Generic Separations*

Lance Fortnow†

University of Chicago
Department of Computer Science
1100 E. 58th St.
Chicago, IL 60637

Tomoyuki Yamakami

University of Toronto
Department of Computer Science
Toronto, Ontario
Canada M5S 1A4

second version: September 15, 1995

*An early version was in the Proceedings of the 9th Annual Conference on Structure in Complexity Theory, Amsterdam, 1994

†Partially supported by NSF grant CCR 92-53582.

Running Head:

Generic Separations

Mailing address:

Tomoyuki Yamakami
Department of Computer Science
University of Toronto
10 King's College Road
Sandford Fleming Building
Toronto, Ontario
Canada M5R 2T8

email: yamakami@cs.toronto.edu
tel: (416) 978-6025
fax: (416) 978-1931

Abstract

In 1987, Blum and Impagliazzo, using techniques of Hartmanis and Hemachandra and Rackoff, show that if $\mathbf{P} = \mathbf{NP}$ then $\mathbf{P}(G) = \mathbf{NP}(G) \cap \mathbf{coNP}(G) = \mathbf{UP}(G)$, where G is a generic oracle. They leave open the question as to whether these collapses occur at higher levels of the polynomial-time hierarchy, *i.e.*, $\Delta_k^p(G) = \Sigma_k^p(G) \cap \Pi_k^p(G) = \mathbf{U}\Delta_k^p(G)$ for $k \geq 2$. Here we give a negative answer to these questions.

In fact, we demonstrate that, relative to any generic oracle G and for every $k \geq 2$, there exists a tally set in $\mathbf{U}\Delta_k^p(G) \cap \Pi_k^p(G)$ but not in $\Delta_k^p(G)$ by showing an exponential lower bound of a certain type of families of constant-depth circuits. An immediate corollary is that generic oracles separate $\Sigma_k^p \cap \Pi_k^p$ and Δ_k^p.

We also show that related results hold for type-2 complexity.

1 Introduction

In 1975, Baker, Gill and Solovay [1] created an oracle A relative to which $\mathbf{P}(A) \neq \mathbf{NP}(A)$. The proof used a then new technique of diagonalization by finite extension. First Baker, Gill and Solovay create a language $L(A)$ that depends on A designed such that $L(A) \in \mathbf{NP}(A)$ for all oracles A. They define requirements R_i that state that the ith polynomial-time Turing machine does not accept $L(A)$. They then create the oracle A to fulfill R_i for all i, and thus $\mathbf{P}(A) \neq \mathbf{NP}(A)$.

The construction of A proceeds by having only a finite number of strings of A defined at any given stage. Each requirement is fulfilled by only defining finitely many more strings.

Many other oracle constructions proceeded in this way including, for example, Yao's [19] oracle that separates the polynomial-time hierarchy. One can then look at what happens when we try to combine all of these constructions.

Generic oracles as defined in recursion theory (see [15]) do exactly this. Generics take all possible definable finite extensions. Thus, relative to any generic G, the polynomial-time hierarchy is infinite.

Blum and Impagliazzo [3] showed that generics can also collapse some classes. They showed how to use techniques from Rackoff [12] and Hartmanis and Hemachandra [8] to show that if $\mathbf{P} = \mathbf{NP}$ in the unrelativized universe, then $\mathbf{P} = \mathbf{UP} = \mathbf{NP} \cap \mathbf{coNP}$ relative to generic oracles. Though the $\mathbf{P} = \mathbf{NP}$ assumption seems unlikely, all the techniques used by Blum and Impagliazzo relativize so one can build their generic on top of an oracle that makes $\mathbf{P} = \mathbf{NP}$ [1]. See Fenner, Fortnow, Kurtz and Li [6] for other applications of generic oracles.

Blum and Impagliazzo leave open the question as to whether collapses hold higher up in the polynomial-time hierarchy: is it possible for generic oracles G that $\Sigma_k^p(G) \cap \Pi_k^p(G) = \Delta_k^p(G)$ or $\mathbf{U}\Delta_k^p(G) = \Delta_k^p(G)$ for $k \geq 2$, where $\Delta_k^p = \mathbf{P}(\Sigma_{k-1}^p)$ and $\mathbf{U}\Delta_k^p = \mathbf{UP}(\Sigma_{k-1}^p)$? We give a negative answer to this question: for any generic oracle G and any $k \geq 2$, $\mathbf{U}\Delta_k^p(G) \cap \Pi_k^p(G) \neq \Delta_k^p(G)$.

For the proof, we create a series of tally languages $L_k(A)$ based on permutations such that $L_k(A) \in \mathbf{U}\Delta_k^p(A) \cap \Pi_k^p(A)$ for all oracles A. For $k = 2$, we can show that, for generic G, $L_2(G) \notin \Delta_2^p(G)$ by using properties of relativized Δ_2^p due to Wilson [16]. For $k > 2$, we build on the $k = 2$ case by using some circuit complexity techniques of Håstad [9], Ko [10] and Sheu and Long [13].

Our results also have implications for type-two complexity. In type-two complexity, a Turing machine has an oracle for input as well as a string. See Yamakami [17, 18] for a background in type-two complexity. Cook, Impagliazzo, and Yamakami [5] showed a close connection between

generic oracle separations and type-two complexity class separations. Thus our result also shows that $U\Delta_k^{0,p} \cap \Pi_k^{0,p} \neq \Delta_k^{0,p}$ for all $k \geq 2$.

