until either $n = d$ or query 0^n returns 1. If $n = d$, reject; otherwise $n = n_1$ by step 3, so $d_1 = C_{n_1}(n_1) = C_n(n)$. If $C_n(n) > d$, reject; if $C_n(n) = d$ accept. Otherwise, compute $d_1 = C_n(n) < d$ as before. We continue this process to find n_2, d_2, n_3, d_3, etc., stopping when we get up to d. The entire procedure takes time polynomial in d, and accepts if and only if $0^d \in D$. This completes the proof. $\square$

If we neglect the non-r.e. degrees below $\mathbf{0}'$ and restrict our attention just to the r.e. degrees, then Theorem 8 gives the best possible lower bound on genericity for guaranteeing (4). As was mentioned above, if we include the complete degree $\mathbf{0}'$, then $\text{rec}(\mathbf{0}')$-generics are exactly the weakly 2-generics, and hence they are 1-generics as well, and so Theorem 3 holds for them.

With a little care, the construction above can be altered to make $G \leq_T \emptyset'' \oplus X$ as follows: replace the g_i with the partial functions $\psi_i \stackrel{\text{df}}{=} \{j\}^{W_k}$ where $i \stackrel{\text{df}}{=} \langle j,k \rangle$. We need not worry about whether any particular ψ_i is a genericity requirement with W_k incomplete, as long as we make sure in step 2 that n_i exists. That is, given γ_i we check if there exists an $n \geq \max(m_i, \ell(\gamma_i))$ such that

1. $\psi_i(\mathcal{Z}(\gamma_i, n)\hat{}1)\downarrow \succeq \gamma_i$, and
2. $\ell(\psi_i(\mathcal{Z}(\gamma_i, n)\hat{}1)) = \varphi_n(n)$.

If no such n exists, set $\gamma_{i+1} \stackrel{\text{df}}{=} \gamma_i$. Otherwise, set n_i to be the first such n we find and continue with the algorithm. Questions 1 and 2 above can both be answered in $\emptyset''$. The set X is needed in steps 7 and 8.

We can do better than $G \leq_T \emptyset'' \oplus X$, however. The proof of Theorem 14 in the next section can be modified easily to get $G \leq_T K \oplus X$, or more strongly, $G \equiv_T S \oplus X$ for any set S such that $K \leq_T S$. See the remark following Theorem 14.

6 Degrees of $Aw2$-Generic Sets

The degrees of $aw2$-generic sets differ drastically from 1-generic degrees inside the arithmetic hierarchy. As mentioned before, below every nonrecursive r.e. set there is a 1-generic set, but no $aw2$-generic set exists below any incomplete r.e. set by Proposition 13 below. On the other hand, no 1-generic sets exist above $\mathbf{0}'$ by virtue of equation (1), but as we show in Theorem 14, $aw2$-generic sets exist in *all* degrees above $\mathbf{0}'$. Therefore, equation (1) does not hold for all $aw2$-generic sets, although it holds for "enough" of them, as we show by Theorem 19.

Proposition 13 *If A is an incomplete r.e. set, then there is no $aw2$-generic $G \leq_T A$.*

Proof: Suppose $G \leq_T A$. Define $h(\sigma) \stackrel{\text{df}}{=} \sigma\hat{}b$ where $b \stackrel{\text{df}}{=} 1 - G(|\sigma|)$. Clearly $h \leq_T G \leq_T A$, and h is a genericity requirement unmet by G. Thus G cannot be $aw2$-generic. □

The crucial property of our construction in the proof of Theorem 8 was that there were infinitely many regions (the numbers of length d_i) where we could perform arbitrary coding without affecting the $aw2$-genericity of G. Moreover, these regions were decidable in $\mathbf{P}^G$, independent of what we put in them. We can adapt this technique to code an arbitrary set S into G while meeting all the other $aw2$-genericity requirements. This will allow us in Theorem 14 to construct an $aw2$-generic set in every degree $\mathbf{d} \geq \mathbf{0}'$. Thus $aw2$-genericity is essentially the strongest notion of genericity which is 'dense upwards' in the Turing degrees.

Theorem 14 is the second of our two main results. It bounds the complexity of G as tightly as possible, combining the self-coding technique from the proof of Theorem 8 with a permitting argument adapted from Shore's construction of a 1-generic set below every nonrecursive r.e. degree.

Theorem 14 (Kurtz, Fenner) *For every degree* $\mathbf{d} \geq \mathbf{0}'$ *there exists an* $aw2$*-generic set* $G \in \mathbf{d}$.

Proof: Fix an arbitrary set S such that $K \leq_{\mathrm{T}} S$. We will construct an $aw2$-generic $G \equiv_{\mathrm{T}} S$. We build G to satisfy the requirements

$R_{\langle e,i\rangle}$**:** If $K \not\leq_{\mathrm{T}} W_e$ and $\{i\}^{W_e}$ is a genericity requirement, then G meets $\{i\}^{W_e}$

for all $e, i \in \omega$. This implies that G is $aw2$-generic. We build G in stages $0, 1, 2, \ldots$ by initial segments. At each stage s we define a string γ_s so that

$$\emptyset = \gamma_0 \prec \gamma_1 \prec \gamma_2 \prec \ldots \prec G.$$

We code $S(s)$ as the last digit of γ_{s+1}, and as in Theorem 8, we code into γ_{s+1} the information on where to find $S(s)$. At each stage, we *act upon* at most one requirement, and each requirement, never being injured, is acted upon at most once, after which it is satisfied forever. The whole construction is recursive in S.

