Constructing Oracles by Lower Bound Techniques for Circuits [1]

Ker-I Ko
Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794

Separating or collapsing complexity hierarchies has always been one of the most important problems in complexity theory. For most interesting hierarchies, however, we have been so far unable to either separate them or collapse them. Among these unsolvable questions, whether P equals NP is perhaps the most famous one. In view of the fundamental difficulty of these questions, a less interesting but more realistic alternative is to consider the question in the relativized form. Although a separating or collapsing result in the relativized form does not imply directly any solution to the original unrelativized question, it is hoped that from such results we do gain more insight into the original questions and develop new proof techniques toward their solutions. Recent investigation in the theory of relativization shows some interesting progress in this direction. In particular, some separating results on the relativized polynomial hierarchy have been found using the lower bound results on constant-depth circuits [Yao, 1985; Hastad, 1986, 1987]. This new proof technique turns out to be very powerful, capable of even collapsing the same hierarchy (using, of course, different oracles) [Ko, 1989].

In this paper, we survey recent separating and collapsing results on several complexity hierarchies, including the polynomial hierarchy, the probabilistic polynomial hierarchy, the bounded Arthur-Merlin hierarchy, the generalized Arthur-Merlin hierarchy (or, the interactive proof systems), and the low hierarchy in NP. All these results rely on the newly developed lower bound results on constant-depth circuits.

[1] This paper is based on a lecture presented in the International Symposium on Combinatorial Optimization, held at Nankai Institute of Mathematics in Tienjing, People's Republic of China, August, 1988; research supported in part by the NSF Grant CCR-8801575.

We show how these new combinatorial proof techniques are combined with the classical recursion-theoretic proof techniques to construct the desirable oracles. Our focus is on how the diagonalization and the encoding requirements can be set up without interference to each other and how such complicated requirements can be simplified using lower bound results on constant-depth circuits.

In Section 1, we give the formal definitions of the relativized complexity hierarchies mentioned above. We avoid the definitions of different machine models but use the polynomial length-bounded quantifiers to give simpler definitions. This allows us to define easily, in Section 2, the related circuits for different complexity hierarchies. In Section 3, the basic proof technique of diagonalization is introduced and a simple example is given to show how oracles for separation results are constructed in this standard setting. Section 4 shows how the lower bound results on circuits can be combined with the diagonalization technique to separate various complexity hierarchies. In Section 5, the proof technique of encoding is introduced. Examples are given to show that to collapse hierarchies to a particular level requires both the diagonalization and the encoding techniques, together with more general lower bound results on circuits. These results on some hierarchies are given in Section 6. Section 7 deals with less familiar hierarchies as the applications of the standard proof techniques. This includes the generalized Arthur-Merlin hierarchy and the low hierarchy in *NP*. The last section lists some open questions in the theory of relativization of hierarchies related to our approach.

Notation. We will deal with strings over alphabet $\{0,1\}$. For each string x, let $|x|$ denote its length, and for each finite set A, let $\|A\|$ denote its cardinality. We write $A^{=n}$, $A^{<n}$, and $A^{\leq n}$ to denote the subsets $\{x \in A|\ |x| = n\}$, $\{x \in A|\ |x| < n\}$ and $\{x \in A|\ |x| \leq n\}$, respectively, while $\{0,1\}^n$ denotes all strings over $\{0,1\}$ of length n. We let $\langle, \cdots, \rangle$ be a fixed one-to-one pairing function from $\cup_{n\geq 1}(\{0,1\}^*)^n$ to $\{0,1\}^*$ such that $|x_i| < |\langle x_1, \cdots, x_n\rangle|$ for all $i \leq n$ if $n > 1$. All of our use of log is the logarithm function of base 2.

1. Relativized Complexity Classes

We begin with the formal definitions of relativized complexity classes. We assume that the reader is familiar with ordinary deterministic Turing machines (TMs)

and the complexity classes P, NP and $PSPACE$.[2] All the relativized complexity classes to be considered here, except the class $PSPACE(A)$, can be defined in terms of polynomial length-bounded quantifiers over polynomial-time predicates. So, we will only define the deterministic oracle machine and avoid specific machine structures of nondeterministic, probabilistic and alternating machines.

A (deterministic) oracle TM is an ordinary TM equipped with an extra tape, called the *query* tape, and three extra states, called the *query* state, the *yes* state and the *no* state. The query tape is a write-only tape to be used to communicate with the oracle set A. The oracle machine M operates in the same way as ordinary TMs if it is not in the query state. When it enters the query state, the oracle A takes over and performs the following tasks: it reads the query y from the query tape, cleans up the query tape, puts the head of the query tape to the original position, and puts machine M into the yes state if $y \in A$ or puts it into the no state if $y \notin A$. All of these actions made by the oracle count only one step of machine move. Let M be an oracle machine and A be a set. We write M^A to denote the computation of machine M using A as the oracle and write $L(M, A)$ to denote the set of strings accepted by M^A.

The time and space complexity of an oracle machine is defined in a natural way. Namely, for any fixed oracle A, M^A has time (space) complexity $\leq f(n)$ if for all x, $M^A(x)$ halts in $\leq f(|x|)$ moves (or, respectively, if $M^A(x)$ halts using $\leq f(|x|)$ cells, including the space of the query tape[3]). We say M has time (space) complexity $\leq f(n)$ if for all oracles A, M^A has time (space) complexity $\leq f(n)$. M is said to have a polynomial time (space) complexity if M has time (space) complexity $\leq p(n)$ for some polynomial function p. The complexity classes $P(A)$ and $PSPACE(A)$ can be defined as follows:

$$P(A) = \{L(M, A) |\ M^A \text{ has a polynomial time complexity}\},$$
$$PSPACE(A) = \{L(M, A) |\ M^A \text{ has a polynomial space complexity}\}.$$

The class $NP(A)$ may be defined to be the class of sets which are computed

[2] See Section 9 for references.

[3] The definition of space complexity of an oracle machine can vary depending upon whether we include the space of the query tape in the space measure. The different definitions may result in very different types of separation results. See, for example, Buss [1986] and Wilson [1988] for detailed discussions. In this paper, we are concerned only with polynomial space. Our definition allows the machine to query only about strings of polynomially-bounded length which is a natural constraint.

by nondeterministic oracle machines in polynomial time. Here, we avoid the formal definition of nondeterministic machines and define this class in terms of polynomial-time predicates. A predicate σ of a set variable A and a string variable x is called a polynomial-time predicate, or a P^1-predicate if there exist an oracle TM M and a polynomial p such that M has time complexity $\leq p(n)$ and for all sets A and all strings x, M^A accepts x iff $\sigma(A;x)$ is true. It is clear that $B \in P(A)$ iff there exists a P^1-predicate σ such that for all x, $x \in B \iff \sigma(A;x)$ holds. It is well known that the set $NP(A)$ can be characterized as follows: a set B is in $NP(A)$ iff there exist a P^1-predicate σ and a polynomial q such that $x \in B \iff (\exists y, |y| \leq q(|x|))\ \sigma(A;\langle x,y\rangle)$. In the rest of the paper, we will write $\exists_{\rm p} y$ (or $\forall_{\rm p} y$) to denote $(\exists y, |y| \leq q(|x|))$ (or, respectively, $(\forall y, |y| \leq q(|x|))$), if the polynomial bound q is clear in the context or if the exact bound is irrelevant. Using this notation, we define a $\Sigma_0^{P,1}$-predicate to be a P^1-predicate, and a $\Sigma_k^{P,1}$-predicate, for $k \geq 1$, to be a predicate having the form

$$\tau(A;x) \equiv (\exists_{\rm p} y_1)(\forall_{\rm p} y_2)\cdots(Q_k y_k)\ \sigma(A;\langle x, y_1, \cdots y_k\rangle),$$

where σ is a P^1-predicate and $Q_k = \exists_{\rm p}$ if k is odd and $Q_k = \forall_{\rm p}$ if k is even (this notation will be used through out the paper). Now the relativized polynomial-time hierarchy can be defined as follows:

$$\Sigma_k^P(A) = \{L|\ (\exists \Sigma_k^{P,1}\text{-predicate } \sigma)[x \in L \iff \sigma(A;x)]\},$$
$$\Pi_k^P(A) = \{L|\ \overline{L} \in \Sigma_k^P(A)\}, \qquad k \geq 1.$$

In addition to the polynomial-time hierarchy, we are also interested in complexity classes defined by probabilistic machines. Here we avoid the precise definition of probabilistic machines but use probabilistic quantifiers to define the related complexity classes. Let q be a polynomial and σ a predicate of a set variable and a string variable. We write $(\exists^+ y, |y| \leq q(|x|))\ \sigma(A;\langle x,y\rangle)$ to denote the predicate which states that for more than 3/4 of strings y, $|y| \leq q(|x|)$, $\sigma(A;\langle x,y\rangle)$ is true. When the polynomial q is known or is irrelevant, we use the abbreviation $\exists_{\rm p}^+ y$ for $(\exists^+ y, |y| \leq q(|x|))$.