2 Definitions

In this paper, we will give the definitions necessary to describe our main result. For an overview of standard complexity classes, see [2], for generic oracles, see [3, 6], and for circuits, see [4]. We need circuit complexity in order to show our result for $k > 2$.

2.1 Complexity Classes

For this paper, fix $\Sigma = \{0, 1\}$ and let Σ^* be the set of all finite strings over Σ. By Σ^n (resp. $\Sigma^{<n}$), we denote the set of strings of length n (resp. $< n$). By ${}^{\Sigma^*}\Sigma^*$ we mean the set of all functions from Σ^* to Σ^*.

Denote by $\mathbf{P}$ and $\mathbf{NP}$ the collections of all sets which are deterministically and nondeterministically computable in polynomial time, respectively. Let $\mathbf{coNP}$ be the class of all complements of sets in $\mathbf{NP}$. Write $\mathbf{UP}$ for the class of sets computable by $\mathbf{NP}$ machines which have at most one accepting configuration on each input.

For a complexity class $\mathcal{C}$, we will often use the notation $\mathcal{C}(A)$ to represent the usual relativization of $\mathcal{C}$ to the oracle A.

We define the polynomial-time hierarchy recursively: Let $\Delta_0^p = \mathbf{U}\Delta_0^p = \Sigma_0^p = \Pi_0^p = \mathbf{P}$. Let $\Sigma_{k+1}^p = \mathbf{NP}(\Sigma_k^p)$, $\Pi_{k+1}^p = \mathbf{coNP}(\Sigma_k^p)$, $\mathbf{U}\Delta_{k+1}^p = \mathbf{UP}(\Sigma_k^p)$ and $\Delta_{k+1}^p = \mathbf{P}(\Sigma_k^p)$.

Let $\mathbf{SAT}^A$ be the standard relativized version of satisfiability (see [7]). For every oracle A, $\mathbf{SAT}^A$ is $\mathbf{NP}(A)$-complete.

A type-2 relation R on $\Sigma^* \times ({}^{\Sigma^*}\Sigma^*)$ is *polynomially bounded* if, for some polynomial p, $R(x, \alpha) = R(x, \alpha_{p(|x|)})$ for all string x and all function α, where $\alpha_t(y)$ is the first t symbols of $\alpha(y)$. We say that a class $\mathcal{C}$ of type-2 relations is *polynomially bounded* if all relations of $\mathcal{C}$ are polynomially-bounded. Denote by $U\Delta_k^{0,p}$, $\Delta_k^{0,p}$, and $\Pi_k^{0,p}$ the polynomially-bounded type-2 counterparts of $\mathbf{U}\Delta_k^p$, Δ_k^p, and Π_k^p, respectively [17, 18].

2.2 Generic Oracles

A *condition* σ is a function mapping Σ^* to $\{0,1\}$ with finite domain and is often identified with a finite string in a natural fashion. A condition τ *extends* a condition σ if the domain of σ is contained in the domain of τ and $\sigma(x) = \tau(x)$ for every x in the domain of σ. A set A *extends* a condition σ if $A(x) = \sigma(x)$ for all x in the domain of σ. A condition σ is also referred to as an finite oracle, and thus M^σ for an oracle Turing machine M denotes the machine that behaves as a regular oracle Turing machine with oracle $\{x \mid \sigma(x) = 1\}$ as long as a query is made in the domain of σ; otherwise, the output of the machine is undefined.

A set of conditions S is *dense* if, for *every* condition σ, there is a condition $\tau \in S$ such that τ extends σ. A set $A \subseteq \Sigma^*$ *meets* a set of conditions S if there is a condition $\sigma \in S$ such that A extends σ. A set S is *arithmetical* if it is exactly definable in first-order arithmetic, *i.e.*, there is a computable relation R on strings such that $S = \{x \mid Q_1 y_1 Q_2 y_2 \cdots Q_k y_k R(x, y_1, \ldots, y_k)\}$, where each Q_i is a quantifier $\forall$ or $\exists$ (see, *e.g.*, [11]). A set G is *(Cohen) generic* if G meets every dense arithmetical set of conditions.

2.3 Circuits

A *circuit* is a rooted tree, where each non-leaf node is associated with a gate, and each leaf is associated with a constant 0, a constant 1, a variable x, or a negated variable $\overline{x}$. This paper uses four types of gates: AND, OR, XOR, and AsymOR, where XOR denotes the exclusive OR, and AsymOR(x, y) means $x \vee \overline{y}$. For simplicity, we assume that each variable is represented by a string in Σ^*.

The *fanin* of a gate is the number of inputs to it. The *bottom fanin* of a circuit is the maximum fanin of its bottom gate. The *level* of a gate G is the length of the path in a circuit from the top gate (*i.e.*, the root) to G. The *depth* of a circuit is the length of the longest path from the top gate to the bottom gates. *Dual circuits* are recursively defined only for circuits that have no XORs or AsymORs as follows: the dual of a single constant or a variable is its negation, and the dual of a circuit C that is an OR (resp. AND) of subcircuits C_i, $1 \leq i \leq m$, is the AND (resp. OR) of the duals of C_i's.

A restriction of a circuit is a mapping from variables to $\{0, 1, *\}$. For a circuit C and a restriction ρ, $C\lceil_\rho$ denotes the circuit obtained from C by replacing each variable x with $\rho(x) = 0$ by 0 and each y with $\rho(y) = 1$ by 1, and by leaving the other variables unchanged. For a set A, ρ_A denotes

the restriction such that, for all variable $z \in \Sigma^*$, $\rho_A(v_z) = 1$ if $z \in A$, and $\rho_A(v_z) = 0$ if $z \notin A$.