Define the function

$$t(x) \stackrel{\mathrm{df}}{=} (\mu z)[K_z \restriction x = K \restriction x].$$

Clearly, $t \equiv_{\mathrm{T}} K$.

Stage 0**:** $\gamma_0 \stackrel{\mathrm{df}}{=} \emptyset$.
End of Stage 0**.**

Stage $s+1$**:** Let $t \stackrel{\mathrm{df}}{=} t(s)$. Given γ_s we define $\gamma_{s+1} \succ \gamma_s$ as follows: let $\langle e, i\rangle \leq s$ be least such that $R_{\langle e,i\rangle}$ has not yet been acted upon and there exists a least $x \leq t$ such that (letting $\rho \stackrel{\mathrm{df}}{=} \gamma_s \hat{}\, 0^{x+1} \hat{}\, 1$),

1. $\{i\}^{W_e}(\rho)$ halts in no more than t steps, and
2. $\rho \preceq \{i\}^{W_e}(\rho) = \varphi_x(x) \leq t$.

If no such $\langle e, i\rangle$ exists, set $\gamma_{s+1} = \gamma_s \hat{}\, 1 \hat{}\, S(s)$. Otherwise, we act upon requirement $R_{\langle e,i\rangle}$ by setting $\gamma_{s+1} = \tau \hat{}\, S(s)$, where $\tau \stackrel{\mathrm{df}}{=} \{i\}^{W_e}(\rho) = \varphi_x(x)$. (Note that in this case γ_{s+1} meets $\{i\}^{W_e}$.)
End of Stage $s+1$**.**

The construction above is recursive in S: since $K \leq_{\mathrm{T}} S$ we can use S to compute $t = t(s)$ and W_e uniformly in e, from which condition (1) is effectively checkable. Condition (2) is checkable in S by asking if $\varphi_x(x)\!\downarrow$. It follows that $G \leq_{\mathrm{T}} S$.

Computing S from G can be done in a way similar to the proof of Theorem 8. To find the value of $S(y)$ we reconstruct $\gamma_0, \ldots, \gamma_{y+1}$, then read off the last digit of γ_{y+1}. Suppose we

are given $\gamma_i \prec G$ for $i \leq y$, and $|\gamma_i| = m$. We find γ_{i+1} by examining $G(m), G(m+1), G(m+2), \ldots$ to find the least j such that $G(m+j) = 1$. If $j = 0$, then $\gamma_{i+1} = \gamma_i \hat{} 1 \hat{} G(m+1)$ (that is, no requirement was acted upon at stage $i+1$). If $j > 0$, then compute $\tau \stackrel{\text{df}}{=} \varphi_{j-1}(j-1)$ from which $\gamma_{i+1} = \tau \hat{} G(|\tau|)$ (the computation $\varphi_{j-1}(j-1)$ must halt by our construction). Therefore, we can reconstruct $\gamma_0, \ldots, \gamma_{y+1}$ to find $S(y)$, so $G \equiv_{\mathrm{T}} S$.

Now we need only show that G satisfies all requirements. Suppose that $R_{\langle e,i \rangle}$ is the least requirement not satisfied by G. Since each requirement is acted upon at most once, there is a stage $s_0 > \langle e, i \rangle$ after which no lesser requirement is acted upon. Because $R_{\langle e,i \rangle}$ is never satisfied, it must be the case that W_e is incomplete and $\{i\}^{W_e}$ is a genericity requirement, otherwise $R_{\langle e,i \rangle}$ would be satisfied vacuously. We now describe how to compute the function t from W_e, which contradicts the fact that W_e is incomplete. Note that t is recursive in any function that dominates t, so it suffices to compute from W_e a function $\hat{f}$ such that $\hat{f}(s) \geq t(s)$ for all $s \geq s_0$.

Fix an arbitrary string $\gamma \in 2^{<\omega}$. Define the function

$$r_\gamma(x) \stackrel{\text{df}}{=} \{i\}^{W_e}(\gamma \hat{} 0^{x+1} \hat{} 1).$$

The function r_γ is clearly W_e-recursive. Since W_e is incomplete, by Lemma 12 there is an x such that $r_\gamma(x) = \varphi_x(x)$. Using W_e, we can search for such an x. Let x_γ be the first such x we find, and let c_γ be the number of steps it takes for $\{i\}^{W_e}(\gamma \hat{} 0^{x_\gamma+1} \hat{} 1)$ to halt. Both x_γ and c_γ can be computed from γ using W_e. Define the function

$$f(\gamma) \stackrel{\text{df}}{=} \max(\gamma, x_\gamma, c_\gamma, r_\gamma(x_\gamma)).$$

It is clear from the arguments above that $f \leq_{\mathrm{T}} W_e$.

