Notions of resource-bounded category and genericity *Preliminary Report*

Stephen A. Fenner
University of Chicago

January 23, 1991

Abstract

Generic sets, like random sets, have proved useful in providing a coherent picture of relativized computation. Unfortunately, they are not decidable. Lutz [Lut87, Lut89, Lut90] has studied the notions of resource-bounded measure and category, and has shown that there are decidable sets similar to generic sets ("pseudogeneric sets") which are of reasonably low complexity. Lutz's original definition has drawbacks, however: pseudogenerics (in the sense of Lutz) are often not strong enough even to ensure basic oracle separations, such as $\mathbf{P} \neq \mathbf{NP}$. We modify Lutz's definition to create much more powerful pseudogenerics that are better at performing these oracle separations, and we further show that these sets exist in the same complexity classes as with Lutz's definition.

On the negative side, we show that results of Blum and Impagliazzo regarding $\mathbf{NP} \cap$ co-$\mathbf{NP}$ and one-way functions, which hold for true generics, cannot be guaranteed even by very powerful pseudogenerics. We thus isolate a crucial difference between generic and pseudogeneric sets.

1 Introduction

Both Lebesgue measure and Baire category provide useful and differing notions of "smallness" for a set of reals. In Lebesgue measure, the small sets are those with measure zero (null sets). In Baire category, the small sets are those of first category (meager sets). Both notions of smallness satisfy the following basic properties [Oxt80].

1. All countable sets are small.

2. Any subset of a small set is small.

3. The countable union of small sets is small.

4. The complement of a small set is dense.

We say that a set of reals is "large" if its complement is small. A real number X is "typical" with respect to a countable collection Γ of large sets if $X \in \cap\Gamma$. By properties 3 and 4 above, the reals typical for any fixed Γ comprise a dense set. In Lebesgue measure, typical reals are called "random;" in Baire category, they are called "generic."

A recursive notion of category was introduced by Mehlhorn in 1973, and was also investigated by Lisagor in 1979 (see [Lut90]). In a series of seminal papers, Lutz has introduced resource-bounded versions of both measure and category, then studied the notions of typicality that result. The chief advantage of these ideas is that the typical sets now exist in subrecursive complexity classes, and one can then study the typical sets for a given complexity class. Lutz has concentrated mostly on the measure theoretic side; in the present paper, we are interested in the category side. In particular, we wish to study the power of resource-bounded generics—compared to full generics—in determining certain questions of relativized resource-bounded computation. We will show that generic oracles produced by Lutz's original notion of polynomial time-bounded category are not strong enough to separate **NP** from **P**, but that minor modifications of his idea are indeed adequate.

Minor adjustments of this sort are often needed when a recursion theoretic notion is borrowed for use in complexity theory. For example, the distinction between unary and binary representations for input—irrelevant in recursion theory—becomes crucial in complexity theory. Another example is the oracle Turing machine model (compare [Soa87], pp. 46–47 with [BGS75]). One can make a number of different changes, with varying strengths, to Lutz's definition of category while still adhering to the spirit of Lutz's original ideas. In the present paper we concentrate on one such improvement: we define category and genericity not in terms of strings (which is good enough for recursion theory), but rather in terms of finite functions. The two ideas are equivalent in the classical and recursion theoretic settings, but are quite different in the context of complexity theory.

We show that polynomial time-bounded generics (in our sense) are strong enough to separate **P** from **NP**, and that these generics exist in exponential time. Thus the typical exponential time set (in the sense of polynomial time-bounded category) separates **P** and **NP**. This is not true for Lutz's original definition of resource-bounded category based on strings.

We also prove a negative result. In [BI87], Blum and Impagliazzo show that if **P** = **NP** (unrelativized), then with respect to a (full) generic set, **P** = **NP** $\cap$ co-**NP** and **P** = **UP**. We show that the degree of genericity needed for these results is essentially tight; no reasonable notion of resource-bounded genericity is strong enough to guarantee either **P** = **NP** $\cap$ co-**NP** or **P** = **UP**, regardless of whether **P** = **NP** in the unrelativized case.

2 Preliminaries

We let ω be the set of natural numbers. We define a *finite function* to be a partial function from ω to $\{0,1\}$ whose domain is finite. A *binary string* is a finite function whose domain is an initial segment of ω. We use λ to denote the empty function. Departing from custom, we let $2^{<\omega}$ denote the set of all finite functions (others reserve $2^{<\omega}$ for the set of binary strings). We return to custom by letting 2^{ω} denote the set of all total functions from ω to $\{0,1\}$, which we also call *real numbers*. We identify a subset of ω with its characteristic function in 2^{ω}. We use $X, Y, Z, \ldots$ to denote sets of reals, $A, B, C, \ldots$ to denote subsets of ω (real numbers), lower case Roman letters to denote natural numbers, and lower case Greek letters to denote finite functions. We sometimes identify ω with $\{0,1\}^*$ via the usual dyadic representation, and if x is a natural number, we let $|x|$ be the length of the dyadic representation of x. If f is any function, we use $\mathrm{dom}(f)$ and $\mathrm{rng}(f)$ to denote the domain and range of f, respectively.

Let σ be a finite function. If x is a natural number, we write $\sigma(x)\downarrow$ to mean $x \in \mathrm{dom}(\sigma)$, and we write $\sigma(x)\uparrow$ to mean $x \notin \mathrm{dom}(\sigma)$. We let $||\sigma||$ denote the cardinality of $\mathrm{dom}(\sigma)$, we let $\min(\sigma)$ denote the smallest x *not* in $\mathrm{dom}(\sigma)$, and we let $\max(\sigma)$ denote the smallest x such that $(\forall y \geq x)\sigma(y)\uparrow$. If $b \in \{0,1\}$, we write $\sigma\widehat{\ }b$ to mean $\sigma \cup \{(\min(\sigma), b)\}$. If f is any partial or total function from ω to $\{0,1\}$, we write $\sigma \subseteq f$ to mean, "σ is extended by f," and we write $\sigma \subset f$ to mean, "σ is properly extended by f." Let $B_\sigma = \{A \in 2^{\omega} \mid \sigma \subset A\}$ (we assume the Cantor topology on 2^{ω}, for which the B_σ form a basis). If τ is a finite function, we say σ and τ are *compatible* if $\sigma(x) = \tau(x)$ for all $x \in \mathrm{dom}(\sigma) \cap \mathrm{dom}(\tau)$. It is useful to define a "fill-with-zeros" operator: for any $\sigma \in 2^{<\omega}$ and any $k \in \omega$, define $\mathrm{zfill}(\sigma, k)$ to be the unique $\tau \supseteq \sigma$ such that $\mathrm{dom}(\tau) = \mathrm{dom}(\sigma) \cup \{0, \ldots, k-1\}$ and $\tau(i) = 0$ for all $i < k$ such that $\sigma(i)\uparrow$. In other words, $\mathrm{zfill}(\sigma, k)$ is the result of filling any gaps in σ with zeros up to the kth position. Finally, we will sometimes have occasion to use a finite function σ as an oracle, in which case we interpret σ as the set $\{x \mid \sigma(x)\downarrow = 1\}$.