Definition 2.1 *Let $k \geq 1$. A $\Sigma_k(n)$-circuit is a depth-$(k+1)$ circuit such that*

(1) it has alternating OR and AND gates with a top OR gate,

(2) the number of gates at each level 1 to level k is $\leq 2^n$, and

(3) fanins of level $k+1$ are $\leq n$.

A $\Pi_k(n)$-circuit is a dual circuit of $\Sigma_k(n)$-circuit.

Ko [10] introduced a C_k^n circuit to prove the separation of a relativized polynomial-time hierarchy.

Definition 2.2 *For each $n \geq 1$ and $k \geq 1$, a C_k^n circuit is a depth-k circuit such that*

(1) C_k^n has alternating OR and AND gates, with a top OR gate,

(2) all the fanins of C_k^n are exactly n, except the bottom fanins which are exactly $\sqrt{n}$, and

(3) each leaf of C_k^n is a unique positive variable.

Let f_k^n be a function computed by C_k^n.

For convenience sake, let C_0^n be a positive variable. Note that any $\Sigma_k(n)$-circuit contains a subcircuit computing a function $f_k^{2^n}$.

Sheu and Long [13] introduced the $\Delta_k(m)$-circuits which relates to $\Delta_k^p(A)$ sets.

Definition 2.3 *For $k \geq 2$, C is a $\Delta_k(m)$-circuit if*

(1) C has $k+2$ levels, and its top gate is an OR with fanin $\leq 2^m$,

(2) the gates at the second level are AND gates with fanins $\leq m$,

(3) each AND gate G_j at the second level has a distinct label of m bits, $b_1 \cdots b_m$, and if $b_i = 1$ then the i-th subcircuit from the left of G_j is a $\Sigma_{k-1}(m)$-circuit, and if $b_i = 0$ then the i-th subcircuit from the left of G_j is a $\Pi_{k-1}(m)$-circuit,

(4) for any two AND gates G_j and G_n, labeled l_j and l_n, at the second level, if the first r bits, $r < m$, of labels l_j and l_n are the same, then the first r subcircuits from the left of G_j and G_n are the same, and

(5) for any two AND gates G_j and G_n, labeled l_j and l_n, at the second level, if the first bit from the left of l_j and l_n, where l_j and l_n differ, is b_r, $r \leq m$, then the r-th subcircuits from the left of G_j and G_n are dual circuits.

Lemma 2.4 [13] *Let $k > 1$. For every $\Delta_k^p(A)$ machine M^A, there is a polynomial q such that, for every input string x, there is a corresponding $\Delta_k(q(|x|))$-circuit $C_{M,x}$ such that M^A accepts $x \Longleftrightarrow C_{M,x}\lceil_{\rho_A} \equiv 1$.*

Let V be a set of variables, and let $\mathcal{B} = \{B_j\}_{j=1}^r$ be a partition of V. Let q be a real number between 0 and 1. Let $R_{q,\mathcal{B}}^+$ be a probability space of restrictions which take values as follows: for each B_j, let $s_j = *$ with probability q and $s_j = 0$ with probability $1-q$; and then independently, for each variable $x \in B_j$, let $\rho(x) = s_j$ with probability q and $\rho(x) = 1$ with probability $1-q$. Similarly, a $R_{q,\mathcal{B}}^-$ is defined by interchanging the roles of 0 and 1.

Define a restriction $g(\rho)$ for a $\rho \in R_{q,\mathcal{B}}^+$ as follows: for all B_j with $s_j = *$, let V_j be the set of all variables in B_j which are given the value $*$ by ρ; $g(\rho)$ selects one variable y which has the highest index in V_j and gives value $*$ to y and value 1 to all others in V_j. Similarly $g(\rho)$ for a $\rho \in R_{q,\mathcal{B}}^-$ is defined by interchanging the roles of 0 and 1. Let $\rho g(\rho)$ denote the composition of ρ and $g(\rho)$. We will make use of the following switching lemma due to Håstad:

Lemma 2.5 [9] *Let s, t be integers and q be a real number between 0 and 1. Let G be an AND of ORs with bottom fanin $\leq t$, F be an arbitrary function and $\mathcal{B} = \{B_j\}_j$ be a partition of variables in G and F. Then, for a random restriction ρ from $R_{q,\mathcal{B}}^+$, the probability that $G\lceil_{\rho g(\rho)}$ is not equivalent to a circuit of OR of ANDs with bottom fanin $\leq s$ under the condition that $F\lceil_\rho \equiv 1$ is bounded by α^s, where $\alpha < 6qt$.*

In the lemma, we can replace $R_{q,\mathcal{B}}^+$ by $R_{q,\mathcal{B}}^-$, and an AND of ORs by an OR of ANDs.