We know that no requirement less than $\langle e, i \rangle$ is acted upon at any stage later than s_0. Therefore, for any $s \geq s_0$ it must be the case that

$$f(\gamma_s) > t(s), \tag{8}$$

otherwise conditions (1) and (2) would hold for $x = x_{\gamma_s}$, and $R_{\langle e,i \rangle}$ would be acted upon and satisfied at stage $s+1$, contradicting our hypothesis. We would now be done if we could only use W_e to compute γ_s for all $s \geq s_0$, but unfortunately we cannot hope to do this: we cannot compute the value of $S(s)$ from W_e, nor can we determine whether or not some greater requirement $R_{\langle e',i' \rangle}$ is acted upon in any given stage.

Fortunately, without knowing γ_s directly, we can still get an upper bound on $f(\gamma_s)$ for $s \geq s_0$, which is then sufficient to compute $t(s)$. Notice that at stage $s+1 > s_0$, we have

$$|\gamma_{s+1}| \leq \max(|\gamma_s| + 2, |t(s)| + 1) \leq |f(\gamma_s)| + 2$$

by condition (2), the definition of f, and equation (8). This implies that given γ_s there is only a finite set of possibilities for γ_{s+1}, which we can compute using f. To bound $t(s+1)$, we guess a string γ from among these possibilities, compute $f(\gamma)$ for each guess, then take

the maximum over all guesses. This value bounds $f(\gamma_{s+1})$—and hence $t(s+1)$—because γ_{s+1} is one of the guessed strings.

We enumerate a sequence of canonical representations of finite sets $V_0, V_1, V_2, \ldots \subseteq 2^{<\omega}$ effectively in f as follows:

$$\begin{aligned} V_0 &\stackrel{\text{df}}{=} \{\gamma_{s_0}\}, \\ V_{n+1} &\stackrel{\text{df}}{=} \{\sigma : (\exists \gamma \in V_n)|\sigma| \leq |f(\gamma)| + 2\}. \end{aligned}$$

All the V_n are finite, and a trivial induction shows that $\gamma_s \in V_{s-s_0}$ for all $s \geq s_0$. Now for all $s \in \omega$, define

$$\hat{f}(s) \stackrel{\text{df}}{=} \begin{cases} 0 & \text{if } s < s_0, \\ \max_{\gamma \in V_{s-s_0}} f(\gamma) & \text{if } s \geq s_0. \end{cases}$$

We have for all $s \geq s_0$,

$$\hat{f}(s) \geq f(\gamma_s) \geq t(s).$$

It follows that

$$t \leq_{\mathrm{T}} \hat{f} \leq_{\mathrm{T}} f \leq_{\mathrm{T}} W_e,$$

contradicting the assumption that W_e is incomplete. The theorem follows. □

Remark: Given an arbitrary $X \subseteq \omega$, we could easily modify the above proof to construct an $aw2$-generic $G \equiv_{\mathrm{T}} S \oplus X$ satisfying Theorem 8 and causing all the other results in Appendix A to fail simultaneously.

Corollary 15 *Every degree* $\mathbf{a} \geq \mathbf{0}'$ *is hyperimmune with respect to every r.e. degree* $\mathbf{b} < \mathbf{0}'$.

Proof Sketch: By slightly modifying the proof of the 'only if' part of Theorem 2.3 in [22], one can show that every $aw2$-generic degree is hyperimmune with respect to every incomplete r.e. degree. □

7 Other Properties of $Aw2$-Generic Sets

7.1 Many-One Degrees Below $Aw2$-Generic Sets

Jockusch [19, Prop. 2.9] showed that the ordering of the m-degrees (except $\{\emptyset\}$ and $\{\omega\}$) below any 1-generic set A is isomorphic to the inclusion ordering of the r.e. sets modulo the finite sets. The same holds for weakly 1-generic sets and thus for $aw2$-generic sets by the same proof, but there is one important difference between the 1-generic and $aw2$-generic cases: If A is 1-generic, then it is not hard to show that all non-maximum m-degrees below A have Turing degree strictly less than that of A. This assertion fails for $aw2$-generic sets

in the worst possible way, however. We show that there is an $aw2$-generic set G with 'm-minimal Turing degree' in the sense that there are no nonrecursive m-degrees below G that are not in the Turing degree of G. We obtain as a corollary that G is the join of two sets (or indeed infinitely many sets) all Turing equivalent to G itself. This stands in sharp contrast with 1-generic sets ([20, Lemma 2] and [19]).

Theorem 16 *There exists an $aw2$-generic set G such that, for all nonrecursive B, if $B \leq_m G$ then $G \leq_T B$.*

Theorem 16 rests on the following lemma, which proves the somewhat remarkable fact that there is an $aw2$-generic set which is fully encoded within any infinite recursive part of its characteristic function.