The most important probabilistic complexity classes are the class R of sets computable by polynomial time probabilistic machines with one-sided errors and the class BPP of sets computable by polynomial time probabilistic machines with bounded two-sided errors. The class BPP can be naturally generalized to the BP operator. Namely, for every complexity class $\mathcal{C}$, we may define $BP\mathcal{C}$ as follows: a set L is in $BP\mathcal{C}$ if there exists a set $B \in \mathcal{C}$ such that for all x, $x \in L \Rightarrow (\exists_{\rm p}^+ y)[\langle x,y\rangle \in B]$

and $x \notin L \Rightarrow (\exists^+_{\rm p} y)[\langle x, y\rangle \notin B]$. In particular, we are interested in the following relativized probabilistic polynomial hierarchy:

$$\begin{aligned} R(A) = \{L|\ & (\exists P^1\text{-predicate } \sigma)[x \in L \Rightarrow (\exists^+_{\rm p} y)\ \sigma(A; \langle x, y\rangle)] \\ & \text{and } [x \notin L \Rightarrow (\forall_{\rm p} y) \text{ not } \sigma(A; \langle x, y\rangle)]\}. \\ BP\Sigma^P_k(A) = \{L|\ & (\exists \Sigma^{P,1}_k\text{-predicate } \sigma)[x \in L \Rightarrow (\exists^+_{\rm p} y)\ \sigma(A; \langle x, y\rangle)] \\ & \text{and } [x \notin L \Rightarrow (\exists^+_{\rm p} y) \text{ not } \sigma(A; \langle x, y\rangle)]\}, \qquad k \geq 0. \end{aligned}$$

We let $BPH(A) = \cup_{k=0}^{\infty} BP\Sigma^P_k(A)$. It is clear that for all oracles $P(A) \subseteq R(A) \subseteq \Sigma^P_1(A)$, and $\Sigma^P_k(A) \subseteq BP\Sigma^P_k(A) \subseteq \Pi^P_{k+1}(A)$ for all $k \geq 0$. Figure 1 shows the relations between these classes. In Section 4, we will prove some separation results to show that most of these relations are the best we know to hold for all oracles.

The above complexity classes are formed by adding an $\exists^+_{\rm p}$-quantifier to the alternating $\exists_{\rm p}$- and $\forall_{\rm p}$-quantifiers over polynomial predicates. We may also consider the hierarchy formed by alternating the $\exists^+_{\rm p}$-quantifiers with the $\exists_{\rm p}$-quantifiers over polynomial predicates. The resulting hierarchy is the Arthur-Merlin hierarchy of Babai [1985] or, equivalently, the interactive proof systems of Goldwasser, Micali and Rackoff [1985]. We will follow the literature and use the term "Arthur-Merlin."

$$\begin{aligned} AM_k(A) = \{L|\ & (\exists P^1\text{-predicate } \sigma) \\ & [x \in L \Rightarrow (\exists^+_{\rm p} y_1)(\exists_{\rm p} y_2) \cdots (Q'_k y_k)\ \sigma(A; \langle x, y_1, y_2, \cdots, y_k\rangle) \text{ and} \\ & [x \notin L \Rightarrow (\exists^+_{\rm p} y_1)(\forall_{\rm p} y_2) \cdots (Q''_k y_k)\ \sigma(A; \langle x, y_1, y_2, \cdots, y_k\rangle)\}, \\ MA_k(A) = \{L|\ & (\exists P^1\text{-predicate } \sigma) \\ & [x \in L \Rightarrow (\exists_{\rm p} y_1)(\exists^+_{\rm p} y_2) \cdots (Q'_{k+1} y_k)\ \sigma(A; \langle x, y_1, y_2, \cdots, y_k\rangle) \text{ and} \\ & [x \notin L \Rightarrow (\forall_{\rm p} y_1)(\exists^+_{\rm p} y_2) \cdots (Q''_{k+1} y_k)\ \sigma(A; \langle x, y_1, y_2, \cdots, y_k\rangle)\}, \end{aligned}$$

where Q'_k is $\exists^+_{\rm p}$ if k is odd and is $\exists_{\rm p}$ if k is even, and Q''_k is $\exists^+_{\rm p}$ if k is odd and is $\forall_{\rm p}$ if k is even. This hierarchy is one of the very few that were known to collapse. It is known that $AM_1(A) = BPP(A)$, $MA_1(A) = NP(A)$, $MA_2(A) \subseteq AM_2(A)$, and $AM_k(A) = MA_k(A) = AM_2(A) = BP\Sigma^P_1(A)$ for all oracles A if $k \geq 3$. For more discussions about the complexity classes definable by the $\exists^+_{\rm p}$ quantifier and the AM-hierarchy, see Zachos [1986].

It is well known that the class $PSPACE(A)$ can also be defined using alternating $\exists_{\rm p}$- and $\forall_{\rm p}$-quantifiers. That is, a set L is in $PSPACE(A)$ iff there exist a polynomial q and a P^1-predicate σ such that for all x, $x \in L$ iff

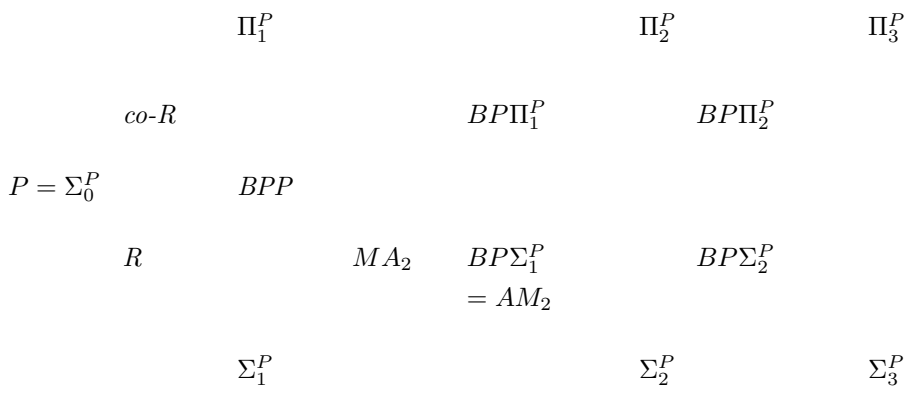

Figure 1. Polynomial and probabilistic polynomial hierarchies.

$(\exists_{\mathrm{p}} y_1)(\forall_{\mathrm{p}} y_2)\cdots(Q_{q(|x|)}y_{q(|x|)})\ \sigma(A;\langle x, y_1, y_2, \cdots y_{q(|x|)}\rangle)$. This suggests that if we replace the fixed integer k by a fixed polynomial bound $q(n)$ where n is the length of the input, then we obtain a generalized polynomial hierarchy which locates between the polynomial hierarchy and $PSPACE(A)$. More formally, we define, for any function $f(n)$ such that $f(n) \leq q(n)$ for some polynomial q, the following class:

$$\Sigma_{f(n)}^P(A) = \{L|\ (\exists P^1\text{-predicate } \sigma)(\forall x)[x \in L \iff (\exists_{\mathrm{p}} y_1)(\forall_{\mathrm{p}} y_2)\cdots(Q_{f(|x|)}y_{f(|x|)})\ \sigma(A;\langle x, y_1, \cdots y_{f(|x|)}\rangle)]\}.$$

Similar generalization can be done on the AM-hierarchy. However, the equivalence between the generalization by quantifiers and the generalization by machine models is no longer trivial. We postpone this definition to Section 7.

2. Oracle Computation and Circuits

The construction of oracles separating or collapsing hierarchies usually involves counting arguments about the computation of oracle machines. When a complex machine model, such as the nondeterministic machine or the alternating machine, is used, the computation tree is often too complicated to be comprehended. Mathematical induction has been used to simplify the arguments about the computation trees. Still, from time to time, the computation trees become so complicated that they seem beyond our understanding (cf. Baker and Selman [1979]). In addition to the mathematical induction, an important progress on simplifying the arguments

about computation trees of oracle machines is to view the oracle computation tree as a circuit. This technique translates unstructured oracle computation trees into more structured, more uniform circuit computation trees, and facilitates the lower bound arguments. We illustrate this technique in this section.