3 Main Theorem

Since Blum and Impagliazzo [3] proved a collapse result relative to a generic oracle at the first level of the polynomial-time hierarchy under the assumption $\mathbf{P} = \mathbf{NP}$, it is long left open whether similar collapses hold at any level of the hierarchy. In this paper, we show:

Theorem 3.1 *For $k \geq 2$, $\boldsymbol{U}\Delta_k^p(G) \cap \Pi_k^p(G) \neq \Delta_k^p(G)$ with a generic oracle G.*

Since $\mathbf{U}\Delta_k^p(G) \subseteq \Sigma_k^p(G)$, this theorem has the following immediate corollary:

Corollary 3.2 *For $k \geq 2$, $\Sigma_k^p(G) \cap \Pi_k^p(G) \neq \Delta_k^p(G)$ with a generic oracle G.*

Complexity classes relative to a generic oracle have strong relationships to their associated *resource-bounded* type-2 complexity classes. Due to the recent result by Cook, Impagliazzo, and Yamakami [5], separations between complexity classes in the polynomial-time hierarchy can be translated into separations between the associated type-2 complexity classes of polynomially-bounded relations. Hence, we immediately get the following corollary about type-2 complexity:

Corollary 3.3 *$U\Delta_k^{0,p} \cap \Pi_k^{0,p} \neq \Delta_k^{0,p}$ for all $k \geq 2$.*

Proof of Theorem 3.1. Given an oracle A, define a length-preserving function f^A that reads its output directly from the oracle by

$$f^A(x) = A(x0^{|x|})A(x0^{|x|-1}1)A(x0^{|x|-2}1^2)\cdots A(x01^{|x|-1}).$$

Let $f_n^A : \Sigma^n \to \Sigma^n$ be f^A restricted to inputs of length n and let

$$\mathrm{PERM}^A = \{1^n \mid f_{2n}^A \text{ is a permutation}\}.$$

It is not difficult to see that PERM^A is in $\mathbf{coNP}^A$ for all A.

Let S_{2n} be the set of strings of the form $1^n(0+1)^n + (0+1)^{n-1}0^{n+1}$. Note that $|S_{2n}| = \frac{3}{2} \cdot 2^n$. We then define a test set $L(A)$ as follows:

$$L(A) = \{1^n \in \mathrm{PERM}^A \mid \exists y \in \Sigma^n \forall z \in \Sigma^n [f_{2n}^A(yz) \in S_{2n}]\}.$$

In § 4, we will prove the following lemma:

Lemma 3.4 *For any oracle A,*

1. *$L(A) \in \boldsymbol{U}\Delta_2^p(A)$, and*
2. *$L(A) \in \Pi_2^p(A)$.*

Now we define $L_k(A) = L(E_{k-2}(A))$ for each $k \geq 2$, where $E_0(A) = A$ and

$$E_k(A) = \{x \mid (\exists y_1 \in \Sigma^{|x|})(\forall y_2 \in \Sigma^{|x|})\cdots(Q_k y_k \in \Sigma^{|x|})[x y_1 y_2 \cdots y_k \in A]\},$$

where Q_k denotes $\exists$ if k is odd, and $\forall$ otherwise. Since $E_k(A) \in \Sigma_k^p(A)$ for every $k \geq 0$, we immediately get the following corollary:

Corollary 3.5 *For any oracle A and any $k \geq 2$, $L_k(A) \in \boldsymbol{U}\Delta_k^p(A) \cap \Pi_k^p(A)$.*

In § 5, we will prove:

Lemma 3.6 *For $k \geq 2$ and G generic, $L_k(G) \notin \Delta_k^p(G)$.*

Theorem 3.1 follows from Corollary 3.5 and Lemma 3.6. □

4 Containments

In this section, we will prove the containments of Lemma 3.4.

Proof of Lemma 3.4 (1). Note that if f_{2n}^A is a permutation, then there can only be at most one y such that, for all z, $f_{2n}^A(yz) \in S_{2n}$ since otherwise there will be too many strings mapping into S_{2n}. So, we can create a $\mathbf{UP}(A)$ machine that first makes one $\mathbf{NP}(A)$ query to verify that f_{2n}^A is a permutation and then guesses a y and makes one additional $\mathbf{NP}(A)$ query to verify that, for all z, $f_{2n}^A(yz) \in S_{2n}$. □

For Lemma 3.4(2), we will use the probabilistic method to show that $L(A) \in \Pi_2^p(A)$ for all oracles A. Gács [14] first used the probabilistic method in this manner when he showed that $\mathbf{BPP} \subseteq \Sigma_2^p \cap \Pi_2^p$.

Proof of Lemma 3.4 (2). As in the previous proof, a $\Pi_2^p(A)$ machine can first verify that f_{2n}^A is indeed a permutation. Given that f_{2n}^A is a permutation, we can define the following $\Pi_2^p(A)$ expressions to decide whether $1^n \in L(A)$:

$$\forall z_1, \ldots, z_{5n} \exists y [f_{2n}^A(yz_1) \in S_{2n} \wedge \ldots \wedge f_{2n}^A(yz_{5n}) \in S_{2n}],$$

where the quantification is taken over strings of length n.

Clearly if $1^n \in L(A)$, then this expression will be true. Suppose that $1^n \notin L(A)$ and f_{2n}^A is a permutation. Define

$$T_y = \{z \mid f_{2n}^A(yz) \in S_{2n}, |y| = |z| = n\}.$$

Let y^* maximize $|T_{y^*}|$. Note that, for every $y \neq y^*$, $|T_y| \leq \frac{3}{4} \cdot 2^n$.

Let A_y be the event that $(f_{2n}^A(yz_1) \in S_{2n}) \wedge \ldots \wedge (f_{2n}^A(yz_{5n}) \in S_{2n})$ given that $z_1, \ldots, z_{5n}$ are chosen uniformly and independently at random from Σ^n. We will show that $\sum_{y \in \Sigma^n} \Pr(A_y) < 1$, and thus the claim follows.