Lemma 17 *There exists an $aw2$-generic set G such that, for all infinite recursive sets A, $G \leq_T G \cap A$.*

Proof of Theorem 16: Let $B \leq_m G$ via the function f. Since B is nonrecursive, range(f) must be infinite. Let A be an infinite recursive subset of range(f). For all $x \in A$ we have

$$x \in G \iff (\mu z)[f(z) = x] \in B.$$

Thus $G \cap A \leq_T B$. By Lemma 17, $G \leq_T B$. $\square$

Proof of Lemma 17: Let $A_0, A_1, A_2, \ldots$ be an arbitrary listing of all the infinite recursive sets, and let $g_0, g_1, g_2, \ldots$ be an arbitrary listing of all the genericity requirements recursive in incomplete r.e. sets, as in the proof of Theorem 8 above. We again build G by initial segments

$$\emptyset = \gamma_0 \prec \gamma_1 \prec \gamma_2 \prec \cdots \prec G.$$

Given a string σ and $n \in \omega$, we will define a recursive function $\eta_{n,\sigma}(x)$ such that

1. $\sigma \preceq \eta_{n,\sigma}(x)$ for all $x \in \omega$.
2. For each $i \in \omega$, there is a recursive operator $\Xi_i(X;\sigma)$ such that if $n \geq i$, then for all $x \in \omega$ and for all sets $B \succ \eta_{n,\sigma}(x)$,

$$\Xi_i(B \cap A_i;\sigma) = x.$$

Intuitively, we use $\eta_{s,\gamma_s}(x)$ to extend to a larger portion of G, where x is chosen so that $\gamma_{s+1} = \varphi_x(x)$ as in previous proofs. The string $\eta_{s,\gamma_s}(x)$ is just long enough to encode x in a particular way. The operators Ξ_i are then designed to recover x by looking only at those positions of the oracle in $\mathrm{dom}(\eta_{s,\gamma_s}(x)) \cap A_i$. We will define $\eta_{n,\sigma}$ and Ξ_i precisely later on.

We construct G as follows:

Stage 0: $\gamma_0 \stackrel{\text{df}}{=} \emptyset$.
End of Stage 0.

Stage $s+1$: We are given $\gamma \stackrel{\text{df}}{=} \gamma_s$. Let x_s be the least x such that $g_s(\eta_{s,\gamma}(x)) = \varphi_x(x)$, and set γ_{s+1} to $g_s(\eta_{s,\gamma}(x_s)) = \varphi_{x_s}(x_s)$. [Note that $g_s \circ \eta_{s,\gamma}$ satisfies the hypothesis of Lemma 12.]
End of Stage $s+1$.

Set $G \stackrel{\text{df}}{=} \bigcup_s \gamma_s$. Clearly, G meets every g_s, so G is $aw2$-generic. Fix $i \in \omega$. We show that $G \leq_{\mathrm{T}} G \cap A_i$. Note first that for all $s \geq i$, we have $x_s = \Xi_i(G \cap A_i; \gamma_s)$ since $G \succ \eta_{s,\gamma_s}(x_s)$. Starting with γ_i, we compute $x_i = \Xi_i(G \cap A_i; \gamma_i)$. We then compute $\gamma_{i+1} = \varphi_{x_i}(x_i)$. We compute $x_{i+1} = \Xi_i(G \cap A_i; \gamma_{i+1})$ to get $\gamma_{i+2} = \varphi_{x_{i+1}}(x_{i+1})$ and so on, to reconstruct all of G.

It remains to define $\eta_{n,\sigma}$ and Ξ_i appropriately. The string $\eta_{n,\sigma}(x)$ must encode x so that for all $i \leq n$, $\Xi_i(B;\sigma)$ can recover x by looking at the oracle $B \succ \eta_{n,\sigma}(x)$ only at locations $y \in \mathrm{dom}(\eta_{n,\sigma}(x)) \cap A_i$. For $n = 0$ this is not a problem. Let $y_1 < y_2 < \cdots$ be the elements of A_0 outside $\mathrm{dom}(\sigma)$. Set $\eta_{0,\sigma}(x) \stackrel{\text{df}}{=} \sigma\hat{}00\cdots001$, where the last 1 occurs at position y_{x+1}. The function $\Xi_0(B;\sigma)$ then recovers x by determining where the first 1 occurs in the sequence $B(y_1), B(y_2), \ldots$. [Note that $B(y_j) = (B \cap A_0)(y_j)$.]

For $n > 0$, we would like to do the same trick simultaneously for $\Xi_0, \ldots, \Xi_n$. We must be careful to avoid conflicts, however. For example, if $n = 1$, the number $y_{x+1} \in A_0$ may also be in A_1, and placing a 1 at that position for the sake of Ξ_0 may mess up Ξ_1's count. We remedy this by having $\Xi_1(B;\sigma)$ read past the first 1 it sees (whether or not at position y_{x+1}), and take x to be the number of 0's between the first and second 1. Thus Ξ_1 ignores the position of the first 1, since it may be used to encode x for Ξ_0. In general, $\eta_{n,\sigma}(x)$ encodes x for $\Xi_0, \ldots, \Xi_n$ in descending order of priority, and each Ξ_i reads past a certain number of 1's before computing x.