A circuit is usually defined as a directed acyclic graph. For our purpose, we simply define it as a rooted tree. Each interior node of the tree is attached with a gate, and has an unlimited number of child nodes. Three types of gates will be used here: the AND gate, the OR gate and the MAJ gate (standing for "majority"). The MAJ gate outputs 1 if at least 3/4 of inputs are 1, outputs 0 if at least 3/4 of inputs are 0, and is undefined otherwise (for convenience, we say the undefined output is ?). Each leaf is attached with a constant 0, a constant 1, a variable x, or a negated variable $\bar{x}$. That is, all negation gates are moved down to the bottom by De Morgan's law. Each circuit C having only OR and AND gates has a dual circuit $\tilde{C}$ which can be defined inductively as follows: the dual of a constant or a variable is its negation, the dual of a circuit C which is an OR (or, AND) of n children C_i, $1 \leq i \leq n$, is the AND (or, respectively, OR) of $\tilde{C}_i$, $1 \leq i \leq n$.

Each circuit computes a function on its variables. In this paper, each variable is represented by v_z for some string $z \in \{0,1\}^*$. Let V be the set of variables occurred in a circuit C. Then a *restriction* ρ of C is a mapping from V to $\{0,1,*\}$. For each restriction ρ of C, $C\lceil_\rho$ denotes the circuit C' obtained from C by replacing each variable x with $\rho(x) = 0$ by 0 and each y with $\rho(y) = 1$ by 1. Assume that ρ' is a restriction of $C\lceil_\rho$. We write $C\lceil_{\rho\rho'}$ to denote $(C\lceil_\rho)\lceil_{\rho'}$. We also write $\rho\rho'$ to denote the combined restriction on C with values $\rho\rho'(x) = \rho(x)$ if $\rho(x) \neq *$ and with values $\rho\rho'(x) = \rho'(x)$ if $\rho(x) = *$. If a restriction ρ of C maps no variable to $*$, then we say ρ is an *assignment* of C. Let ρ be a restriction of C, we say that ρ *completely determines* C if $C\lceil_\rho$ computes a constant function 0 or 1. An assignment ρ of C always completely determines the circuit C. Note that we represent each variable v_z by a string $z \in \{0,1\}^*$. Therefore, for every set $A \subseteq \{0,1\}^*$, there is a natural assignment ρ_A on all variables v_z, $z \in \{0,1\}^*$: $\rho_A(v_z) = 1$ if $z \in A$ and $\rho_A(v_z) = 0$ if $z \notin A$.

Let M be a polynomial time-bounded deterministic oracle TM, and let x be an input string to M. Without knowing the answers from the oracle A, we can represent the computation of $M^A(x)$ as a tree, where each computation path corresponds to a sequence of possible answers to some queries made by M. Since the runtime of M is

bounded by a polynomial p, each path consists of at most $p(|x|)$ many queries, and there are at most $2^{p(|x|)}$ many paths. Each path ends after $p(|x|)$ moves and either accepts or rejects x. For each accepting path π, let $Y_\pi = \{y|\ y$ is queried in the path π and receives the *yes* answer$\}$, and $N_\pi = \{y|\ y$ is queried in the path π and receives the *no* answer$\}$. Then, for any oracle A, $Y_\pi \subseteq A$ and $N_\pi \subseteq \overline{A}$ imply $M^A(x)$ accepts. More precisely, $M^A(x)$ accepts iff there exists an accepting path π such that $Y_\pi \subseteq A$ and $N_\pi \subseteq \overline{A}$.

Now define the circuit $C_{M,x}$ as follows. Assume that the computation tree of $M^A(x)$ has r accepting paths. Then, $C_{M,x}$ has a top OR gate with r children, each being an AND gate and corresponding to an accepting path of the computation tree of $M^A(x)$. For each path π, the corresponding AND gate has the following children: $\{v_y|\ y \in Y_\pi\} \cup \{\overline{v}_y|\ y \in N_\pi\}$. Then, circuit $C_{M,x}$ satisfies the following property: $C_{M,x}\lceil_{\rho_A}$ outputs 1 iff $M^A(x)$ accepts. In summary, we have

Lemma 2.1. Let M be an oracle TM with runtime $\leq p(n)$. Then, for each x, there is a depth-2 circuit $C = C_{M,x}$ satisfying the following properties:

(a) C is an OR of ANDs,

(b) the top fanin of C is $\leq 2^{p(|x|)}$ and the bottom fanin of C is $\leq p(|x|)$, and

(c) for any set A, $C\lceil_{\rho_A} = 1$ iff $M^A(x)$ accepts.

Remark 2.2. The above also holds if we require that the circuit C be an AND of ORs. This can be seen by considering the machine M' which computes the complement of $L(M, A)$, and noting that the dual circuit of the circuit $C_{M',x}$ satisfies properties (b) and (c).

From the above basic relation, we can derive relations regarding other complexity classes. Let τ be a $\Sigma_k^{P,1}$-predicate; that is, $\tau(A; x) \equiv (\exists_{\rm p} y_1) \cdots (Q_k y_k)$ $\sigma(A; \langle x, y_1, \cdots, y_k\rangle)$ for some P^1-predicate σ. Let q be a polynomial bounding the length of y_i, $1 \leq i \leq k$, as well as the runtime of the oracle machine for σ. When translating this predicate into a circuit, it is natural to identify an AND gate with a $\forall_{\rm p}$-quantifier and an OR gate with an $\exists_{\rm p}$-quantifier. We call a depth-$(k+1)$ circuit a Σ_k-*circuit* if it has alternating OR and AND gates, starting with a top OR gate. A Σ_k-circuit is called a $\Sigma_k(m)$-*circuit* if its fanins are $\leq 2^m$ and the bottom fanins are $\leq m$. (Note that a Σ_k-circuit has depth $k+1$ rather than k, because a $\Sigma_k^{P,1}$-predicate corresponds to a depth-$(k+1)$ circuit.) A Π_k-*circuit* is the dual circuit of a Σ_k-circuit and a $\Pi_k(m)$-*circuit* is the dual circuit of a $\Sigma_k(m)$-circuit. Then we have the following relations.

Lemma 2.3. Let $k \geq 1$. For every $\Sigma_k^{P,1}$-predicate τ there is a polynomial q such that for every x, there exists a $\Sigma_k(q(|x|))$-circuit $C_{\tau,x}$, having the property that for any set A, $C_{\tau,x}\lceil_{\rho_A} = 1$ iff $\tau(A;x)$ is true. The similar relation holds between $\Pi_k^{P,1}$-predicates and Π_k-circuits.

Note that the depth of the above circuit is $k+1$ rather than $k+2$ because the gate corresponding to the last quantifier can always be combined with the top gate of the circuit $C = C_{\sigma,\langle x,y_1,\cdots y_k\rangle}$: depending upon whether the last quantifier is an $\exists_{\rm p}$ (when k is odd) or a $\forall_{\rm p}$ (when k is even), the circuit C may be made to be an OR of ANDs (as in Lemma 2.1) or an AND of ORs (as in Remark 2.2), respectively. Also note that in Lemma 2.3, the circuit $C_{\tau,x}$ exists for every x, therefore we may let the parameter k be a function of x. In other words, the above lemma generalizes immediately to the generalized polynomial hierarchy $\Sigma_{f(n)}^P(A)$.

We can also extend the above lemma to the BP-hierarchy. For each $\exists_{\rm p}^+$-quantifier, the natural corresponding gate is a MAJ gate. Hence, we have the following relation between circuits with MAJ gates and probabilistic complexity classes. A circuit C is a $BP\Sigma_k(m)$-*circuit* if it is an MAJ gate having $\leq 2^m$ many $\Sigma_k(m)$-circuits as children.

Lemma 2.4. For every $BP\Sigma_k^{P,1}$-predicate τ there is a polynomial q such that for every x, there exists a $BP\Sigma_k(q(|x|))$-circuit $C_{\tau,x}$, having the property that for any set A, $C_{\tau,x}\lceil_{\rho_A} = 1$ if $\tau(A;x)$ is true and $C_{\tau,x}\lceil_{\rho_A} = 0$ if $\tau(A;x)$ is false.

3. Diagonalization

Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two complexity classes. A standard proof technique of constructing oracles A to separate $\mathcal{C}_1$ from $\mathcal{C}_2$ (so that $\mathcal{C}_1(A) \not\subseteq \mathcal{C}_2(A)$) is the technique of diagonalization. In a proof by diagonalization, we consider a set $L_A \in \mathcal{C}_1(A)$ against all sets in $\mathcal{C}_2(A)$ and, at each stage, extend set A on a finite number of strings so that L_A is not equal to the specific set in $\mathcal{C}_2(A)$ under consideration in this stage. Therefore, after we considered all sets in $\mathcal{C}_2(A)$, it is established that L_A is not in $\mathcal{C}_2(A)$. We illustrate this technique in this section.