Let z^* be such that $f_{2n}^A(y^*z^*) \notin S_{2n}$. We then have

$$\Pr(A_{y^*}) \leq \Pr(z_1 \neq z^*) = 1 - 2^{-n}.$$

For $y \neq y^*$, we have that

$$\Pr(A_y) = \Pr(z_1 \in T_y)^{5n} \leq \left(\frac{3}{4}\right)^{5n} < 2^{-2n}.$$

Thus,

$$\sum_y \Pr(A_y) = \Pr(A_{y^*}) + \sum_{y \neq y^*} \Pr(A_y) < 1 - 2^{-n} + 2^n \cdot 2^{-2n} = 1.$$

□

5 Separations

In § 5.2, we will prove Lemma 3.6 in its full generality. However, this proof requires some deep theorems from circuit complexity. To aid in understanding, we will first present in § 5.1 a much simpler proof for the separation at the second level.

5.1 Separation at the Second Level

The idea of our proof is that a $\Delta_2^p(A)$ machine can be determined by some setting of a polynomial number of strings in oracle A, but $L(A)$ can not be. Wilson [16] developed this technique when he created an oracle relative to which Δ_2^p has linear-size circuits.

Proof of Lemma 3.6 for $k = 2$. Let $M_1, M_2, \ldots$ be an effective enumeration of deterministic oracle Turing machines, where M_i runs in time $n^i + i$. We will show that there exists a set A such that $L(A) \neq L(M_i(\mathbf{SAT}^A))$ for all i. For defining A, we first recursively construct conditions $\sigma_0, \sigma_1, \ldots$ as follows: initially, let σ_0 be empty, and, for $i > 0$, let σ_i be a condition extending σ_{i-1} such that $L(A) \neq L(M_i(\mathbf{SAT}^A))$ for all set A extending σ_i. Then, take a set A which extends all σ_i's.

Let $\sigma : \Sigma^* \to \{0, 1\}$ be a condition. Fix i. We need to show that there is a τ extending σ such that, for all A extending τ, $L(A) \neq L(M_i(\mathbf{SAT}^A))$.

Pick n large enough. Initially, let $\tau = \sigma$. Simulate M_i^τ on 1^n.

During the simulation of M_i, it may ask the oracle question "Is $\phi^A \in \mathbf{SAT}^A$?". We then do the following:

1. There does not exist an A that extends τ where f_{2n}^A is a permutation and makes ϕ^A satisfiable: answer no.

2. There is an A that extends τ where f_{2n}^A is a permutation and makes ϕ^A satisfiable: pick the smallest extension τ' of τ consistent with the fact that f_{2n}^A is a permutation and that forces ϕ^A satisfiable. Let $\tau = \tau'$ and answer yes.

Note that, after the simulation, τ forces M_i to either accept or reject, and $|\tau| - |\sigma|$ is bounded by a polynomial in n.

We can assume, without loss of generality, that if τ forces one bit of $f_{2n}^A(w)$, then it forces every bit of $f_{2n}^A(w)$. Let A be the set of all strings w of length $2n$ such that τ forces $f_{2n}^A(w)$. Let $B = f^A(\Sigma^{2n})$ as forced by τ. Note that $|A| = |B|$ which is bounded by a polynomial in n, and thus

1. $|\Sigma^{2n} - B - S_{2n}| > 2^n$;
2. $|S_{2n} - B| > 2^n$;
3. for every $y \in \Sigma^n$, there is a string $z \in \Sigma^n$ such that $yz \notin A$; and
4. there exists a $y \in \Sigma^n$ such that $yz \notin A$ for all $z \in \Sigma^n$.

If τ forces M_i to accept, then, for each y, pick a $z_y \in \Sigma^n$ and a $w_y \in \Sigma^{2n} - S_{2n} - B$ such that $yz_y \notin A$ and $w_y \neq w_{y'}$ for $y \neq y'$. Extend τ to force f_{2n}^A to be a permutation such that $f_{2n}^A(yz_y) = w_y$ for every y.

If τ forces M_i to reject, then pick a $y \in \Sigma^n$ such that $yz \notin A$ for all $z \in \Sigma^n$. For every z, pick a w_z in $S_{2n} - B$ such that $w_z \neq w_{z'}$ for $z \neq z'$. Extend τ to force f_{2n}^A to be a permutation such that $f_{2n}^A(yz) = w_z$ for every z. □

5.2 Exponential Lower Bound of Circuits

This section proves Lemma 3.6 by showing an exponential lower bound of a certain type of circuits. The proof uses ideas from [9, 10, 13].

We first introduce several types of circuits to describe a set $L_k(A)$.

Definition 5.1 *Let $k \geq 2$. An S_k^n circuit is a depth-$(k+2)$ circuit such that*

(1) S_k^n has alternating OR and AND gates at each level 1 to level 2, with a top OR gate,

(2) all the fanins of S_k^n at each level 1 to level 2 are exactly n,

(3) S_k^n has AsymOR gates at level 3, with fanin 2,

(4) the left gate of an AsymOR is an AND with fanin $\lfloor \log n \rfloor$, and the right gate of an AsymOR

is an OR with fanin $\lfloor \log n \rfloor + 1$,

(5) each subcircuit from a gate at level 4 is a C^n_{k-2} *circuit and every leftmost subcircuit from an OR gate at level 4 is called* redundant,

(6) for each AsymOR gate H *at level 3, same are the rightmost* C^n_{k-2} *subcircuit from an AND gate which is the left of* H *and the redundant* C^n_{k-2} *subcircuit from an OR gate which is the right of* H, *and*

(7) every C^n_{k-2} *circuit at level 4 which is not redundant has unique variables.*

An $S^{2^n}_k$ circuit relates to the condition that there exists a string $y \in \Sigma^n$ such that, for all $z \in \Sigma^n$, $f^{E_{k-2}(A)}_{2n}(yz) \in S_{2n}$.