We define $\Xi_i(B;\sigma)$ as follows: let $y_1 < y_2 < \cdots$ be the elements of A_i not in $\mathrm{dom}(\sigma)$. $\Xi_i(B;\sigma)$ examines the sequence $B(y_1), B(y_2), \ldots$ until exactly $2^i - 1$ many 1's appear, the last occurring at, say, $B(y_d)$. $\Xi_i(B;\sigma)$ continues examining the sequence $B(y_{d+1}), B(y_{d+2}), \ldots$ until the next 1 appears, say $B(y_c)$. $\Xi_i(B;\sigma)$ immediately outputs $c - d$ and halts.

Fix σ and x, and let η_n denote the string $\eta_{n,\sigma}(x)$. We will define η_n by induction on n, maintaining the following invariants throughout:

1. $\sigma \prec \eta_n$,
2. the last digit of η_n is 1, and
3. η_n has at most $2^{n+1} - 1$ many 1's at positions outside $\mathrm{dom}(\sigma)$.

We defined $\eta_0 = \eta_{0,\sigma}(x)$ above. For $n > 0$, assume $\eta \stackrel{\text{df}}{=} \eta_{n-1}$ is defined, and let $y_1 < y_2 < \cdots$ be the elements of A_n outside $\mathrm{dom}(\sigma)$. Let w be the number of distinct y_j

such that $\eta(y_j)\downarrow = 1$. By the third invariant, we have $w \leq 2^n - 1$. We first want to extend η to a string θ that has exactly $2^n - 1$ many 1's, including its last digit, among the positions $y_1, y_2, \ldots$. If $w = 2^n - 1$, then we can take $\theta \stackrel{\text{df}}{=} \eta$, since in this case the positions of all 1's in η outside $\text{dom}(\sigma)$ must be of the form y_j, including the final digit of η. If $w < 2^n - 1$, then let $v \stackrel{\text{df}}{=} 2^n - 1 - w$, let k be least such that $y_k \notin \text{dom}(\eta)$, and set

$$\theta \stackrel{\text{df}}{=} \eta \hat{} \underbrace{00\cdots0100\cdots0100\cdots01}_{v \text{ many } 1\text{'s}},$$

where the last v many 1's occur at positions $y_k, y_{k+1}, \ldots, y_{k+v-1}$. Finally, we define

$$\eta_n \stackrel{\text{df}}{=} \theta \hat{} 00\cdots01,$$

where the last 1 appears at position y_{k+v+x}. Note that there are exactly x many 0's appearing at positions y_{k+v+j} for $0 \leq j \leq x - 1$.

The first two invariants clearly hold for all n. The third invariant clearly holds for $n = 0$. Assume $n > 0$ and the third invariant holds for $n-1$. There are at most 2^n many additional $y \in \text{dom}(\eta_n) - \text{dom}(\eta_{n-1})$ with $\eta_n(y) = 1$. Thus the number of $y \notin \text{dom}(\sigma)$ with $\eta_n(y)\downarrow = 1$ is at most $2^n - 1 + 2^n = 2^{n+1} - 1$, so the third invariant holds for n.

The function $\eta_{n,\sigma}(x)$ is easily seen to be recursive for all n and σ: $\eta_{n,\sigma}(x)$ can be computed effectively given recursive indices for $A_0, \ldots, A_n$. By viewing the definitions of $\eta_{n,\sigma}$ and Ξ_i simultaneously, it is clear that $\Xi_i(B;\sigma) = x$ for all $i \leq n$ and $B \succ \eta_{n,\sigma}(x)$. Also, since $\Xi_i(B;\sigma)$ depends on B only at positions in A_i, we have $\Xi_i(B \cap A_i;\sigma) = \Xi_i(B;\sigma) = x$ as desired. $\square$

For a given set $S \subseteq \omega$, define S^0 and S^1 to be the unique sets such that $S^0 \oplus S^1 = S$, and for $n \in \omega$ define $S^{[n]} \stackrel{\text{df}}{=} \{x \mid \langle x, n\rangle \in S\}$. If A is 1-generic, then A^0 is Turing incomparable with A^1, and no set $A^{[n]}$ is computable in $\bigoplus_{k \neq n} A^{[k]}$ [20, 19]. By contrast, we have the following:

Corollary 18 *There exists an $aw2$-generic set G such that*

$$G \equiv_{\text{T}} G^0 \equiv_{\text{T}} G^1 \equiv_{\text{T}} G^{[0]} \equiv_{\text{T}} G^{[1]} \equiv_{\text{T}} G^{[2]} \equiv_{\text{T}} \ldots.$$

Remark: If G is $aw2$-generic, the most we can say about G^0 and G^1 is that they are both $aw2$-generic and truth-table incomparable. The same thing holds true for all the $G^{[n]}$, and also if '$aw2$-generic' is replaced by 'weakly 1-generic'.