We begin by looking at a simple example: separating the class $\mathcal{C}_1 = NP = \Sigma_1^P$ from the class $\mathcal{C}_2 = co\text{-}NP = \Pi_1^P$. First, we need to enumerate all sets in $co\text{-}NP(A)$. We do it by considering an enumeration of all $\Sigma_1^{P,1}$-predicates $\{\sigma_i\}$. We assume that

the ith $\Sigma_1^{P,1}$-predicate $\sigma_i(A;x)$ is of the form $(\exists y, |y| \leq q(|x|))\tau_i(A;\langle x,y\rangle)$ such that τ_i is a P^1-predicate whose runtime is bounded by the ith polynomial $p_i(|x|)$ and that $q(n) \leq p_i(n)$. Let

$$L_A = \{0^n |\ (\exists y, |y| = n)\ y \in A\}.$$

Then, for all A, $L_A \in NP(A)$. We will construct set A by stages such that in stage n, the following requirement R_n is satisfied:

R_n: $(\exists x_n)\ [x_n \in L_A \iff \sigma_n(A;x_n)]$.

Observe that for every set $B \in \Pi_1^P(A)$, there must be a $\Sigma_1^{P,1}$-predicate σ_i such that for all x, $x \in B$ iff *not* $\sigma_i(A;x)$. Now, if requirement R_i is satisfied then $x_i \in L_A \iff \sigma_i(A;x_i) \iff x_i \notin B$, and hence $L_A \neq B$. Thus, if all requirements R_n are satisfied, then $L_A \notin \Pi_1^P(A)$.

We now describe the construction of set A. Set A will be constructed by stages. In each stage, we will reserve some strings for set A and some strings for set $\overline{A}$. We let $A(n)$ and $A'(n)$ be sets containing those strings reserved for A and $\overline{A}$, respectively, up to stage n. We begin with sets $A(0) = A'(0) = \emptyset$. In stage n, we try to satisfy requirement R_n. Assume that before stage n we have defined $t(n-1)$ such that $A(n-1) \cup A'(n-1) \subseteq \Sigma^{\leq t(n-1)}$. We let m be the least integer greater than $t(n-1)$ such that $2^m > p_n(m)$, and let $x_n = 0^m$. Now we consider the computation tree generated by $\sigma_n(A;x_n)$, with the following modification: if the tree contains a query "$y \in ?A$" with $|y| < m$, then prune the *no* child if $y \in A(n)$ and prune the *yes* child if $y \notin A(n)$. Thus we obtain a computation tree of queries "$y \in ?A$" only if $|y| \geq m$. Consider two cases:

Case 1. There exists an accepting path π in the tree. Then, this accepting path π is of length at most $p_n(m)$ and hence asks at most $p_n(m)$ many queries about strings y of length m. Since $2^m > p_n(m)$, there must be at least one string z of length m not being queried in the path π. We fix such a string z_0. Also, let $B_0 = \{y|\ y$ is queried and answered *no* in path $\pi\}$, and $B_1 = \{y|\ y$ is queried and answered *yes* in path $\pi\}$. We define $A(n) = A(n-1) \cup B_1 \cup \{z_0\}$ and $A'(n) = A'(n-1) \cup B_0$. Note that requirement R_n is satisfied by set $A(n)$ and string x_n: $x_n \in L_{A(n)}$ because $|z_0| = m$ and $z_0 \in A(n)$, and $\sigma_n(A(n);x_n)$ because $B_1 \subseteq A(n)$ and $B_0 \cap A(n) = \emptyset$ imply that the computation of $\sigma_n(A(n);x_n)$ follows the path π and accepts.

Case 2. All paths in the tree reject. Then, consider the path π all of whose queries "$y \in ?A$" receive the answer *no*. Let $B_0 = \{y|\ y$ is queried in the path $\pi\}$. We let $A(n) = A(n-1)$ and $A'(n) = A'(n-1) \cup B_0$. Note that requirement R_n is

satisfied by set $A(n)$ and string x_n: $x_n \notin L_{A(n)}$ because $A(n) \cap \{0,1\}^m = \emptyset$, and *not* $\sigma_n(A(n); x_n)$ because $B_0 \cap A(n) = \emptyset$ implies that the computation of $\sigma_n(A(n); x_n)$ follows the path π and rejects.

Finally we complete stage n by letting $t(n) = p_n(m)$. Note that for all y in $A(n) \cup A'(n)$, $|y| \leq t(n)$ (assuming $p_n(m) > m$).

We define set A to be $\cup_{n=0}^{\infty} A(n)$. Note that the set A has the property that $A^{\leq t(n)} = A(n)$. Thus, the requirement R_n is satisfied by set A and string x_n because both computations of $x_n \in L_A$ and $\sigma_n(A; x_n)$ involve only with strings of length $\leq t(n)$ and so $A^{\leq t(n)} = A(n)$ implies that requirement R_n is satisfied by A and x_n.

In the above we have described in detail the construction of a set A such that $NP(A) \neq co\text{-}NP(A)$. Now we re-examine the construction and make the following observations about its general properties.

(1) We need an enumeration of sets in $\mathcal{C}_2$. This is usually simple. For the classes we defined in Section 1, all of them have simple representations by a number of polynomial length-bounded quantifiers followed by P^1-predicates. Since the class of all P^1-predicates have a simple enumeration, we can enumerate these classes accordingly. Namely, we may assume, for any sequence of polynomial length-bounded quantifiers $Q'_1, \cdots, Q'_k$, an enumeration $\{\sigma_i\}$ of predicates of the form $(Q'_1 y_1) \cdots (Q'_k y_k)\, \tau(A; \langle x, y_1, \cdots, y_k \rangle)$ such that both the runtime of the P^1-predicate τ and the length of y_j's are bounded by the ith polynomial p_i.

(2) The requirement $\mathcal{C}_1(A) \not\subseteq \mathcal{C}_2(A)$ is divided into an infinite number of requirements:

$$R_n\colon\ (\exists x_n)[x_n \in L_A \iff \textit{not } \sigma_n(A; x_n)],$$

where L_A is a fixed set in $\mathcal{C}_1(A)$ and σ_n is the nth predicate in our enumeration of sets in $\mathcal{C}_2(A)$.

(3) Assume that $A(n) = A^{\leq t(n)}$ and let $D(n) = \{y|\ t(n-1) < |y| \leq t(n)\}$. We call set $D(n)$ the *diagonalization region* for stage n. Then, sets $D(n)$ and $A(n)$ satisfy

(a) $x_n \in L_A \iff x_n \in L_{A(n)} \iff x_n \in L_{A \cap D(n)}$, and

(b) $\sigma_n(A; x_n) \iff \sigma_n(A(n); x_n) \iff \sigma_n(A(n-1) \cup (A \cap D(n)); x_n)$.

Therefore, in stage n we essentially only need to satisfy the following simpler requirement R'_n, instead of R_n.

$$R'_n\colon\ (\exists x_n)(\exists B \subseteq D(n))[x_n \in L_B \iff \textit{not } \sigma_n(A(n-1) \cup B; x_n)].$$

(4) Inside the diagonalization region $D(n)$, we need to show that requirement R'_n can be satisfied, usually by a counting argument. We observe that the counting argument used in the above example is essentially equivalent to a lower bound argument for $\Sigma_1(m)$-circuits versus $\Pi_1(p_n(m))$-circuits. More precisely, in stage n, the computation tree of $\sigma_n(A; x_n)$, after pruning to remove queries about strings of length less than m, can be translated to a $\Sigma_1(p_n(m))$-circuit C (Lemma 2.3). Let $\tilde{C}$ be its dual circuit. Then $\tilde{C}$ is a $\Pi_1(p_n(m))$-circuit. Also, the question of whether $x \in L_A$ is equivalent to the predicate $(\exists z, |z| = m)[z \in A]$ and hence can be expressed by a simple $\Sigma_1(m)$-circuit C_0: C_0 is the OR of 2^m variables v_z, $|z| = m$. Now observe that the counting argument in the above example essentially establishes that there is an assignment ρ on variables such that $C_0\lceil_\rho \neq \tilde{C}\lceil_\rho$.