We fix a bijection $\langle\cdot,\cdot\rangle: \omega \times \omega \to \omega$ that is computable in polynomial time and polynomial time invertible. For convenience we further assume that $\langle\cdot,\cdot\rangle$ is a bijection on the set of numbers of the form 0^k for $k \in \omega$ (numbers whose dyadic representation consists solely of 0's). If f is a function with domain ω and k is a natural number, we define $f_k(x) = f(\langle 0^k, x\rangle)$ for all $x \in \omega$.

The central concept in the study of resource-bounded category is that of a finite extension strategy.

Definition 1 *A* finite extension strategy *is a function $h: 2^{<\omega} \to 2^{<\omega}$ such that $\sigma \subseteq h(\sigma)$ for all $\sigma \in 2^{<\omega}$. A finite function τ* meets *h if there is a $\sigma \in 2^{<\omega}$ such that $h(\sigma) \subseteq \tau$. If A is a real number, we say that A* meets *h if some $\tau \subset A$ meets h. The set A* avoids *h if A does not meet h.*

For brevity, we will use the word "strategy" to refer to a finite extension strategy. The following definitions are adapted from classical point-set topology.

Definition 2 *A set X of reals is* nowhere dense *if there exists a strategy h such that $X \cap B_{h(\sigma)} = \emptyset$ for all $\sigma \in 2^{<\omega}$. Equivalently, X is nowhere dense if and only if there is a single strategy that is avoided by every element of X.*

Definition 3 *A set X of reals is* meager *(of* first category*) if X is contained in a countable union of nowhere dense sets. Equivalently, X is meager if and only if there is a countable family $h_0, h_1, h_2, \ldots$ of strategies such that every $A \in X$ avoids one or more of the h_i's.*

2.1 Encodings of Finite Functions

We are interested in resource-bounded versions of these notions. We do this by restricting the resources allowed by a Turing machine to compute finite extension strategies. The remaining ambiguity is how we encode finite functions as inputs and outputs to a machine. We consider two encodings:

String Encode a finite function σ as a string of symbols from the alphabet $\{0, 1, \uparrow\}$, where the $(i+1)$st symbol (0, 1, or $\uparrow$) of the encoding determines the value of $\sigma(i)$ (0, 1, or undefined, respectively). For uniqueness, we require that the encoding not have $\uparrow$ as its last symbol.

List-of-pairs Identify a finite function with its graph and encode it as a list of ordered pairs, using any fixed reasonable encoding scheme over a finite alphabet Σ.

We also assume that finite functions encoded either by the Strings or List-of-pairs scheme are further encoded as natural numbers. We thus have the following diagram:

$$\begin{tikzcd} & 2^{<\omega} \arrow[dl] \arrow[dr] & \\ \{0,1,\uparrow\}^* \arrow[dr] & & \Sigma^* \arrow[dl] \\ & \omega & \end{tikzcd}$$

We denote the Strings encoding (following the left path) as $\mathrm{St}: 2^{<\omega} \to \omega$, and the List-of-pairs encoding (following the right path) as $\mathrm{Lp}: 2^{<\omega} \to \omega$. Both encodings can also be assumed to be bijections and still be "reasonable" in the sense that the basic operations on finite functions—$\sigma(x)$, $||\sigma||$, $\min(\sigma)$, $\max(\sigma)$, union of compatible functions, etc.—are all computable in polynomial time. Furthermore, it is natural to require that $|\mathrm{St}(\sigma)|$ be polynomially related to $\max(\sigma)$, and that $|\mathrm{Lp}(\sigma)|$ be polynomially related to $\|\sigma\| + |\max(\sigma)|$. We fix once and for all a nondecreasing polynomial q_0 such that for all $\sigma \in 2^{<\omega}$,

$$\begin{aligned}
|\mathrm{St}(\sigma)| &\leq q_0(\max(\sigma)) && (1)\\
\max(\sigma) &\leq q_0(|\mathrm{St}(\sigma)|) && (2)\\
|\mathrm{Lp}(\sigma)| &\leq q_0(\|\sigma\| + |\max(\sigma)|) && (3)\\
\|\sigma\| + |\max(\sigma)| &\leq q_0(|\mathrm{Lp}(\sigma)|). && (4)
\end{aligned}$$

We allow inputs and outputs of a strategy to be encoded independently of each other, possibly by different encodings. If each of e_1 and e_2 is one of the above encodings and $h: 2^{<\omega} \to 2^{<\omega}$ is a strategy, we call the map $e_2 \circ h \circ e_1^{-1}: \omega \to \omega$, the e_1, e_2-*encoding* of the strategy h. In other words, the e_1, e_2-encoding of h (mapping numbers to numbers) represents h where the inputs are encoded by e_1 and the outputs are encoded by e_2. In the present paper we will limit our attention to three encoding combinations:

A. St, St

B. Lp, Lp

C. St, Lp.

We will see in section 4 that for $\Delta = \mathrm{p}_2$, (A) and (B) are of incomparable strength, and (C) is strictly stronger than both.

2.2 Function Classes

The following definitions are adapted from [Lut89]. If $P(x)$ is a predicate, we write $P(x)$ a.e. to mean "$P(x)$ holds for all but finitely many $x \in \omega$," and $P(x)$ i.o. to mean "$P(x)$ holds for infinitely many $x \in \omega$." We define a class of growth rates $\{G_i\}_{i \in \omega}$ as follows:

$$\begin{aligned}
G_0 &= \{f \mid (\exists k) f(n) \leq kn \text{ a.e. }\}\\
G_{i+1} &= \{f \mid (\exists g \in G_i) f(n) \leq 2^{g(\log n)} \text{ a.e. }\}
\end{aligned}$$

It is easy to show that all the G_i are closed under composition, and only contain functions of subexponential growth. Our underlying model of computation will be the deterministic Turing Machine, but all our results will hold with respect to any other reasonable model of uniform computation. We are interested in the following function classes:

$$\begin{aligned}
\omega^\omega &= \{f \mid f: \omega \to \omega\}\\
\mathrm{rec} &= \{f \in \omega^\omega \mid f \text{ is recursive}\}\\
\mathrm{p}_i &= \{f \in \omega^\omega \mid f \in \mathbf{DTIME}(G_i)\} \ \ (i \geq 1)\\
\mathrm{pspace}_i &= \{f \in \omega^\omega \mid f \in \mathbf{DSPACE}(G_i)\} \ \ (i \geq 1)
\end{aligned}$$

We write p for p_1 and pspace for pspace_1. It is well-known that for every i, p_{i+1} and pspace_{i+1} contain universal functions for p_i and pspace_i respectively, where a universal function for a class Δ is a function g such that $\Delta = \{g_i \mid i \in \omega\}$. From now on in this paper, unless stated otherwise, Δ and Δ' will be either ω^ω, rec, p_i for $i \geq 1$, or pspace_i for $i \geq 1$.