Definition 5.2 *Let* $k \geq 2$ *and fix an* S^n_k *circuit* C. *Let* c^j_i, $1 \leq i \leq \lfloor \log n \rfloor$, *be the* i*-th subcircuit from the AND gate that is the left of the* j*-th AsymOR gate of* C, *and let* $c^j_{i+\lfloor \log n \rfloor}$, $1 \leq i \leq \lfloor \log n \rfloor$, *be the* $(i+1)$*-th subcircuit of the OR gate that is the right of the* j*-th AsymOR gate of* C, *where* $1 \leq j \leq \lfloor \log n \rfloor^2$.

A P^n_k *circuit (with respect to* C*) is a circuit such that*

(1) P^n_k *has a top AND gate with fanin* $\frac{1}{2}\lfloor \log n \rfloor^2(\lfloor \log n \rfloor^2 - 1)$,

(2) P^n_k *has OR gates at level 2 with fanin* $2 \cdot \lfloor \log n \rfloor$, *and every OR gate at level 2 is labeled with* $\langle j, j' \rangle$, *where* $1 \leq j < j' \leq \lfloor \log n \rfloor^2$,

(3) P^n_k *has XOR gates at level 3 with fanin 2, and*

(4) the i*-th XOR gate of the OR gate with label* $\langle j, j' \rangle$ *at level 2 has child nodes* c^j_i *and* $c^{j'}_i$.

A $P^{2^n}_k$ circuit relates to a set $\mathrm{PERM}^{E_{k-2}(A)}$.

Finally we define a family of U^n_k circuits which are pertinent to $L_k(A)$.

Definition 5.3 *A* U^n_k *circuit is an AND of an* S^n_k *circuit and an associated* P^n_k *circuit. Let* u^n_k *be a function computed by a* U^n_k *circuit.*

The definitions clearly show the following lemma:

Lemma 5.4 *For every* $k > 1$ *and every set* A, *there exists a family* $\{U^{2^n}_k\}$ *of circuits which computes* $L_k(A)$.

Note that, for any restriction ρ, if $U^{2^n}_2 \lceil_\rho$ is completely determined, then the number of variables of $U^{2^n}_2$ which have the value 0 or 1 by ρ is at least $n \cdot 2^n$ since either the left or right gates of all

AsymOR gates at level 3 that are inputs of some AND gate at level 2 of the associated $S_k^{2^n}$ circuit should be completely determined.

In what follows, we will prove that no $\Delta_k(r)$-circuit computes a $u_k^{2^n}$ function, where r satisfies $r < \frac{1}{48} \cdot 2^{\frac{n}{3}}$. To show this, here we need the following two claims.

(1) For every $\Delta_k(n)$-circuit C, with a high probability, $C\lceil_{\rho g(\rho)}$ is equivalent to a $\Delta_{k-1}(n)$-circuit.

(2) With a high probability, $U_k^n\lceil_{\rho g(\rho)}$ contains a subcircuit computing u_{k-1}^n.

We first prove claim (1). This proof is similar to [13].

Lemma 5.5 *For each $k \geq 3$, let C be a $\Delta_k(n)$-circuit, and let $\mathcal{B} = \{B_j\}$ be a partition of the variables in C. Let q be a real number between 0 and 1, and ρ be a random restriction from $R^+_{q,\mathcal{B}}$ if k is even or from $R^-_{q,\mathcal{B}}$ if k is odd. The probability that $C\lceil_{\rho g(\rho)}$ computes a $\Delta_{k-1}(n)$-circuit is $\geq \frac{2}{3}$ if $3n(24qn)^n \leq 1$.*

Proof. Let $k \geq 3$, and C be a $\Delta_k(n)$-circuit. Let C' be an arbitrary subcircuit of the two higher levels of C. Assume C' is an AND of ORs. By Lemma 2.5, the probability that $C'\lceil_{\rho g(\rho)}$ cannot be written as an OR of ANDs with bottom fanin $\leq n$ is $\leq \alpha^n$. Since there are 2^n such subcircuits, the probability that at least one $C'\lceil_{\rho g(\rho)}$ cannot be written as an OR of ANDs with bottom fanin $\leq n$ is $\leq (2\alpha)^n$. Similarly, if C' is an OR of ANDs, then the probability that at least one $C'\lceil_{\rho g(\rho)}$ cannot be written as an AND of ORs is $\leq (2\alpha)^n$.

Hence, the probability that one $\Sigma_{k-2}(n)$ or $\Pi_{k-2}(n)$-circuit cannot be converted to a $\Sigma_{k-3}(n)$ or $\Pi_{k-3}(n)$-circuit is $\leq (2\alpha)^n$. Note that there are $n \cdot 2^n$ $\Sigma_{k-2}(n)$ or $\Pi_{k-2}(n)$-circuits. Therefore, the probability that $C\lceil_{\rho g(\rho)}$ is equivalent to a $\Delta_{k-1}(n)$-circuit is $> 1 - n2^n(2\alpha)^n = 1 - n(4\alpha)^n$. This probability is $\geq \frac{2}{3}$ if $3n(4\alpha)^n \leq 1$. Clearly $\alpha < 6qn$ and $3n(24qn)^n \leq 1$ conclude that $3n(4\alpha)^n \leq 1$. □

We next prove claim (2) in a similar fashion to [10].