In general, we can see that the requirement R'_n can be reduced to the requirement R''_n about circuits:

R''_n: $(\exists x_n)(\exists B \subseteq D(n))[x_n \in L_B \iff C\lceil_{\rho_B} = 0]$,

where C is the circuit corresponding to predicate $\sigma_n(A; x_n)$, with each variable v_y having $|y| \leq t(n-1)$ replaced by value $\chi_{A(n-1)}(y)$. Depending upon which class $\mathcal{C}_1$ is, the predicate $x_n \in L_B$ may also be represented by a circuit C_0. In this case, requirement R''_n states that $(\exists x_n)(\exists B \subset D(n))[C_0\lceil_{\rho_B} \neq C\lceil_{\rho_B}]$.

The above discussion shows that whenever possible, the diagonalization process is reduced to a lower bound problem about circuits. In the next section, we will see more examples using the above setting of diagonalization.

4. Separation Results

In this section, we use the general setting of diagonalization discussed in Section 3 to separate some complexity hierarchies defined in Section 1 by oracles.

4.1. Separating PSPACE from PH

First, we consider the case when $\mathcal{C}_1 = PSPACE$ and $\mathcal{C}_2 = \Sigma_k^P$, for arbitrary $k \geq 1$. We set up the following items:

(a) $L_A = \{0^n | \ \|A^{=n}\| \text{ is odd}\} \in PSPACE(A)$.

(b) For each n, let m be a sufficiently large integer greater than $t(n-1)$ (exact bound for m to be determined later). Let $x_n = 0^m$ and $t(n) = p_n(m)$, and recall that $D(n) = \{y|\ t(n-1) < |y| \leq t(n)\}$.

(c) For each n, let σ_n be the nth $\Sigma_k^{P,1}$-predicate and let C_n be the $\Sigma_k(p_n(m))$-circuit such that for all sets $B \subseteq D(n)$, $C_n\lceil_{\rho_B} = 1$ iff $\sigma_n(A(n-1) \cup B; x_n)$. (That is, C_n is obtained from predicate $\sigma_n(A; x_n)$ by Lemma 2.3 with the modification that each variable v_y is assigned value $\chi_{A(n-1)}(y)$ if $|y| < m$.)

Now the separation problem for *PSPACE* versus Σ_k^P is reduced to the following requirement:

R_n'': $(\exists B \subseteq D(n))$ $[\|B^{=m}\|$ is odd $\iff C_n\lceil_{\rho_B} = 0]$.

Or, equivalently, it is reduced to the following lower bound result on parity circuits. For each $k \geq 1$, let $s_k(n)$ be the minimum size r of a depth-k circuit[4] which has size r and bottom fanin $\leq \log r$ and computes the (odd) parity of n variables. Requirement R_n'' states that the function $s_k(n)$ grows faster than the functions $2^{(\log n)^k}$ for all k. In the following we show an even stronger result: $s_k(n)$ is greater than an exponential function on n. This is the main breakthrough in the theory of relativization.

Theorem 4.1 [Yao, 1985; Hastad, 1987]. For sufficiently large n, $s_k(n) \geq 2^{(1/10)n^{1/(k-1)}}$.

Since the proof of this theorem is quite involved, we will only include a sketch of the main ideas here. The interested reader is referred to Hastad [1987] for the complete proof.

Proof of Theorem 4.1. The proof is done by induction on k. A stronger statement is easier to use in the induction proof. More precisely, we are going to show the following stronger form of the theorem:

Induction statement. Let $k \geq 2$ and $\delta = 1/10$. Let C_n be a depth-k circuit having $\leq 2^{\delta n^{1/(k-1)}}$ gates not at the bottom level and having bottom fanin $\leq \delta n^{1/(k-1)}$. Then, for sufficiently large n, C_n does not compute the parity of n variables.

The base case of $k = 2$ is very easy to see as no depth-2 circuit having bottom fanin $\leq n-1$ computes the parity of n variables.

For the inductive step, assume that $k > 2$. By way of contradiction, we assume that there exists a circuit C_n of the above form that computes the parity of

[4] In this and the next subsections, all circuits are circuits that have only OR and AND gates.

n variables. To apply the inductive hypothesis, we need to show that there exists a restriction ρ such that it leaves m variables v unassigned (i.e., $\rho(v) = *$) but makes circuit $C_n\lceil_\rho$ to be equivalent to a depth-$(k-1)$ circuit having $\leq 2^{\delta m^{1/(k-1)}}$ gates not at the bottom level and having bottom fanin $\leq \delta m^{1/(k-1)}$. Then, the assumption that C_n computes the parity of n variables implies that $C_n\lceil_\rho$ (or its dual circuit) computes the parity of m variables, which leads to a contradiction to the inductive hypothesis.

To this end, we consider the following probability space R_p of restrictions on the n variables: a random restriction ρ from R_p satisfies that $\Pr[\rho(v) = *] = p$ and $\Pr[\rho(v) = 0] = \Pr[\rho(v) = 1] = (1-p)/2$, where p is a parameter, $0 \leq p \leq 1$. Then, we consider the circuit $C_n\lceil_\rho$ resulted from applying a random restriction ρ from R_p to C_n.

If we take p, for instance, to be $n^{-1/(k-1)}$, then the expected number of variables assigned with $*$ by ρ is $n^{(k-2)/(k-1)}$. Therefore, for probability $\geq 1/3$, a random restriction ρ will leave more than $n^{(k-2)/(k-1)}$ variables unassigned. To apply the inductive hypothesis, we need only to show that the probability is greater than $2/3$ that the circuit $C_n\lceil_\rho$ is equivalent to a depth-$(k-1)$ circuit having $\leq 2^{\delta r}$ many gates not at the bottom level and having bottom fanin $\leq \delta r$, where $r = (n^{(k-2)/(k-1)})^{1/(k-2)} = n^{1/(k-1)}$, and ρ is a random restriction from R_p with $p = n^{-1/(k-1)}$. This is the consequence of the following Switching lemma:

Lemma 4.2 (Switching Lemma I). Let G be a depth-2 circuit such that G is the AND of ORs with bottom fanin $\leq t$ and ρ a random restriction from R_p. Then, the probability that $G\lceil_\rho$ is not equivalent to an OR of ANDs with bottom fanin $\leq s$ is bounded by α^s, where α satisfies $\alpha \leq 5pt$. The above also holds if G is an OR of ANDs to be converted to a circuit of AND of ORs.

The proof of the Switching Lemma is too complicated to be included here. We omit it and continue the induction proof for Theorem 4.1. Without loss of generality, assume that k is odd and so the bottom gate of C_n is an OR gate. Let $s = t = \delta n^{1/(k-1)}$. Note that C_n has $\leq 2^s$ many depth-2 subcircuits. From the Switching Lemma, the probability that every bottom depth-2 subcircuit G of $C_n\lceil_\rho$ (which is an AND of ORs) is equivalent to a depth-2 circuit of OR of ANDs, with bottom fanins $\leq s$, is

$$\geq (1-\alpha^s)^{2^s} \geq 1 - (2\alpha)^s.$$

Now, $\alpha < 5pt = 1/2$ and so $\lim_{n\to\infty}(2\alpha)^s = 0$. In particular, there exists an integer

n_k such that if $n > n_k$ then $(2\alpha)^s < 1/3$. (This integer n_k depends only on k.) Thus, for sufficiently large n, the probability is greater than $1 - (2\alpha)^s \geq 2/3$ that $C_n\lceil_\rho$ is equivalent to a depth-$(k-1)$ circuit having $\leq 2^{\delta n^{1/(k-1)}}$ gates not at the bottom level and having bottom fanin $\leq \delta n^{1/(k-1)}$. This completes the proof of Theorem 4.1. □

Remark 4.3. The above gave the lower bound $s_k(n) \geq 2^{(1/10)n^{1/(k-1)}}$ on the size r of Σ_k-circuits C for parity if C must have bottom fanin $\leq \log r$. We may treat a Σ_k-circuit as a Σ_{k+1}-circuit with bottom fanin 1, and so we obtain the lower bound $s'_k(n) \geq 2^{(1/10)n^{1/k}}$ on the size of any Σ_k-circuit for parity. Hastad [1987] has pointed out that we may also first apply the Switching Lemma to shrink the bottom fanin to $\log r$ and so obtain a better bound $s'_k(n) \geq 2^{\epsilon n^{1/(k-1)}}$, where $\epsilon = 10^{-k/(k-1)}$.

From Theorem 4.1, it is clear that if we choose integer m to be large enough such that Theorem 4.1 holds for $s_k(2^m)$ and that $(1/10)2^{m/(k-1)} > k \cdot p_n(m)$ (note that $2^{k \cdot p_n(m)}$ bounds the size of a $\Sigma_k(p_n(m))$-circuit), then requirement R''_n can be satisfied. By dovetailing the enumeration of $\Sigma_k^{P,1}$-predicates for all $k \geq 1$, we obtain the following theorem.