7.2 Jump Inversion with $Aw2$-Generic Sets

Are there $aw2$-generic degrees strictly below $\mathbf{0}'\Gamma$ Since there are $aw2$-generics above K, it is not true that equation (1),

$$G' \leq_{\text{T}} G \oplus K,$$

holds for all $aw2$-generic G. Despite this, can the Friedberg Completeness Criterion be proved with $aw2$-generic setsΓ The answer to both these questions is yes. Shore [30] has modified the proof of Theorem 14 to prove the Friedberg Completeness Criterion using $aw2$-generic sets:

Theorem 19 (Shore) *For every degree $\mathbf{d} \geq \mathbf{0}'$ there is an $aw2$-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \in \mathbf{d}$. (G is also 1-generic.)*

Proof: Fix a set $S \in \mathbf{d}$. We build G by initial segments $\emptyset = \gamma_0 \prec \gamma_1 \prec \gamma_2 \prec \cdots \prec G$ in stages $0, 1, 2, \ldots$ as before, where $S(s)$ is coded as the last digit of γ_{s+1}. We need to satisfy the requirements

$R_{\langle e,i\rangle}$**:** If $K \not\leq_{\mathrm{T}} W_e$ and $\{i\}^{W_e}$ is a genericity requirement, then G meets $\{i\}^{W_e}$,

Q_s**:** If φ_s is a partial genericity requirement, then G meets φ_s.

The first set of requirements $\{R_{\langle e,i\rangle}\}$ will guarantee that G is $aw2$-generic; the second set $\{Q_s\}$ will guarantee that G is 1-generic. The requirements are ranked in descending order of priority as follows:

$$Q_0, R_0, Q_1, R_1, Q_2, R_2, \ldots.$$

By equation (1), it suffices to make sure that $G \oplus K \equiv_{\mathrm{T}} S$. As before, no requirement is injured, and each requirement is acted upon at most once, after which it is satisfied forever.

Stage 0**:** $\gamma_0 \stackrel{\mathrm{df}}{=} \emptyset$. $u_0 \stackrel{\mathrm{df}}{=} 0$.
End of Stage 0**.**

Stage $s+1$**:** We are given γ_s and u_s.

Step 1: Let $u_{s+1} \stackrel{\mathrm{df}}{=} 1 + \max\{(\mu z)[K_z \restriction s = K \restriction s], u_s, m\}$, where

$$m \stackrel{\mathrm{df}}{=} \max\{|\varphi_j(\sigma)| : j \leq s \ \ \& \ \ |\sigma| \leq u_s \ \ \& \ \ \varphi_j(\sigma)\!\downarrow \succeq \sigma\}.$$

For $j \leq s$, we say that requirement Q_j is *hungry* if Q_j has not yet been acted upon and $\varphi_j(\gamma_s)\!\downarrow \succeq \gamma_s$.

Step 2: For all $\langle e, i\rangle \leq s$, say that requirement $R_{\langle e,i\rangle}$ is *hungry* if $R_{\langle e,i\rangle}$ is not yet acted upon and

1. $\{i\}^{W_e}(\gamma_s)$ halts in no more than u_{s+1} steps,
2. $\gamma_s \preceq \{i\}^{W_e}(\gamma_s)$, and
3. $|\{i\}^{W_e}(\gamma_s)| \leq u_{s+1} - 1$.

Step 3: If there are no hungry requirements, we set $\gamma_{s+1} \stackrel{\text{df}}{=} \gamma_s \hat{} S(s)$. Otherwise, we act upon the hungry requirement of highest priority as follows: if Q_j is the highest priority hungry requirement, we set $\gamma_{s+1} \stackrel{\text{df}}{=} \varphi_j(\gamma_s) \hat{} S(s)$; if $R_{\langle e,i \rangle}$ is the highest priority hungry requirement, we set $\gamma_{s+1} \stackrel{\text{df}}{=} \left[\{i\}^{W_e}(\gamma_s)\right] \hat{} S(s)$.

End of Stage $s+1$.

The function $u \stackrel{\text{df}}{=} \lambda s.u_{s+1}$ serves the same purpose as the function t in the proof of Theorem 14. Its definition is complicated slightly by our wish to maintain the following invariant for all s, which can be shown easily by induction:

$$|\gamma_s| \leq u_s.$$

Note that u dominates t, so $K \leq_{\mathrm{T}} u$. Conversely, it is clear that $u \leq_{\mathrm{T}} K$: the function u is defined independently from the set S. Thus $u \equiv_{\mathrm{T}} K$.

Since $K \leq_{\mathrm{T}} S$, the entire construction is evidently recursive in S, hence $G \oplus K \leq_{\mathrm{T}} S$. On the other hand, the construction is also recursive in $G \oplus K$: given γ_s and u_s, we use K to compute u_{s+1} and to determine which requirement, if any, is acted upon at stage $s+1$; we use G to determine the last digit of γ_{s+1}, which encodes $S(s)$. Therefore, $G \oplus K \equiv_{\mathrm{T}} S$. Since G will be shown to be 1-generic, by equation (1) we have $G' \equiv_{\mathrm{T}} S$ as well.

It remains to show that all requirements are satisfied. Suppose this is not the case. We have two possibilities:

Case 1: The highest priority unsatisfied requirement is Q_j. Let s_0 be a stage beyond which no higher priority requirement is acted upon. Since Q_j is not acted upon at stage s_0+1, it must be because either $\varphi_j(\gamma_{s_0}) \uparrow$ or $\varphi_j(\gamma_{s_0}) \not\succeq \gamma_{s_0}$. In either case, Q_j is satisfied. Contradiction.