We now define meagerness with respect to Δ.

Definition 4 *Let e_1 and e_2 be encodings taken from the two possibilities described above. A set $X \subseteq 2^\omega$ is* Δ^{e_1,e_2}-meager *if there is a function $h \in \Delta$ such that*

- *For all $i \in \omega$, h_i is the e_1, e_2-encoding of some strategy g_i.*
- *For every $A \in X$, there exists a $k \in \omega$ such that A avoids g_k.*

A set X is Δ^{e_1,e_2}-comeager *if $2^\omega - X$ is Δ^{e_1,e_2}-meager.*

If $\Delta = \omega^\omega$, then definition 4 coincides with definition 3. If we take $e_1 = e_2 = \mathrm{St}$, we get an equivalent formulation of Lutz's original definition of Δ-meagerness in [Lut87, Lut90]. (There, Lutz defines Δ-meagerness using binary strings rather than finite functions.) For $\Delta = \mathrm{rec}$, the choice of encodings does not matter. In section 4, however, we show that for $\Delta = \mathrm{p}_2$, the choice of encodings does matter: there are $\mathrm{p}_2^{\mathrm{Lp,Lp}}$-meager sets which are not $\mathrm{p}_2^{\mathrm{St,St}}$-meager and vice versa, and there are $\mathrm{p}_2^{\mathrm{St,Lp}}$-meager sets which are neither $\mathrm{p}_2^{\mathrm{St,St}}$-meager nor $\mathrm{p}_2^{\mathrm{Lp,Lp}}$-meager. These results can also be extended to p_i for $i \geq 2$ as well.[1] In any event, the notion of $\Delta^{\mathrm{St,Lp}}$-meagerness is no less inclusive than the other two.

Proposition 5 *For any $X \subseteq 2^\omega$, if X is $\Delta^{\mathrm{St,St}}$-meager or $\Delta^{\mathrm{Lp,Lp}}$-meager, then X is $\Delta^{\mathrm{St,Lp}}$-meager.*

Proof. Note that the function $\mathrm{Lp} \circ \mathrm{St}^{-1}$ is computable in polynomial time. If X is $\Delta^{\mathrm{St,St}}$-meager as witnessed by $h \in \Delta$, then the function $\mathrm{Lp} \circ \mathrm{St}^{-1} \circ h$ witnesses that X is $\Delta^{\mathrm{St,Lp}}$-meager. Likewise, the function

$$d(\langle x, y\rangle) = \langle x, \mathrm{Lp}(\mathrm{St}^{-1}(y))\rangle$$

is computable in polynomial time. If X is $\Delta^{\mathrm{Lp,Lp}}$-meager as witnessed by $h \in \Delta$, then X is $\Delta^{\mathrm{St,Lp}}$-meager witnessed by $h \circ d \in \Delta$. ▯

We now define resource-bounded genericity for countable classes Δ.

Definition 6 *Let Δ be a countable function class, and let $e_1, e_2 \colon 2^{<\omega} \to \omega$ be encodings. A real number $G \in 2^\omega$ is* Δ^{e_1,e_2}-generic *if G is an element of every Δ^{e_1,e_2}-comeager set (equivalently, the singleton $\{G\}$ is not Δ^{e_1,e_2}-meager).*

3 Basic properties of Δ^{e_1,e_2}-meager sets

In this section we present properties of Δ^{e_1,e_2}-meager sets analogous to those of classical meager sets mentioned in the introduction. These results generally follow Lutz [Lut87, Lut90] with some occasional alterations. These results also hold for all combinations of the String and List-of-pairs encodings except Lp, St, so in this section only we will fix implicit encodings e_1 and e_2 ($e_1 \neq \mathrm{Lp}$ or $e_2 \neq \mathrm{St}$), and identify strategies g with their e_1, e_2-encodings. We will also identify $2^{<\omega}$ and ω via e_1 for *inputs* to strategies, and we will use e_2 to make the identification for *outputs*. We will also dispense with the superscripts on Δ.

3.1 Resource-bounded Baire category theorem

A *constructor* is a finite extension strategy $\delta \in \Delta$ such that for all binary strings σ, $\delta(\sigma) = \sigma\widehat{\ }b$, where $b \in \{0,1\}$. Given a binary string ι, define $R_\iota(\delta)$ to be the unique real number extending $\delta^i(\iota)$ for all $i \in \omega$. Define $R(\delta)$ to be $R_\lambda(\delta)$, and let $R(\Delta) = \{R(\delta) \mid \delta \in \Delta\}$. Notice that $\iota \subset R_\iota(\delta)$ and that

$$R(\Delta) = \bigcup_{\text{all binary strings } \iota} \{R_\iota(\delta) \mid \delta \in \Delta\}.$$

[1]The fourth possible combination, Lp, St, will not be considered. For the time-bounded classes Δ, even the Lp, St-encoding of the identity function $\mathrm{id}\colon 2^{<\omega} \to 2^{<\omega}$ is not computable in Δ, thus no sets are $\Delta^{\mathrm{Lp,St}}$-meager.

Lemma 7 ([Lut87, Lut89])

1. *$R(\omega^\omega) = 2^\omega$.*
2. *$R(\text{rec}) = \mathbf{REC} =_{\text{df}} \{A \subseteq \omega \mid A$ is recursive$\}$.*
3. *For $i \geq 1$, $R(\text{p}_i) = \mathbf{E}_i =_{\text{df}} \mathbf{DTIME}(2^{G_{i-1}})$.*
4. *For $i \geq 1$, $R(\text{pspace}_i) = \mathbf{ESPACE}_i =_{\text{df}} \mathbf{DSPACE}(2^{G_{i-1}})$.*

Definition 8 ([Lut87, Lut89])

- *A subset $X \subseteq R(\Delta)$ is* Δ-countable *if there is a function $\delta \in \Delta$ such that δ_k is a constructor for each $k \in \omega$ and $X \subseteq \{R(\delta_k) \mid k \in \omega\}$.*
- *A Δ-union of Δ-meager sets is a subset $X \subseteq 2^\omega$ such that $X = \bigcup_{i\in\omega} X_i$ and there is an $h \in \Delta$ such that h_i witnesses the Δ-meagerness of X_i for each i.*

Theorem 9

1. *All Δ-countable sets are Δ-meager.*
2. *All subsets of a Δ-meager set are Δ-meager.*
3. *The Δ-union of Δ-meager sets is Δ-meager.*

Proof.