Theorem 4.4. There exists an oracle A such that for every $k \geq 1$, $PSPACE(A) \neq \Sigma_k^P(A)$.

Let $\oplus P(A)$ (read "parity-$P(A)$) be the complexity class defined as follows: a set L is in $\oplus P(A)$ iff there exists a P^1-predicate σ such that for all x, $x \in L$ iff the number of y, $|y| \leq q(|x|)$, such that $\sigma(A; \langle x, y\rangle)$ holds is odd. It is easy to see that $\oplus P(A) \subseteq PSPACE(A)$ for all A, but it is not known whether $NP \subseteq \oplus P$ or $\oplus P \subseteq \Sigma_k^P$ for any $k \geq 1$. It is easy to see that the set L_A in the above proof is in $\oplus P(A)$. That is, we have actually established a stronger separation result.

Corollary 4.5. There exists a set A such that $\oplus P(A) \not\subseteq \Sigma_k^P(A)$ for all $k \geq 0$.

4.2. Separating PH

In this subsection, we show that there is an oracle A such that $PH(A)$ is an infinite hierarchy. It is sufficient to find, for each $k > 0$, a set $L_A \in \Sigma_k^P(A)$ such that $L_A \notin \Pi_k^P(A)$. Then, by dovetailing the diagonalization for each $k > 0$, the oracle A can be constructed so that $\Sigma_k^P(A) \neq \Pi_k^P(A)$ for all k.

Let $k > 0$ be fixed. Define

$$L_A = \{0^{(k+1)n} |\ (\exists y_1, |y_1| = n)(\forall y_2, |y_2| = n) \cdots (Q_k y_k, |y_k| = n)\ 0^n y_1 y_2 \cdots y_k \in A\}.$$

Then, clearly, $L_A \in \Sigma_k^P(A)$. The setup for m, $t(n)$ and $D(n)$ is similar to that in Section 4.1. (In particular, m must be *sufficiently* large.) From the general setting discussed in Section 3, we only need to satisfy the requirements

$$R_n'': (\exists x_n = 0^m)(\exists B \subseteq D(n))[0^m \in L_B \iff C\lceil_{\rho_B} = 0],$$

where C is a $\Pi_k(p_n(|m|))$-circuit corresponding to the nth $\Pi_k^{P,1}$-predicate $\sigma_n(A; 0^m)$, with all variables v_y having $|y| \leq t(n-1)$ replaced by the constant $\chi_{A(n-1)}(y)$.

We note that the predicate $0^m \in L_B$ is a $\Sigma_k^{P,1}$-predicate. That is, there is a depth-k circuit C_0 having the following properties:

(a) C_0 has alternating OR and AND gates, starting with a top OR gate,

(b) all the fanins of C_0 are exactly $2^{m/(k+1)}$,

(c) each leaf of C_0 has a unique positive variable v_z with $|z| = m$, and

(d) for all $B \subseteq D(n)$, $C_0\lceil_{\rho_B} = 1 \iff 0^m \in L_B$.

So, the requirement R_n'' is reduced to the following problem on circuits: there exists a set $B \subseteq D(n)$ such that $C_0\lceil_{\rho_B} \neq C\lceil_{\rho_B}$. We restate it in the lower bound form. Let D be a depth-k circuit having the following properties: (a) it has alternating OR and AND gates, with a top OR gate, (b) all its fanins are exactly n, except the bottom fanins which are exact $\sqrt{n}$, and (c) each leaf of D has a unique positive variable. We let the function f_k^n be the function computed by D. (Note that C_0 contains a subcircuit computing a function $f_k^{2^{m/(k+1)}}$.) Let $s_k(n)$ be the minimum r of a Π_k-circuit computing a function f_k^n that has size $\leq r$ and bottom fanin $\leq \log r$. The new requirement R_n'' is satisfied by the following lower bound on $s_k(n)$.

Theorem 4.6. For sufficiently large n, $s_k(n) \geq 2^{(1/12)n^{1/3}}$.

Sketch of proof. Again, we prove it by induction. The induction is made easier on the following stronger form.

Induction statement. Let $k \geq 1$ and $\delta = 1/12$. Let C_n be a Π_k-circuit having $\leq 2^{\delta n^{1/3}}$ gates not at the bottom level and having bottom fanin $\leq \delta n^{1/3}$. Then, for sufficiently large n, C_n does not compute the function f_k^n.

The base step of the induction involves two circuits: C_n is an AND of ORs with small bottom fanins ($\leq \delta n^{1/3}$), and C_0 (which computes the function f_1^n) is an OR of $n^{1/2}$ variables. We need to show that C_n does not compute the same function as C_0. But this is exactly what we proved in the example of Section 3.

For the inductive step, consider $k > 1$. We define two new probability spaces of restrictions: $R_{q,\mathcal{B}}^+$ and $R_{q,\mathcal{B}}^-$, where $\mathcal{B} = \{B_j\}_{j=1}^r$ is a partition of variables and q a

value between 0 and 1. To define a random restriction ρ in $R^+_{q,\mathcal{B}}$, first, for each B_j, $1 \leq j \leq r$, let $s_j = *$ with probability q and $s_j = 0$ with probability $1-q$; [5] and then, independently, for each variable $x \in B_j$, let $\rho(x) = s_j$ with probability q and $\rho(x) = 1$ with probability $1-q$. Next, define, for each $\rho \in R^+_{q,\mathcal{B}}$, a restriction $g(\rho)$: for all B_j with $s_j = *$, let V_j be the set of all variables in B_j which are given value $*$ by ρ; $g(\rho)$ selects one variable y in V_j and gives value $*$ to y and value 1 to all others in V_j. The probability space $R^-_{q,\mathcal{B}}$ and $g(\rho)$ are defined by interchanging the roles played by 0 and 1.

Now let $\mathcal{B} = \{B_j\}$ be the partition of variables in C_0 such that each B_j is the set of all variables leading to a bottom gate in C_0. Let $q = n^{-1/3}$. If k is even, then apply a random restriction ρ from $R^+_{q,\mathcal{B}}$; otherwise, apply a random restriction ρ from $R^-_{q,\mathcal{B}}$.

In order to apply the inductive hypothesis, we need to verify that (a) with a high probability, $C\lceil_{\rho g(\rho)}$ is equivalent to a Π_{k-1}-circuit having $\leq 2^{\delta n^{1/3}}$ gates not at the bottom level and having bottom fanin $\leq \delta n^{1/3}$, and (b) with a high probability, $C_0\lceil_{\rho g(\rho)}$ contains a subcircuit computing the function f^n_{k-1}. We give sketches of these facts.

To prove part (a), we need a new version of the Switching Lemma.

Lemma 4.7 (Switching Lemma II). Let G be an AND of ORs with bottom fanin $\leq t$, and $\mathcal{B} = \{B_j\}$ be a partition of variables in G. 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 $\leq s$ is bounded by α^s, where $\alpha < 6qt$. The above also holds with $R^+_{q,\mathcal{B}}$ replaced by $R^-_{q,\mathcal{B}}$, or with G being an OR of ANDs to be converted to a circuit of an AND of ORs.

We again omit the proof of the second Switching Lemma; the interested reader is referred to Hastad [1987] for details.

Without loss of generality, assume that k is even. From Lemma 4.7, if we choose $s = t = \delta n^{1/3}$, then we have $\alpha < 6qt = 1/2$, and so $\lim_{n\to\infty}(2\alpha)^s = 0$. Therefore, the probability that every bottom depth-2 subcircuit $G\lceil_{\rho g(\rho)}$ (which is an AND of ORs) of $C\lceil_{\rho g(\rho)}$ is equivalent to a depth-2 circuit of OR of ANDs is

$$\geq (1-\alpha^s)^{2^s} \geq 1-(2\alpha)^s \geq 2/3,$$

[5] Do not confuse the s_j here (which is a value in $\{0,1,*\}$) with the size function $s_k(n)$.

for sufficiently large n. The above proved part (a). The following lemma proves part (b).

Lemma 4.8. For sufficiently large n, the probability that $C_0\lceil_{\rho g(\rho)}$ contains a subcircuit for f^n_{k-1} is greater than $2/3$.

Sketch of proof. We want to show that $q_1 q_2 \geq 2/3$, where q_1 is the probability that all bottom AND gates $H_j\lceil_{\rho g(\rho)}$ (corresponding to block B_j) of $C_0\lceil_{\rho g(\rho)}$ takes the value s_j, and q_2 is the probability that all OR gates at level 2 from bottom of $C_0\lceil_{\rho g(\rho)}$ have $\geq n^{1/2}$ many child nodes $H_j\lceil_{\rho g(\rho)}$ of AND gates having value $s_j = *$.