Case 2: The highest priority unsatisfied requirement is $R_{\langle e,i \rangle}$. It must be that $\{i\}^{W_e}$ is a genericity requirement with W_e incomplete, otherwise $R_{\langle e,i \rangle}$ is satisfied vacuously. We describe a W_e-recursive function which dominates u and thus dominates t. This implies $K \leq_{\mathrm{T}} W_e$, which contradicts the fact that W_e is incomplete.

Let s_0 be a stage beyond which no higher priority requirement is acted upon. For all $n \in \omega$ define

$$g(n) \stackrel{\text{df}}{=} \max(\ell + 1, r),$$

where

$$\ell \stackrel{\text{df}}{=} \max_{|\sigma| \leq n} |\{i\}^{W_e}(\sigma)|,$$

and

$$r \stackrel{\text{df}}{=} \max_{|\sigma| \leq n} [\text{running time of } \{i\}^{W_e}(\sigma)].$$

Note that g is total, W_e-recursive, and nondecreasing. Since $R_{\langle e,i\rangle}$ is not acted upon at any stage $s+1 > s_0$, conditions (1) and (3) cannot both hold in step 2 of stage $s+1$. Thus $g(u_s) > u_{s+1}$, since $|\gamma_s| \leq u_s$. Define

$$\begin{aligned} f(k) &\stackrel{\text{df}}{=} u_k \text{ for all } k < s_0, \\ f(s_0) &\stackrel{\text{df}}{=} u_{s_0}, \\ f(s+1) &\stackrel{\text{df}}{=} g(f(s)) \text{ for all } s \geq s_0. \end{aligned}$$

Clearly, f is W_e-recursive, and $f(s) \geq u_s$ for all $s \geq s_0$. Thus $f(s) \geq u_s$ for all $s \in \omega$, and so the function $\lambda s.f(s+1)$ dominates u and is W_e-recursive.

□

Remark: In the case where $\mathbf{d} = \mathbf{0}'$, the degree of the set G constructed in the proof above, as well as being low, is incomparable with every r.e. degree except $\mathbf{0}$ and $\mathbf{0}'$. This follows immediately by Proposition 13 and the fact that there is no nonzero r.e. degree below a 1-generic degree.

8 Further Research

We have shown that the notions of $aw2$-genericity and 1-genericity are of incomparable strength. It is interesting to compare the properties of $aw2$-generics with those of 1-generics and other sets of comparable genericity. For example, can an $aw2$-generic degree form a minimal coverΓ For another example, Martin (see [19]) showed that 2-generic degrees cannot bound minimal degrees. More recently, Chong & Downey [7] and Kumabe [21] independently constructed a 1-generic degree bounding a minimal degree. There are $aw2$-generic degrees bounding minimal degrees, trivially because *every* degree is bounded by an $aw2$-generic degree by Theorem 14. Do there exist $aw2$-generic degrees which are themselves minimalΓ We suspect not, despite the evidence suggested by Theorem 16 about m-degrees. More generally, what can we say about the structure of the Turing degrees below an $aw2$-generic degreeΓ

As was mentioned in Section 4, almost nothing is known about sets that are rec($\mathbf{d}$)-generic for all $\mathbf{d} < \mathbf{0}'$, not necessarily r.e. In particular, are any of these sets not weakly 2-genericΓ It is easy to see that none of these sets can exist strictly below $\mathbf{0}'$, so by Theorem 19 there are $aw2$-generic sets which do not fit this description.

Is the notion 'almost weakly n-generic' useful for $n > 2$Γ

Acknowledgments

I would like to thank my thesis advisor, Stuart Kurtz, for his astute guidance and inspiration, as well as for suggesting that Theorem 16 is true and providing the key insight into its proof.

I am also grateful to Richard Shore for communicating Theorem 19, Lance Fortnow for many good conversations and for proving an earlier result similar to Theorem 8, and to James Foster for helpful discussions and correspondence. Finally, I wish to thank William Gasarch for his helpful comments on earlier drafts of this paper.

A Analogues to Theorem 4

The techniques used by Blum & Impagliazzo to prove Theorem 4 can be adapted to prove a number of similar results. A general study of these and other techniques can be found in [12]. We list most of the known results here; in the listing, G can be any 1-generic set, and SAT is the **NP**-complete set of satisfiable Boolean formulae. Definitions of **UP** and **FewP** can be found, for example, in [15] and [1] respectively. The class **SPP** is defined in [11] and in [27] under the name **XP**. The class **BPP** is well-studied; see [4] for example.