1. Let $X \subseteq R(\Delta)$ be Δ-countable as witnessed by the function $\delta \in \Delta$. Let h be a function such that for all $i \in \omega$ and $\sigma \in 2^{<\omega}$,

$$h_i(\sigma) = \sigma\hat{}\,[1 - [R(\delta_i)](\min(\sigma))]\,.$$

   We can compute $[R(\delta_i)](\min(\sigma))$ by composing δ_i with itself $\min(\sigma) + 1$ times starting with the empty function λ, then taking the result on argument $\min(\sigma)$. It follows that h is in Δ. Also, $R(\delta_i)$ avoids h_i for all $i \in \omega$. Thus X is Δ-meager as witnessed by h.

2. Immediate by the definition.

3. Let $X = \bigcup_{i\in\omega} X_i$ and $h \in \Delta$ such that X_i is Δ-meager witnessed by h_i for all $i \in \omega$. Define g so that

$$g(\langle\langle x, y\rangle, z\rangle) = h(\langle x, \langle y, z\rangle\rangle)$$

   for all $x, y, z \in \omega$. Clearly, $g \in \Delta$, and g_k is a strategy for all k. Given $A \in X$, choose i_0 such that $A \in X_{i_0}$. Then X_{i_0} is Δ-meager via h_{i_0}, so there is a j_0 such that A avoids $h_{i_0 j_0} = g_{\langle i_0, j_0\rangle}$. Thus g witnesses that X is Δ-meager.

$\square$

We now prove the resource-bounded version of the Baire category theorem.

Theorem 10 *If $X \subseteq 2^\omega$ is Δ-meager, then $R(\Delta) - X$ is dense.*

Proof. Given $h \in \Delta$ witnessing that X is Δ-meager, we describe a constructor $\delta \in \Delta$ such that for any initial binary string ι, $R_\iota(\delta) \notin X$. For this, it suffices to build δ so that $R_\iota(\delta)$ meets all of $h_0, h_1, h_2, \ldots$, that is, for every $n \in \omega$ there exists a binary string σ such that $h_n(\sigma) \subset R_\iota(\delta)$. The constructor δ is computed by the following algorithm:

1. Given as input a binary string σ of length ℓ, compute

$$n_0 \leftarrow \min_n [\, n \leq \ell \ \& \ (\forall \text{ binary strings } \alpha \subseteq \sigma)\ h_n(\alpha) \not\subseteq \sigma \,].$$

   (h_{n_0} is the next function that $R(\delta)$ must meet.)

2. If no such n_0 exists, return $\sigma\hat{\ }0$.

3. Otherwise, find the shortest binary string $\tau \subseteq \sigma$ such that σ and $h_{n_0}(\tau)$ are compatible.

4. Let $\gamma \leftarrow h_{n_0}(\tau)$. If $\gamma(\ell)\downarrow$, return $\sigma\hat{\ }\gamma(\ell)$; otherwise return $\sigma\hat{\ }0$.

It is clear that $h \in \Delta \implies \delta \in \Delta$ for Δ being any of the classes that we are considering. A straightforward induction argument shows that $R_\iota(\delta)$ meets all the h_n. ▯

Lemma 11 ([Lut87]) *If X is Δ-meager, and $\Delta \subseteq \Delta'$, then X is Δ'-meager.*

Proof. Immediate. ▯

3.2 Resource-bounded Banach-Mazur games

Meager sets can be alternatively defined in terms of winning strategies of Banach-Mazur games (see [Oxt80]). An effective version of the Banach-Mazur game was introduced by Lisagor in 1979 (see [Lut90]). Lutz [Lut87, Lut90] has shown that if Δ is one of the space-bounded classes, then Δ-meager sets can likewise be characterized in terms of Banach-Mazur games where the second player must follow a strategy based on an algorithm in Δ. We extend this equivalence to the time-bounded classes as well. Thus resource-bounded category and resource-bounded Banach-Mazur games are equivalent for all of the classes we are considering. Banach-Mazur games are extremely useful in studying resource-bounded category because they allow us to work with single strategies rather than uniform families of them.

The following definitions are adapted from [Lut87] and [Lut90].

Definition 12 ([Lut90])

- *A* play *of a Banach-Mazur game is an ordered pair (f, g) of finite extension strategies such that* $\min(\sigma) \neq \min(g(\sigma))$ *for every $\sigma \in 2^{<\omega}$.*
- *The* result *of the play (f, g) is the unique real number $R(f, g)$ that extends $(g \circ f)^i(\lambda)$ for all $i \in \omega$.*

If $X \subseteq 2^\omega$ and $\Delta_{\mathrm{I}}, \Delta_{\mathrm{II}}$ are function classes, then $G[X; \Delta_{\mathrm{I}}, \Delta_{\mathrm{II}}]$ is the Banach-Mazur game with distinguished set X, where Player I must choose a finite extension strategy $f \in \Delta_{\mathrm{I}}$, and Player II must choose a finite extension strategy $g \in \Delta_{\mathrm{II}}$. We imagine a play as starting with the empty function λ, then players move alternately (starting with Player I), each applying her strategy to the result of the previous moves. Note that after each player plays s times, the resulting finite function is defined at least on all numbers $r < s$. Player I wins the play (f, g) if $R(f, g) \in X$; Player II wins otherwise. A *winning strategy* for Player II is a strategy $g \in \Delta_{\mathrm{II}}$ such that Player II wins (f, g) for every strategy $f \in \Delta_{\mathrm{I}}$.

Theorem 13 is a major improvement of theorem 4.3 of [Lut90] (also theorem 2 of [Lut87]). There, Lutz showed the equivalence under a strong hypothesis that does not appear to hold for the time-bounded classes (the Mazur property). We can avoid having to assume this hypothesis by allowing our requirements to be satisfied slowly.

Theorem 13 *Let X be a set of reals. The following are equivalent:*

1. *Player II has a winning strategy for $G[X; \omega^\omega, \Delta]$.*
2. *X is Δ-meager.*

Proof. ($1 \Longrightarrow 2$): The classical form of this implication was first proved by Banach in a different setting (see [Oxt80], pp. 27–30). The proof here uses the essential ideas of theorem 2 in [Lut87], adapted to handle finite functions instead of strings. Assume $g \in \Delta$ is a winning strategy for Player II. Define h so that, for all $k \in \omega$ and all $\sigma \in 2^{<\omega}$,

$$h_k(\sigma) = g(\mathrm{zfill}(\sigma, k)).$$

Clearly, $h \in \Delta$. Suppose $A \in 2^\omega$ meets h_k for all k. We show that $A \notin X$, which implies that X is Δ-meager as witnessed by h. For this, it suffices to show that for every $\sigma \subset A$ there is a τ such that $\sigma \subseteq \tau \subseteq g(\tau) \subset A$. If this is the case, then Player I has a strategy f to ensure that $R(f, g) = A$: given σ, Player I simply extends to a corresponding τ. We therefore must have $A \notin X$, otherwise g would not be a winning strategy for Player II.