To estimate probability q_1, we note that for a fixed j,

$$\begin{aligned}\Pr[H_j\lceil_{\rho g(\rho)} \text{ has value } \neq s_j] &= \Pr[\text{all inputs to } H_j\lceil_{\rho g(\rho)} \text{ are 1}] \\ &= (1-q)^{n^{1/2}} = (1-n^{-1/3})^{n^{1/2}} < e^{-n^{1/6}},\end{aligned}$$

for sufficiently large n. Since C_0 has n^{k-1} many bottom gates H_j, we obtain $q_1 \geq (1-e^{-n^{1/6}})^{n^{k-1}} \geq 5/6$, for sufficiently large n.

To estimate probability q_2, let G be a bottom depth-2 subcircuit of C_0, and let r_ℓ be the probability that $G\lceil_{\rho g(\rho)}$ has exactly ℓ many child nodes $H_j\lceil_{\rho g(\rho)}$ having values $s_j = *$. Then,

$$r_\ell = \binom{n}{\ell} q^\ell (1-q)^{n-\ell}.$$

Note that for sufficiently large n, if $\ell \leq 2\sqrt{n}$ then $r_\ell \leq 1/2$ and $r_\ell \geq 2r_{\ell-1}$. So,

$$\sum_{\ell=0}^{\sqrt{n}} r_\ell \leq r_{\sqrt{n}} \cdot \sum_{\ell=0}^{\sqrt{n}} 2^{-\ell} \leq 2 \cdot r_{\sqrt{n}} \leq 2 \cdot 2^{-\sqrt{n}} \cdot r_{2\sqrt{n}} \leq 2^{-\sqrt{n}}.$$

Thus, $q_2 \geq (1-2^{-\sqrt{n}})^{n^{k-2}} > 5/6$, for sufficiently large n. This shows that $q_1 \cdot q_2 \geq 2/3$ and completes the proof for Lemma 4.8 and hence Theorem 4.6. □

Theorem 4.6 implies that requirement R''_n can be satisfied if we choose integer m to be so large that Theorem 4.6 holds for $s_k(2^{m/(k+1)})$ and that $(1/12)2^{m/3(k+1)} > k \cdot p_n(m)$.

Theorem 4.9. There exists an oracle A such that for all $k > 0$, $\Sigma^P_k(A) \neq \Pi^P_k(A)$.

4.3. Separating BPH from PH

We now consider the hierarchy $BPH(A) = \cup_{k=0}^{\infty} BP\Sigma^P_k(A)$. Since $\Sigma^P_k(A) \subseteq BP\Sigma^P_k(A) \subseteq \Pi^P_{k+1}(A)$, it follows immediately that $BP\Sigma^P_{k+1}(A) = BP\Sigma^P_k(A)$ implies that the polynomial hierarchy $PH(A)$ collapses to $BP\Sigma^P_k(A)$. Therefore, there

exists an oracle A such that $BPH(A)$ is infinite. What we are going to show in this section is instead that there exists an oracle A such that $BP\Sigma_k^P(A) \not\subseteq \Sigma_{k+1}^P(A)$; or, in other words, the two hierarchies $PH(A)$ and $BPH(A)$ are completely separated.

We let $L_A = \{0^{(k+2)n} |\ (\exists^+ y_0, |y_0| = n)(\exists y_1, |y_1| = n)(\forall y_2, |y_2| = n)\cdots(Q_k y_k, |y_k| = n)0^n y_0 y_1 \cdots y_k \in A\}$. Then, obviously, $L_A \in BP\Sigma_k^P(A)$. Also, for stage n, we set up $x_n = 0^m$ for appropriate value m and let C be the $\Sigma_{k+1}(p_n(m))$-circuit corresponding to the nth $\Sigma_{k+1}^{P,1}$-predicate $\sigma_n(A; 0^m)$ with variables v_y assigned value $\chi_{A(n-1)}(y)$ if $|y| < m$. To satisfy the requirement

$$R_n'': (\exists x_n = 0^m)(\exists B \subseteq D(n))[0^m \in L_A \iff C\lceil_{\rho_B} = 0],$$

we only need a lower bound result like that in Section 4.2. More precisely, let C_i, $1 \le i \le n$, be circuits computing functions f_k^n, and let C_0 be a depth-$(k+1)$ circuit with a top MAJ gate and having $C_1, \cdots, C_n$ as its n children. Then, we need to show that the minimum size r of a Σ_{k+1}-circuit that has bottom fanin $\le \log r$ and is equivalent to C_0 is at least $2^{(1/12)n^{1/3}}$. If we treat the top MAJ gate of C_0 as an AND gate, then the proof for this lower bound result is almost identical to the proof given in Section 4.2. The only difference is that the base step of the induction proof is no longer simple. In fact, it needs a more complicated counting argument. We state it as a separate lemma.

Lemma 4.10. Let C_0 be a depth-2 circuit having a top MAJ gate with n children, each being an OR of $n^{1/2}$ many variables, and let C be a $\Sigma_3(m)$-circuit, where $m = (1/12)n^{1/3}$. Then, for sufficiently large n, there exists an assignment ρ such that $C_0\lceil_\rho \neq ?$ and $C_0\lceil_\rho \neq C\lceil_\rho$.

Proof. Note that we *cannot* apply a random restriction ρ from $R_{q,\mathcal{B}}^-$ to the circuits and use the second Switching Lemma to simplify the circuits, because such a random restriction ρ would oversimplify circuit C into one that computes a constant 1 because the majority (more than 3/4 many) of the OR gates in $C\lceil_\rho$ would have value $s_j = 1$. Instead of using a random restriction to simplify circuits, we make a direct ad hoc counting argument to show that circuit C cannot simulate circuit C_0. This counting argument was first used by Baker and Selman [1979] to construct an oracle A such that $\Sigma_2^P(A) \neq \Pi_2^P(A)$. We give a sketch here.

Assume, by way of contradiction, that for all assignments ρ, $C_0\lceil_\rho \neq ?$ implies $C_0\lceil_\rho = C\lceil_\rho$. Let V be the set of variables in C_0. Without loss of generality, assume

that all variables in C are in V. Let

$$K = \{\rho |\ (\forall \text{ OR gate } H \text{ of } C_0)(\exists \text{ unique } z \text{ in } H)\ \rho(z) = 1\}.$$

Then, $\|K\| = n^{n/2} = 2^{(n \log n)/2}$. For each $\rho \in K$, $C_0\lceil_\rho = 1$, and hence one of AND gate D of C has $D\lceil_\rho = 1$. Note that there are $\leq 2^m$ AND gates in C. So, there exists an AND gates D of C such that $D\lceil_\rho = 1$ for at least $2^{(n \log n)/2 - m}$ many ρ in K. Fix this AND gate D. Let $K_0 = \{\rho \in K |\ D\lceil_\rho = 1\}$, and $Q_0 = \emptyset$. We note that K_0 and Q_0 satisfy the following properties:

(i) $\|K_\mu\| \geq 2^{(n/2 - \mu/3)\log n - c\mu - m}$, where $c = \log 12$,

(ii) $\|Q_\mu\| = \mu$,

(iii) $\rho \in K_\mu \Rightarrow [\rho \in K_0$ and $\rho(w) = 1$ for all $w \in Q_\mu]$.

We claim that we can find sets K_μ and Q_μ satisfying the above properties for $\mu = 0, \cdots, n/4$. Then, we have $\|K_{n/4}\| \geq 2^{(n/2 - n/12)\log n - cn/4 - m} = 2^{(5/12)n \log n - cn/4 - m}$. However, from property (iii), all $\rho \in K_{n/4}$ have the same value 1 on $n/4$ many variables, and there are only $(n^{1/2})^{3n/4} = 2^{(3/8)n \log n}$ many such ρ's. This gives a contradiction.

To prove the claim, we assume that K_μ and Q_μ have been constructed satisfying the above properties, with $\mu < n/4$. Now, we define $K_{\mu+1}$ and $Q_{\mu+1}$ inductively as follows: First, define ρ_μ by $\rho_\mu(v) = 0$ if $v \in V - Q_\mu$ and $\rho_\mu(v) = 1$ if $v \in Q_\mu$. Then, by property (ii), $\|Q_\mu\| = \mu < n/4$ implies that $C\lceil_{\rho_\mu} = C_0\lceil_{\rho_\mu} = 0$ and hence $D\lceil_{\rho_\mu} = 0$. So, one of the OR gates G in D has $G\lceil_{\rho_\mu} = 0$. Fix this gate G.