Lemma 5.6 *For each $k \geq 3$, there exists an integer n_k such that the following hold for all $n \geq n_k$. Let C be a U_k^n circuit (i.e., an AND of an S_k^n circuit and an P_k^n circuit). Let $q = n^{-\frac{1}{3}}$ and $\mathcal{B} = \{B_j\}$, where B_j is the set of variables in a bottom gate of an S_k^n subcircuit of C. Let ρ be a random restriction from $R^+_{q,\mathcal{B}}$ if k is even or from $R^-_{q,\mathcal{B}}$ if k is odd. The probability that $C\lceil_{\rho g(\rho)}$ contains a subcircuit computing a u_{k-1}^n function is $\geq \frac{2}{3}$.*

Proof. Let $k \geq 3$ and assume k is even. Note that the case for odd k is symmetric. Let C be a U_k^n circuit and ρ be a random restriction from $R_{q,\mathcal{B}}^+$. Note that all bottom gates of C are AND gates, and each of such AND gates has $\sqrt{n}$ variables. As in [10], it suffices to show the following two claims.

(1) The probability that all bottom AND gates $H_j\lceil_{\rho g(\rho)}$ (corresponding to block B_j by ρ) of $C\lceil_{\rho g(\rho)}$ take the value s_j is $> \frac{5}{6}$. This is seen as follows. The probability that each $H_j\lceil_{\rho g(\rho)}$ does not take s_j is

$$(1-q)^{\sqrt{n}} = \left((1-q)^{1/q}\right)^{q\sqrt{n}} \leq e^{-q\sqrt{n}} = e^{-n^{\frac{1}{6}}} \leq \frac{1}{6}$$

if $n \geq 27^2$ since $(1-x)^{1/x} \leq \frac{1}{e}$ for all $x > 0$.

(2) The probability that all OR gates G at level $(k+2)$ of $C\lceil_{\rho g(\rho)}$ have at least $\sqrt{n}$ child nodes $H_j\lceil_{\rho g(\rho)}$ of AND gates having value $s_j = *$ is $> \frac{5}{6}$. To show this, consider the probability, say p_i, that $G\lceil_{\rho g(\rho)}$ has exactly i AND gates $H_j\lceil_{\rho g(\rho)}$ which have values $s_j = *$. Since $p_i = \begin{pmatrix} n \\ i \end{pmatrix} q^i(1-q)^{n-i}$, we have $p_i = p_{i-1} \cdot \frac{q(n-i+1)}{i(1-q)} \geq 2p_{i-1}$ if $2\sqrt{n} \geq i$ and $n \geq 16^3$. Hence, for any $i \leq 2\sqrt{n}$, $p_i \leq p_{2\sqrt{n}} \cdot 2^{i-2\sqrt{n}} \leq 2^{i-2\sqrt{n}-1}$ since $p_{2\sqrt{n}} \leq \frac{1}{2}$. So,

$$\sum_{i=0}^{\sqrt{n}} p_i \leq \sum_{i=0}^{\sqrt{n}} 2^{i-2\sqrt{n}-1} \leq 2^{-\sqrt{n}} \leq \frac{1}{6}$$

if $n \geq 9$. □

Lemma 5.7 *For each $k \geq 2$, there exists an $n_k > 0$ such that, for all $n > n_k$, no $\Delta_k(r)$-circuit computes a $u_k^{2^n}$ function if $r < \frac{1}{48} \cdot 2^{\frac{n}{3}}$.*

Proof. Assume $r < \frac{1}{48} \cdot 2^{\frac{n}{3}}$. We first prove the base case $k = 2$. This is based on ideas of the proof in § 5.1.

Let C be a $\Delta_2(r)$-circuit and assume that C computes a $u_2^{2^n}$ function which is computed by an AND of an $P_2^{2^n}$ circuit C_p and an $S_2^{2^n}$ circuit C_s. Note that, by the definition of $\Delta_2(r)$-circuit, there are at most r distinct $\Sigma_1(r)$-circuits and their duals each of which is attached to some AND gate at level 2 of C. To determine $C\lceil_\rho$ completely, we only need to assign either 0 or 1 to at most r^2 variables in C.

We define a restriction ρ as follows. Initially let $\rho(x) = *$ for all x. Until $C\lceil_\rho$ is completely determined under the condition $C_p\lceil_\rho \not\equiv 0$, recursively take a $\Sigma_1(r)$-circuit C', an OR of ANDs,

attached to an AND gate at level 2 of C which is not determined yet, and force $C'\lceil_\rho$ to be 1 with keeping $C_p\lceil_\rho \not\equiv 0$. Note that $C'\lceil_\rho$ can be completely determined by assigning at most r variables in C'. Take a minimal ρ such that ρ is consistent with the restriction defined before, $C'\lceil_\rho \equiv 1$, and $C_p\lceil_\rho \not\equiv 0$. If there is no ρ such that $C'\lceil_\rho \equiv 1$ and $C_p\lceil_\rho \not\equiv 0$, then force $C'\lceil_\rho \equiv 0$ with $C_p\lceil_\rho \not\equiv 0$ by assigning at most r variables in C'.