Given $\sigma \subset A$, we find τ as follows: Let $k = \max(\sigma)$. We know that A meets h_k, which implies that there is a $\sigma' \subset A$ such that

$$\mathrm{zfill}(\sigma', k) \subseteq g(\mathrm{zfill}(\sigma', k)) = h_k(\sigma') \subset A.$$

Since $\mathrm{dom}(\sigma) \subseteq \mathrm{dom}(\mathrm{zfill}(\sigma', k))$, and both σ and $\mathrm{zfill}(\sigma', k)$ are extended by A, it must be that $\sigma \subseteq \mathrm{zfill}(\sigma', k)$. We can then take τ to be $\mathrm{zfill}(\sigma', k)$.

($2 \Longrightarrow 1$): Suppose X is Δ-meager as witnessed by $h \in \Delta$, that is, for every $A \in X$ there exists $i \in \omega$ such that A avoids h_i. We define a winning strategy $g \in \Delta$ for Player II in the game $G[X; \omega^\omega, \Delta]$. For this, it suffices to ensure that for any strategy $f \in \omega^\omega$, $R(f, g)$ meets all the h_i, and hence $R(f, g) \notin X$. It is no problem to force $R(f, g)$ to meet any given h_i just by letting g simulate h_i at some turn of the play. The problem is knowing which h_i to simulate at each turn so as to make sure that we eventually simulate all of them. Our advantage is that we can wait as long as we want before we choose which h_i to simulate.

Let $t\colon \omega \to \omega$ be a divergent function that grows sufficiently slowly and is computable in time polynomial in the *value* of its input. (For all the classes Δ that we are considering, the function

$$t(n) = \lfloor \log n \rfloor$$

is good enough.) The strategy g is computed as follows: given an input $\sigma \in 2^{<\omega}$,

1. $\sigma' \leftarrow \sigma\hat{\ }0$. (This is only to ensure that $\min(\sigma) \neq \min(g(\sigma))$.)

2. Compute
$$n_0 \leftarrow \min_n \left[\begin{array}{l} n < \|\sigma\| \ \ \& \\ (\forall \tau \in 2^{<\omega})[\max(\tau) \leq t(\|\sigma\|) \ \rightarrow \ h_n(\tau) \not\subseteq \sigma] \end{array} \right].$$
(h_{n_0} is the next strategy we must simulate.)

3. If no such n_0 exists, return σ'. Otherwise, return $h_{n_0}(\sigma')$.

Suppose there is a strategy $f \in \omega^\omega$ and a least k such that $R(f, g)$ avoids h_k. Then there is some finite point in the play (f, g) where

- it is Player II's turn,
- the play so far has met all the h_i for $i < k$, and
- Player II has the computational resources to recognize this fact.

In other words, there is an $s \in \omega$ such that (letting $\sigma = f((g \circ f)^s(\lambda))$),
$$k < \|\sigma\| \ \ \& \ \ (\forall i < k)(\exists \tau)[\max(\tau) \leq t(\|\sigma\|) \ \& \ h_i(\tau) \subseteq \sigma].$$
We then have $n_0 = k$ in step 2 of the computation of $g(\sigma)$, so $g(\sigma) = h_k(\sigma')$, where σ' is computed in step 1. Thus $R(f, g)$ meets h_k, contradicting the choice of k, so g is a winning strategy for Player II as long as $g \in \Delta$.

There are $3^{t(\|\sigma\|)}$ many τ with $\max(\tau) \leq t(\|\sigma\|)$. If t is given as above, this number is polynomial in $\|\sigma\|$. Thus in step 2 we only need to compute h at most a polynomial in $\|\sigma\|$ many times, with all inputs uniformly bounded by a polynomial in $\|\sigma\|$. From this it follows that $g \in \Delta$. ▯

3.3 Δ-generic sets

Lemma 14 *Let Δ be one of the countable classes. A real number G is Δ-generic if and only if G meets every strategy in Δ.*

Proof. Suppose $h \in \Delta$ is a strategy avoided by G. Then letting $f(\langle x, y\rangle) = h(y)$, we see that $f \in \Delta$ and G avoids f_i for all $i \in \omega$. Thus, $\{G\}$ is Δ-meager as witnessed by f, so G is not Δ-generic.

Conversely, if $\{G\}$ is Δ-meager as witnessed by $f \in \Delta$, then by definition there is an i such that G avoids the strategy $f_i \in \Delta$. Thus G does not meet every strategy in Δ. ▯

The next lemma describes relationships between resource-bounded category and genericity.

Lemma 15 *Suppose Δ' contains a universal function for Δ. Then for every $X \subseteq 2^\omega$ we have*

- *X is Δ-meager $\Longrightarrow$ X contains no Δ-generic sets.*
- *X contains no Δ-generic sets $\Longrightarrow$ X is Δ'-meager.*

Proof. Suppose $G \in X$ is Δ-generic. Then $\{G\} \subseteq X$ is not Δ-meager, so X is cannot be Δ-meager by theorem 9.

Let $g \in \Delta'$ be a universal function for Δ, and let

$$h(\langle x, \sigma \rangle) = \begin{cases} g(\langle x, \sigma \rangle) & \text{if } \sigma \subseteq g(\langle x, \sigma \rangle), \\ \sigma & \text{otherwise,} \end{cases}$$

for all x and σ. Clearly $h \in \Delta'$, and h_i is a strategy for all i. Suppose X is not Δ'-meager. Then h cannot witness that X is Δ-meager, which means that there is a $G \in X$ that meets h_i for all i. But every strategy in Δ is of the form h_i for some i, so G is Δ-generic. ▯

It is easy to show that for $i \geq 1$, p_i-generics exist in $\mathbf{E}_{i+1}$ and that pspace_i-generics exist in $\mathbf{ESPACE}_{i+1}$, using the same techniques as theorem 10 and the fact that p_{i+1} contains universal functions for p_i. See theorem 6.2 of [Lut89] for a similar statement about p_i-random sets. It is also not hard to show that the rec-generic sets are exactly the weakly 1-generic sets defined in [Kur83].

4 Encoding-Dependent Results

In this section we compare the relative power of the different encoding combinations for defining Δ-meagerness for a time-bounded class Δ. First we show that the definitions of Δ-meagerness using the St, St and Lp, Lp combinations are of incomparable strength. To prove one direction of incomparability, we show that St, St-encodings of strategies are better than Lp, Lp-encodings at producing large gaps in a language. The other direction is much more drastic; we show that $\mathrm{p}^{\mathrm{St},\mathrm{St}}$-generic oracles cannot ensure the separation of $\mathbf{P}$ from $\mathbf{NP}$, but that $\mathrm{p}^{\mathrm{Lp},\mathrm{Lp}}$-generics can.

These results and proposition 5 imply that the St, Lp combination is strictly more inclusive than either the St, St or Lp, Lp combinations. This provides a strong argument for adopting St, Lp as the proper means of encoding strategies for defining resource-bounded category.

We first present a basic lemma limiting the growth rates of time-bounded strategies.