- $\mathbf{NP}^G \cap \text{co-}\mathbf{NP}^G \subseteq \mathbf{P}^{G \oplus \mathrm{SAT}}$ [6].
- $\mathbf{UP}^G \subseteq \mathbf{P}^{G \oplus \mathrm{SAT}}$ [6].
- $\mathbf{FewP} \subseteq \mathbf{P}^{G \oplus \mathrm{SAT}}$ [12].
- Every pair of disjoint $\mathbf{NP}^G$ sets is $\mathbf{P}^{G \oplus \mathrm{SAT}}$-separable [6, 9].
- $\mathbf{BPP}^G \subseteq \mathbf{P}^{G \oplus C}$, where C is any Σ_2^p-complete set [18] ($\Sigma_2^p \stackrel{\mathrm{df}}{=} \mathbf{NP}^{\mathbf{NP}}$).
- $\mathbf{SPP}^G \subseteq \mathbf{P}^{G \oplus E}$, where E is any set complete for **PSPACE** [12].

References

[1] E. W. Allender. The complexity of sparse sets in P. In *Structure in Complexity Theory*, *Lecture Notes in Computer Science*, vol. 223, Springer-Verlag, 1986, pages 1–11.

[2] K. Ambos-Spies, H. Fleischhack, and H. Huwig. P-generic sets. In *Proceedings of the 11th International Colloquium on Automata, Languages, and Programming* (Paredaens, editor), *Lecture Notes in Computer Science*, vol. 172, Springer-Verlag, 1984, pages 58–68.

[3] K. Ambos-Spies, H. Fleischhack, and H. Huwig. Diagonalizations over polynomial time computable sets. *Theoretical Computer Science*, 51:177–204, 1987.

[4] J. L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I*, volume 11 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1988.

[5] M. Blum. A machine-independent theory of the complexity of recursive functions. *Journal of the ACM*, 14(2):322–336, 1967.

[6] M. Blum and R. Impagliazzo. Generic oracles and oracle classes. In *Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science*, pages 118–126, 1987.

[7] C. T. Chong and R. G. Downey. Minimal degrees recursive in 1-generic degrees. *Annals of Pure and Applied Logic*, 48:215–225, 1990.

[8] S. Fenner. Notions of resource-bounded category and genericity. Technical Report 90-32, Department of Computer Science, University of Chicago, 1990.

[9] S. Fenner. Notions of resource-bounded category and genericity. In *Proceedings of the 6th Annual IEEE Structure in Complexity Theory Conference*, pages 196–212, 1991. Journal version in preparation.

[10] S. Fenner. Tight lower bounds on genericity required to prevent one-way functions. Technical Report 91-04, Department of Computer Science, University of Chicago, 1991.

[11] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48:116–148, 1994.

[12] S. Fenner, L. Fortnow, S. Kurtz, and L. Li. An oracle builder's toolkit. In *Proceedings of the 8th IEEE Structure in Complexity Theory Conference*, pages 120–131, 1993.

[13] J. A. Foster. *Forcing and Genericity on the Polynomial Hierarchy.* PhD thesis, Illinois Institute of Technology, 1990.

[14] R. M. Friedberg. A criterion for completeness of degrees of unsolvability. *Journal of Symbolic Logic*, 22:159–160, 1957.

[15] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17:309–335, 1988.

[16] P. G. Hinman. Some applications of forcing to hierarchy problems in arithmetic. *Z. Math. Logik Grundlagen Math*, 15:341–352, 1969.

[17] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[18] R. Impagliazzo and M. Naor. Decision trees and downward closures. In *Proceedings of the 3rd IEEE Structure in Complexity Theory Conference*, pages 29–38, 1988.

[19] C. G. Jockusch. Degrees of generic sets. In F. R. Drake and S. S. Wainer, editors, *Recursion Theory: Its Generalizations and Applications*, pages 110–139. Cambridge University Press, 1980.

[20] C. G. Jockusch and D. B. Posner. Double jumps of minimal degrees. *Journal of Symbolic Logic*, 43:715–724, 1978.

[21] M. Kumabe. A 1-generic degree which bounds a minimal degree. *Journal of Symbolic Logic*, 55(2):733–743, 1990.

[22] S. A. Kurtz. Notions of weak genericity. *Journal of Symbolic Logic*, 48(3):764–770, September 1983.

[23] J. H. Lutz. Resource-bounded Baire category and small circuits in exponential space. In *Proceedings of the 2nd Annual IEEE Structure in Complexity Theory Conference*, pages 81–91, 1987.

[24] J. H. Lutz. Almost everywhere high nonuniform complexity. In *Proceedings of the 4th Annual IEEE Structure in Complexity Theory Conference*, pages 37–53, 1989. An updated version appears as Iowa State University Computer Science Department Technical Report #91-18.

[25] J. H. Lutz. Category and measure in complexity classes. *SIAM Journal on Computing*, 19(6):1100–1131, December 1990.

[26] E. Mayordomo. Almost every set in exponential time is P-bi-immune. Unpublished manuscript, 1991.

[27] M. Ogiwara and L. A. Hemachandra. A complexity theory of feasible closure properties. In *Proceedings of the 6th Annual IEEE Structure in Complexity Theory Conference*, pages 16–29, 1991.

[28] B. Poizat. Q = NQΓ *Journal of Symbolic Logic*, 51:22–32, 1986.

[29] H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, 1967. Reprinted. MIT Press. 1987.

[30] R. A. Shore, 1991. Private communication.

[31] R. I. Soare. *Recursively Enumerable Sets and Degrees.* Springer-Verlag, 1987.