After r steps, $C\lceil_\rho$ can be completely determined. Since $C_s\lceil_\rho$ is also completely determined by our assumption, we have $r^2 \geq n \cdot 2^n$. This is a contradiction since $r < \frac{1}{48} \cdot 2^{\frac{n}{3}}$.

For the induction step, $k \geq 3$, let C be a $\Delta_k(r)$-circuit and assume that C computes a $u_k^{2^n}$ function. Let $q = 2^{-\frac{n}{3}}$ and let $\mathcal{B} = \{B_j\}$, where B_j is the set of variables of the bottom gate H_j of a $U_k^{2^n}$ circuit which represents the given $u_k^{2^n}$ function. Note that we have $3r(24qr)^r \leq 1$ since $r < \frac{1}{48} \cdot 2^{\frac{n}{3}}$. We apply a random restriction ρ to C and $U_k^{2^n}$. Lemmas 5.5 and 5.6 ensure that there exists a restriction ρ' that $C\lceil_{\rho'}$ is equivalent to a $\Delta_{k-1}(r)$-circuit, and it computes a $u_{k-1}^{2^n}$ function. This is absurd by the induction hypothesis. □

Proof of Lemma 3.6. Fix $k \geq 2$. Let $M_1, \ldots$ be an enumeration of Δ_k^p oracle machines. Fix i. Let $\sigma : \Sigma^* \to \{0,1\}$ be a finite condition. We need to show that there is a τ extending σ such that, for all A extending τ, $L_k(A) \neq L(M_i^A)$.

Take a $U_k^{2^n}$ circuit C which computes set $L_k(A) \cap \Sigma^n$. Let $\{C_{M_i,x} \mid x \in \Sigma^*\}$ be the family of $\Delta_k(q(|x|))$-circuits that arise from Lemma 2.4.

Pick n large enough such that all the domain of $\sigma \subseteq \Sigma^{<n}$, and we can apply Lemma 5.7 to $C_{M_i,1^n}$. Pick a setting of the strings of length $2n$ of A in the oracle such that $C_{M_i,1^n}\lceil_{\rho_A} \not\equiv C\lceil_{\rho_A}$. If we let τ be σ extended by setting the strings of length $2n$ in this manner, then we will have the required condition. □

6 Discussions

We have proven that, relative to a generic oracle G, $\mathbf{U}\Delta_k^p(G) \cap \Pi_k^p(G)$ is different from $\Delta_k^p(G)$ for every $k \geq 2$. In fact, our techniques show that there exists a tally set in $\mathbf{U}\Delta_k^p(G) \cap \Pi_k^p(G) - \Delta_k^p(G)$.

Unanswered in this paper is, however, whether there exists a language in $\mathbf{U}\Delta_k^p(G) \cap co\text{-}\mathbf{U}\Delta_k^p(G)$ but not in $\Delta_k^p(G)$ for generic G for any $k \geq 2$.

Acknowledgments

The authors would like to thank Russell Impagliazzo for suggesting this problem and Stuart Kurtz, John Rogers, Steve Fenner and Duke Whang for various discussions on generic oracles.

References

[1] T. Baker, J. Gill, and R. Solovay. Relativizations of the P = NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

[2] Balcázar, Díaz, and Babaró, *Structural Complexity I, II.* Springer,erlin, 1989(I), 1991(II).

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

[4] R. Boppana and M. Sipser. The complexity of finite functions. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, chapter 14, pages 757–804. North-Holland, 1990.

[5] S.A. Cook, R. Impagliazzo, and T. Yamakami. A tight relationship between generic oracles and type-2 complexity theory. In preparation. A preliminary version was written by T. Yamakami under the same title as an unpublished manuscript, Department of Computer Science, University of Toronto, 1993.

[6] 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. IEEE, New York, 1993.

[7] J. Goldsmith and D. Joseph. Relativized isomorphisms of **NP**-complete sets. *Computational Complexity*, 3:186–205, 1993.

[8] J. Hartmanis and L. Hemachandra. One-way functions and the nonisomorphism of NP-complete sets. *Theoretical Computer Science*, 81(1):155–163, 1991.

[9] J. Håstad. Almost optimal lower bounds for small depth circuits. In S. Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 143–170. JAI Press, Greenwich, 1989.

[10] K. Ko. Relativized polynomial time hierarchies having exactly k levels. *SIAM Journal on Computing*, 18:392–408, 1989.

[11] P. Odifreddi. *Classical Recursion Theory*. North-Holland, Amsterdam, 1989.

[12] C. Rackoff. Relativized questions involving probabilistic algorithms. *Journal of the ACM*, 29(1):261–268, 1982.

[13] M. Sheu and T. Long. The extended low hierarchy is an infinite hierarchy. *SIAM journal on Computing*, 23(3):488–509, 1994.

[14] M. Sipser. A complexity theoretic approach to randomness. In *Proceedings of the 15th ACM Symposium on the Theory of Computing*, pages 330–335. ACM, New York, 1983.

[15] R. Soare. *Recursively Enumerable Sets and Degrees*. Springer, Berlin, 1987.

[16] C. Wilson. Relativized circuit complexity. *Journal of Computer and System Sciences*, 31:169–181, 1985.

[17] T. Yamakami. Structural properties for feasibly computable classes of type two. *Mathematical Systems Theory*, 25:177–201, 1992.

[18] T. Yamakami. Feasible computability and resource bounded topology. *Information and Computation*, 116(2):214–230, 1995.

[19] A. Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, pages 1–10. IEEE, New York, 1985.