Lemma 16 *Fix $i \geq 1$ and let f be a strategy. If the* St, St*-encoding of f is in p_i, then there is a function $h \in G_i$ such that*

$$\max(f(\sigma)) \leq h(\max(\sigma)).$$

If the Lp, Lp*-encoding of f is in p_i, then there is a function $h \in G_i$ such that*

$$\|f(\sigma)\| + |\max(f(\sigma))| = h(\|\sigma\| + |\max(\sigma)|).$$

Proof. We prove the first inequality; the second inequality is similar. Let f' be the St, St-encoding of f, and let $h' \in G_i$ be nondecreasing such that for all x, $|f'(x)| \leq h'(|x|)$. By equations (1) and (2), we have

$$\begin{aligned} \max(f(\sigma)) &\leq q_0(|\mathrm{St}(f(\sigma))|) \\ &= q_0(|f'(\mathrm{St}(\sigma))|) \\ &\leq q_0(h'(|\mathrm{St}(\sigma)|)) \\ &\leq q_0(h'(q_0(\max(\sigma)))). \end{aligned}$$

We take $h(n) = q_0(h'(q_0(n)))$. Since $q_0 \in G_i$ and G_i is closed under composition, it follows that $h \in G_i$. ▯

4.1 Making Gaps

Here we show that there are $\mathrm{p}^{\mathrm{St},\mathrm{St}}$-meager sets which are not $\mathrm{p}_i^{\mathrm{Lp},\mathrm{Lp}}$-meager for any i.

Theorem 17 *The set $X = \{A \mid (\forall n \in \omega)(\exists x \in A)\ |x| = n\}$ is $\mathrm{p}^{\mathrm{St},\mathrm{St}}$-meager but not $\mathrm{p}_i^{\mathrm{Lp},\mathrm{Lp}}$-meager for any $i \geq 1$.*

Proof. The function

$$g(\sigma) = \mathrm{zfill}(\sigma, 0^{|\max(\sigma)|+2})$$

is a winning strategy for Player II in the game $G[X; \omega^\omega, \mathrm{p}^{\mathrm{St},\mathrm{St}}]$ (its St,St-encoding is clearly in p). Thus X is $\mathrm{p}^{\mathrm{St},\mathrm{St}}$-meager by theorem 13.

Fix $i \geq 1$ and let g be a strategy whose Lp,Lp-encoding is in p_i. By lemma 16, there is an increasing function $h \in G_i$ such that

$$||f(\sigma)|| + |\max(f(\sigma))| \leq h(||\sigma|| + |\max(\sigma)|).$$

We describe a strategy f for Player I in the game $G[X; \omega^\omega, \mathrm{p}_i^{\mathrm{Lp},\mathrm{Lp}}]$ so that $R(f, g) \in X$. Given $\sigma \in 2^{<\omega}$, let $f(\sigma)$ be as follows:

1. Let N be least such that $N > |\max(\sigma)|$ and $h(2N + ||\sigma||) < 2^N$. [Such an N exists since h grows subexponentially.]

2. Call a number $n \in \omega$ *critical* if

   - for no x of length n does $\sigma(x)\downarrow = 1$, and
   - there exists a y of length n not in the domain of σ.

3. Let $n_1, n_2, \ldots, n_m$ be all the critical numbers less than N, and for $1 \leq j \leq m$ let x_j be the least number of length n_j not in $\mathrm{dom}(\sigma)$.

4. Let $f(\sigma) = \sigma \cup \{(x_j, 1) \mid 1 \leq j \leq m\}$.

The number N is chosen so that on Player II's next turn, she does not have time to define $[g(f(\sigma))](x) = 0$ for all x of length N, and for all $n < N$ there is some x of length n with $[f(\sigma)](x) = 1$. A straightforward induction on the number of turns of the play (f, g) shows that for every n there is an $x \in R(f, g)$ of length n. ▯

4.2 P *versus* NP

In this section we show that for the time-bounded classes, the St,St-encodings and Lp,Lp-encodings differ in a crucial way: $\mathrm{p}^{\mathrm{St},\mathrm{St}}$-generics are not strong enough to guarantee separation of **P** from **NP**, whereas $\mathrm{p}^{\mathrm{Lp},\mathrm{Lp}}$-generics always separate **P** from **NP**. When we relativize **PSPACE** in the following theorem, we include the space used to make oracle queries in the space bound. Thus a **PSPACE** oracle machine can only make polynomial length oracle queries.

Theorem 18 *The set $X = \{A \mid \mathbf{P}^A = \mathbf{PSPACE}^A\}$ is not $\mathrm{p}_2^{\mathrm{St},\mathrm{St}}$-meager.*

Proof. Fix any strategy g whose St, St-encoding is in p_2. We describe a strategy $f \in \omega^\omega$ for Player I that forces

$$\mathbf{P}^{R(f,g)} = \mathbf{PSPACE}^{R(f,g)},$$

and thus g cannot be a winning strategy for Player II in the game $G[X; \omega^\omega, \mathrm{p}^{\mathrm{St},\mathrm{St}}]$. Then by theorem 13, X is not $\mathrm{p}_2^{\mathrm{St},\mathrm{St}}$-meager. The function f uses a standard coding technique to identify $\mathbf{P}$ and $\mathbf{PSPACE}$, while leaving gaps between codings large enough to let Player II play without interfering with the codings.

Let M be an alternating oracle Turing machine which recognizes some categorical **PSPACE**-complete language L^A, e.g., for every oracle A,

$$L^A = L(M^A) = \{\langle i, x, 0^t\rangle \mid N_i^A \text{ accepts } x \text{ in less than } t \text{ steps}\},$$

where $\{N_0, N_1, N_2, \ldots\}$ is some appropriate enumeration of all alternating oracle Turing machines. We can assume that M runs in time p for all oracles, where p is a polynomial and $p(n+1) \geq p(n) > n$ for all n. It follows from lemma 16, by letting $i = 2$ and taking the length of both sides, that there is a strictly increasing polynomial q such that $q(n) > n$ for all n, and

$$|\max(g(\sigma))| < q(|\max(\sigma)|)$$

for all $\sigma \in 2^{<\omega}$. Define the constants

$$\begin{aligned}
c_0 &= 1 \\
c_1 &= q(1) \\
c_2 &= q(q(1)) \\
&\vdots \\
c_{i+1} &= q(c_i) \\
&\vdots
\end{aligned}$$

For every $n \in \omega$, let $F(n)$ be the least c_i greater than $p(n)$. Note that $F(n)$ is polynomially bounded in n, and computable in time polynomial in n. Furthermore, $F(n) > n+1$ for all n. Player I does all her coding on numbers of length c_i, letting Player II play strategy g in between.

Given $\sigma \in 2^{<\omega}$, let $c(\sigma)$ be the least c_i greater than $|\max(\sigma)|$, and let $\tau = \mathrm{zfill}(\sigma, 0^{c(\sigma)})$. Define

$$f(\sigma) = \tau \cup \{\, (x10^{c(\sigma)-|x|-1}, L^\tau(x)) \mid x \in \omega \;\;\&\;\; F(|x|) = c(\sigma)\,\}.$$

Note that $c(\lambda) = c_0$, and if $c(\sigma) = c_i$ for some $i \in \omega$, then $|\max(f(\sigma))| = c_i$, and thus

$$|\max(g(f(\sigma)))| < q(c_i) = c_{i+1}.$$

Therefore, on each successive turn for Player I, she codes $L^\tau(x)$ on numbers of length c_i for successive i, while padding any previously uncommitted gaps with zeros, forcing the computations of $M^\tau(x)$. In other words, for all $x \in \omega$ we have

$$L^{R(f,g)}(x) = [R(f,g)](x10^{F(|x|)-|x|-1}),$$

and so $L^{R(f,g)}$ is polynomial time many-one reducible to $R(f,g)$, which proves the theorem. $\square$

Corollary 19 *There exists a* $\mathrm{p}^{\mathrm{St},\mathrm{St}}$*-generic set* G *such that* $\mathbf{P}^G = \mathbf{PSPACE}^G$.

Proof. Apply lemma 15 with $X = \{A \mid \mathbf{P}^A = \mathbf{PSPACE}^A\}$. $\square$

Theorem 20 *If G is $\mathrm{p}^{\mathrm{Lp},\mathrm{Lp}}$-generic, then $\mathbf{P}^G \neq \mathbf{NP}^G$.*

Proof. For all $A \subseteq \omega$, let

$$L^A = \{x \mid (\exists y) \ \ |y| = |x| \ \& \ x\char`^y \in A\},$$

where $x\char`^y$ is the concatenation of the binary representations of x and y. Clearly, $L^A \in \mathbf{NP}^A$ for all A. Let p be a polynomial, and let M be a deterministic oracle Turing machine running in time p for all oracles. Also let m_0 be the least number such that $p(m) < 2^m$ for all $m \geq m_0$. We compute a strategy h as follows: given an input σ,

1. Let x be smallest such that $|x| \geq m_0$ and $\sigma(x\char`^y)\uparrow$ for all y of length $|x|$.
2. Let $\{q_1, \ldots, q_k\}$ be the set of oracle queries made by $M^\sigma(x)$ that are not in the domain of σ. [Note that $k \leq p(|x|)$.]
3. Set $\tau \leftarrow \sigma \cup \{(q_i, 0) \mid 1 \leq i \leq k\}$.
4. Let y be the least number of length $|x|$ not in $\mathrm{dom}(\tau)$. [Such a y exists by the choice of m_0.]
5. If $M^\sigma(x)$ rejects, return $\tau \cup \{(x\char`^y, 1)\}$.
6. If $M^\sigma(x)$ accepts, return $\tau \cup \{(x\char`^z, 0) \ \mid \ |z| = |x| \ \& \ \tau(x\char`^z)\uparrow\}$.

We show that the Lp, Lp-encoding of h is in p. The input σ is given to us encoded as a list of pairs. In step 1, x can be found in polynomial time by searching for the least x of length at least m_0 whose binary representation does not occur as the first half of some element of $\mathrm{dom}(\sigma)$. Steps 2, 3, 4, and 5 can likewise be done in polynomial time by simulating the machine M. The only problem is step 6. We must add up to $2^{|x|}$ pairs to τ. Notice, however, that for all $u < x$, there is a v such that $u\char`^v \in \mathrm{dom}(\sigma)$ (except possibly if $|u| < m_0$) by the choice of x. Therefore,

$$||\sigma|| \geq x - 2^{m_0} \geq 2^{|x|} - 2^{m_0} - 1,$$

which implies that step 6 can be done in polynomial time after all. It is also clear that for all σ, h forces $M^{h(\sigma)}$ to compute $L^{h(\sigma)}$ incorrectly on at least one input. Therefore, for any A that meets h,

$$L^A \neq L(M^A).$$

If G is $\mathrm{p}^{\mathrm{Lp},\mathrm{Lp}}$-generic, G meets all strategies h whose Lp, Lp-encodings are in p. Thus $L^G \neq L(M^G)$ for any polynomial-time deterministic TM, and so $L^G \notin \mathbf{P}^G$. ▯

Corollary 21 *The set $\{A \mid \mathbf{P}^A = \mathbf{NP}^A\}$ is $\mathrm{p}_2^{\mathrm{Lp},\mathrm{Lp}}$-meager.*

Proof. Apply lemma 15. ▯

Corollary 21 and theorem 17 together show that the notions of $\Delta^{\mathrm{St},\mathrm{St}}$-meagerness and $\Delta^{\mathrm{Lp},\mathrm{Lp}}$-meagerness are incomparable. This together with proposition 5 implies that the notion of $\Delta^{\mathrm{St},\mathrm{Lp}}$-meagerness is strictly more inclusive than both.

5 $\mathbf{P}$ *versus* $\mathbf{UP}$ and $\mathbf{NP} \cap \mathbf{co\text{-}NP}$

In [BI87], Blum and Impagliazzo proved a rather counterintuitive result about generic sets as oracles. They showed, given the assumption $\mathbf{P} = \mathbf{NP}$ (unrelativized), that

$$\mathbf{P}^G = \mathbf{UP}^G = (\mathbf{NP} \cap \text{co-}\mathbf{NP})^G$$

for any generic G.[2] This result runs against the common wisdom that a generic oracle separates complexity classes whenever possible.

A natural question to ask is: What degree of genericity is needed to prove this result? We show in theorem 22 that no recursively bounded notion of genericity is strong enough to ensure the statement above.

In the following theorem, the choices of encodings are irrelevant, so we again assume implicit encodings by identifying $2^{<\omega}$ and ω and omitting the superscripts on function classes.

Theorem 22 *The set* $X = \{A \mid \mathbf{P}^A \neq (\mathbf{UP} \cap \textit{co-}\mathbf{UP})^A\}$ *is not* rec*-meager.*

Proof. Let g be a recursive strategy. It suffices to describe a strategy f such that $R(f,g) \in X$. We use a typical oracle construction to put a language in $(\mathbf{UP} \cap \text{co-}\mathbf{UP}) - \mathbf{P}$, except that we provide for large gaps where the language does not depend on the oracle at all. We only allow Player II to play inside these gaps. Since g is recursive, we can make these gaps recognizable in polynomial time.

Let $\hat{g}$ be a recursive function such that for all $n \in \omega$, we have $\hat{g}(n) > n$ and

$$\hat{g}(n) > |\max(g(\sigma))|$$

for all σ with $|\max(\sigma)| = n$. Define

$$\begin{aligned} h(0) &= 1 \\ h(n+1) &= \hat{g}(h(n)+1). \end{aligned}$$

Clearly, h is recursive. Choose $\hat{h}$ to be a strictly increasing recursive function so that given n, $\hat{h}(n) \geq h(n)$, and one can test if $n \in \operatorname{rng}(\hat{h})$ in time polynomial in n. It is well known that such functions exist (for example, take $\hat{h}$ to be time-constructible). Let $U = \operatorname{rng}(\hat{h})$. Finally, given an oracle A, define the languages

$$S^A = \{0^n \mid (\exists x)\ |x| = n-1 \ \ \&\ \ x0 \in A\}$$

and

$$T = \{0^u \mid u \in U\},$$

and let $L^A = S^A \cap T$. It follows from the remarks above that T is polynomial-time computable, and thus L^A is categorically in $\mathbf{NP}^A$. Throughout the construction, we will guarantee that for all $u \in U$, there is exactly one element of $R(f,g)$ of length u. This implies that

$$(\exists x)[|x| = u-1 \ \ \&\ \ x0 \in R(f,g)] \Longleftrightarrow \neg(\exists y)[|y| = u-1 \ \ \&\ \ y1 \in R(f,g)],$$

and also that

$$L^{R(f,g)} \in (\mathbf{UP} \cap \text{co-}\mathbf{UP})^{R(f,g)}.$$

[2] Although they did not state it explicitly, their proof actually shows that if $\mathbf{P} = \mathbf{NP}$, then $\mathbf{NP}^G$ has no $\mathbf{P}^G$-inseparable sets, from which the equalities $\mathbf{P}^G = \mathbf{UP}^G$ and $\mathbf{P}^G = (\mathbf{NP} \cap \text{co-}\mathbf{NP})^G$ follow as corollaries. For information on $\mathbf{P}$-inseparable sets, see [HS89] for example.

Let $M_0, M_1, M_2, \ldots$ be a list of all deterministic polynomial time oracle Turing machines, each machine M_i running in time p_i (we assume $p_i(n) \geq n$). Let R_i be the requirement that $L^{R(f,g)} \neq L(M_i^{R(f,g)})$. A finite function τ *satisfies* requirement R_i if there is an $n \in \omega$ such that

$$L^\tau(0^n) \neq M^\tau(n),$$

and τ is defined on all numbers of length up through $p_i(n)$. Note that if τ satisfies R_i, then any extension of τ also satisfies R_i.

Given $\sigma \in 2^{<\omega}$, we define $f(\sigma)$ as follows:

1. Let R_i be the least requirement that σ does not satisfy. (We will define $f(\sigma)$ to satisfy R_i.)
2. Let $\ell = |\max(\sigma)|$.
3. Let $u \in U$ be least such that $\ell < u$ and $p_i(u) < 2^{u-1}$, and let $v \in U$ be least such that $p_i(u) < v$.
4. Let $\sigma_1 = \sigma \cup \{(0^{u'}, 1) \mid \ell < u' < v \;\; \& \;\; u' \in U \;\; \& \;\; u' \neq u\}$.
5. Let x and y be the least numbers of length $u - 1$ such that neither $x0$ nor $y1$ is queried by M_i in the computation $M_i^{\sigma_1}(0^u)$. (Both x and y exist by the choice of u.)
6. If $M^{\sigma_1}(0^u)$ accepts, let $\sigma_2 = \sigma_1 \cup \{(y1, 1)\}$; otherwise, let $\sigma_2 = \sigma_1 \cup \{(x0, 1)\}$.
7. Let $f(\sigma) = \text{zfill}(\sigma_2, 0^{v+1})$.

At step 3 we choose a length u at which to diagonalize against the machine M_i. Step 4 is needed to ensure our guarantee of unique elements of $R(f,g)$ of length $u' \in U$. We perform the actual diagonalization at step 6.

The last step fills all remaining holes in the finite function up through length v, satisfying requirement R_i and preventing Player II from disturbing the diagonalization on the next turn. By the definition of $\hat{h}$, Player II is also forced to play entirely within the gap between v and the next larger element v' of U. Thus Player II cannot void our guarantee on her next turn, i.e., $g(f(\sigma))$ is defined only on numbers of length less than v'. Our guarantee remains valid throughout the play, and all the requirements R_i are satisfied. Therefore,

$$L^{R(f,g)} \in [(\mathbf{UP} \cap \text{co-}\mathbf{UP}) - \mathbf{P}]^{R(f,g)},$$

and so $R(f,g) \in X$. ▯

Corollary 23 *If there exists a recursive universal function for the class Δ, then there are Δ-generic sets G such that*

$$\mathbf{P}^G \neq (\mathbf{UP} \cap \textit{co-}\mathbf{UP})^G.$$

Proof. Apply lemma 15. ▯

6 Further Research

One can show that 1-generic sets (see [Joc80]) are adequate for proving the results of Blum & Impagliazzo. We have shown here that subrecursive generics are not adequate. This leaves open the question of rec-generics: are rec-generics adequate to prove these results? This question seems difficult.

One can define resource-bounded strategies in other ways besides the two we have studied. To satisfy a requirement, we often only need to pad our output with zeros. We may take advantage of this compressibility to define a more compact encoding scheme. We may also decide to change the way a strategy accesses its input and writes its output. For example, we may define a strategy as a function $f: 2^{<\omega} \times \omega \rightarrow \{0, 1, \uparrow\}$, where $f(\sigma, x) = \uparrow$ for cofinitely many x, and $f(\sigma, x)$ represents the value at x of the strategy's output (given σ as input). We obtain notions of resource-bounded category by restricting the resources allowed to compute f. Another idea would be to let σ be accessed as an oracle which answers queries either with 0, 1, or $\uparrow$. We will investigate these ideas further and settle more pseudogeneric oracle questions in a future paper.

7 Acknowledgments

I wish to thank my advisor, Stuart Kurtz, for suggesting this line of research, and for a great many helpful discussions. I also wish to thank Jack Lutz for posing several interesting problems regarding resource-bounded generic sets.

References

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P = NP question. *SIAM J. Comp.*, 4:431–442, 1975.

[BI87] 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.

[HS89] S. Homer and A. L. Selman. Oracles for structural properties: The isomorphism problem and public-key cryptography. In *Proceedings of the 4th Annual IEEE Structure in Complexity Theory Conference*, pages 3–14, 1989.

[Joc80] C. G. Jockusch. Degrees of generic sets. In *Recursion Theory: Its Generalizations and Applications.* (F. R. Drake and S. S. Warner, Editors), Cambridge University Press, Cambridge, 1980.

[Kur83] S. A. Kurtz. Notions of weak genericity. *Journal of Symbolic Logic*, 48:764–770, 1983.

[Lut87] 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.

[Lut89] J. H. Lutz. Almost everywhere high nonuniform complexity. In *Proceedings of the 4th Annual IEEE Structure in Complexity Theory Conference*, pages 37–53, 1989.

[Lut90] J. H. Lutz. Category and measure in complexity classes. *SIAM J. Comp.*, 1990. To appear.

[Oxt80] J. C. Oxtoby. *Measure and Category.* Springer-Verlag, 1980.

[Soa87] R. Soare. *Recursively Enumerable Sets and Degrees.* Springer-Verlag, 1987.