Since each ρ in K_μ has $D\lceil_\rho = 1$, we have $G\lceil_\rho = 1$, and so ρ_μ differs from each $\rho \in K_\mu$ at at least one variable in G. This variable u must be in $V - Q_\mu$ and $\rho_\mu(u) = 0$ and $\rho(u) = 1$. Choose such a variable u in G that maximizes the size of the set $\{\rho \in K_\mu |\ \rho_\mu(u) = 0, \rho(u) = 1\}$. Define $Q_{\mu+1} = Q_\mu \cup \{u\}$ and $K_{\mu+1} = \{\rho \in K_\mu |\ \rho(u) = 1\}$. Then it can be verified that $K_{\mu+1}$ and $Q_{\mu+1}$ satisfy properties (i)—(iii). (In particular, G has $\leq m = (1/12)n^{1/3}$ variables, and so

$$\begin{aligned}\|K_{\mu+1}\| &\geq \|K_\mu\|/m \geq 2^{(n/2 - \mu/3)\log n - c\mu - m}/2^{(1/3)\log n + c} \\ &= 2^{(n/2 - (\mu+1)/3)\log n - c(\mu+1) - m}.)\end{aligned}$$

The claim, and hence the lemma, is proven. □

Theorem 4.11. There exists an oracle A such that for every $k > 0$, $BP\Sigma_k^P(A) \not\subseteq \Sigma_{k+1}^P(A)$.

Corollary 4.12. There exists an oracle A such that $MA_2(A) \subsetneq AM_2(A)$.

Proof. It is known that for all oracles A, $MA_2(A) \subseteq \Sigma_2^P(A)$ [Zachos, 1986]. □

5. Encoding and Diagonalization

In order to collapse a hierarchy to a fixed level, a common technique is to encode the information about sets in higher levels into the oracle set A so that it can be decoded using a lower level machine. If the complexity class in the higher level has a complete set,[6] then we need to encode only one set instead of infinitely many sets. For example, Baker, Gill and Solovay [1975] showed that if A is *PSPACE*-complete then $NP(A) = P(A)$ (in fact, $PSPACE(A) = P(A)$). Moreover, when we need to collapse a hierarchy to a fixed level and, using the same oracle, to separate classes below that level, the technique of encoding is used together with diagonalization, and this may create many new problems in constructing the oracle. We illustrate this technique in this section.

We first consider a simple example: collapsing NP to $NP \cap co\text{-}NP$, but keeps $P \neq NP$. That is, we need an oracle A such that $P(A) \neq NP(A) = co\text{-}NP(A)$. First, we follow the general setup in Section 3 for the diagonalization against all sets in P. Namely, we enumerate all P^1-predicate $\sigma_n(A; x)$ and choose appropriate witnesses x_n and diagonalization regions $D(n)$ so that the following requirements are to be satisfied:

$$R_{n,0}\text{: } (\exists x_n)(\exists B \subseteq D(n))[x_n \in L_B \iff \textit{not } \sigma_n(A(n-1) \cup B; x_n)],$$

where $L_A = \{0^{2n} | \ (\exists y, |y| = n)[0^n y \in A]\} \in NP(A)$.

In the meantime, consider a complete set $K(A)$ for $NP(A)$. For example, let $K(A) = \{\langle 0^i, a, 0^j\rangle | \ (\exists b, |b| \leq j)[\sigma_i(A; \langle a, b\rangle)$ is true and decidable in j moves$]\}$. It is important to note that whether an instance $x = \langle 0^i, a, 0^j\rangle$ is in $K(A)$ depends only on the set $A^{<|x|}$ because in j moves the machine for σ_i on input a can only query about strings in $A^{<|x|}$. To satisfy $NP(A) = co\text{-}NP(A)$, we need to make $K(A) \in co\text{-}NP(A)$, or, $\overline{K(A)} \in NP(A)$. We consider a specific $\Sigma_1^{P,1}$-predicate $\tau(A; x) \equiv (\exists y, |y| = |x|)\ 1xy \in A$, and require that for all x, $x \notin K(A) \iff \tau(A; x)$. To fit this requirement into the stage construction for requirements $R_{n,0}$, we divide it into an infinite number of requirements:

[6] A set B is complete for a class $\mathcal{C}$ if $B \in \mathcal{C}$ and for all sets $C \in \mathcal{C}$ there exists a polynomial-time computable function f such that for all x, $x \in C$ iff $f(x) \in B$.

$R_{n,1}$: $(\forall x, |x| = n)[x \notin K(A) \iff \tau(A; x)]$.

Now we describe the construction of set A. In stage n, we will satisfy requirement $R_{n,0}$ by choosing $x^n = 0^m$, where m is even, $m > t(n-1)$ and $2^{m/2} > p_n(m)$. Also in stage n, we will satisfy requirements $R_{i,1}$, for all i, $t(n-1) < 2i+1 \leq t(n)$ (again, $t(n) = p_n(m)$).

More precisely, stage n consists of four steps. In Step 1, we determine the integer m, and let $t(n) = p_n(m)$. In Step 2, we determine the memberships in A or $\overline{A}$ for strings x of length $t(n-1) < |x| < m$, and satisfy requirements $R_{i,1}$, $t(n-1) < 2i+1 < m$. This is done in $m - t(n-1) - 1$ many substeps: Substeps $t(n-1)+1, \cdots, m-1$. To begin, we let $X(t(n-1)) = A(n-1)$ and $X'(t(n-1)) = A'(n-1)$. At Substep j, where j is even, do nothing: $X(j) = X(j-1)$ and $X'(j) = X'(j-1)$. At Substep j, where j is odd, determine, for each string x of length $(j-1)/2$, whether $x \in K(X(j-1))$. If $x \in K(X(j-1))$, then let $Y_x = Y'_x = \emptyset$; otherwise, let $Y_x = \{1xy|\ |y| = |x|\}$ and $Y'_x = \emptyset$. Define $X(j) = X(j-1) \cup (\cup_{|x|=(j-1)/2} Y_x)$ and $X'(j) = X'(j-1) \cup (\cup_{|x|=(j-1)/2} Y'_x)$. Note that by the end of Substep $m-1$, we have $x \notin K(X(m-1)) \iff \tau(X(m-1); x)$ for all x, $t(n-1) < 2|x|+1 < m$, because the question of whether $x \in K(A)$ depends only on $A^{<|x|}$ and hence $x \in K(X(2|x|))$ iff $x \in K(X(m-1))$.

In Step 3, we need to satisfy $R_{n,0}$, and also make sure that requirements $R_{i,1}$, $m \leq 2i+1 \leq t(n)$, can be satisfied in Step 4. We consider the computation tree T of $\sigma_n(A; 0^m)$, with each query "$y \in ?A$" answered by $\chi_{X(m-1)}(y)$ if $|y| < m$. We also consider the following circuits:

(1) the Σ_1-circuit C_0 corresponding to the predicate "$0^m \in L_A$", and

(2) for each x, $m \leq 2|x|+1 \leq t(n)$, the circuit C_x corresponding to the predicate $\tau(A; x)$.

Let $D'(n) = \{y|\ m \leq |y| \leq t(n)\}$. To satisfy requirement $R_{n,0}$, and in the meantime, allowing requirements $R_{i,1}$, $m \leq 2i+1 \leq t(n)$, to be satisfied later, we need to find sets B_0, B_1 such that $B_0 \cup B_1 \subseteq D'(n)$, $B_0 \cap B_1 = \emptyset$, and that the following properties (a), (b) and (c) hold. Let $\rho = \rho_{B_0,B_1}$ be the restriction such that $\rho(v_z) = 1$ if $z \in B_1$, $\rho(v_z) = 0$ if $z \in B_0$, and $\rho(v_z) = *$ if $z \notin B_0 \cup B_1$.

(a) The computation tree T always accepts or always rejects, if all queries $z \in B_1$ are answered *yes* and all queries $z \in B_0$ are answered *no* (and other queries remained unanswered). We abuse the notation and write that $T\lceil_\rho \neq *$.

(b) The circuit $C_0\lceil_\rho$ is completely determined , and $C_0\lceil_\rho \neq T\lceil_\rho$.

(c) The circuits $C_x\lceil_\rho$, for all x such that $m \leq 2|x|+1 \leq t(n)$, are undetermined.

The above conditions (a) and (b) together satisfy requirement $R_{n,0}$ and condition (c) leaves $C_x\lceil_\rho$ undetermined so that all requirements $R_{i,1}$, $m \leq 2i+1 \leq t(n)$, can be satisfied in Step 4. Since variables in C_0 are those v_y's with $|y|$ being even and variables in C_x's are those v_y's with $|y| = 2|x|+1$, we can further simplify the above conditions (a), (b) and (c) into the following requirement:

R'_n: $(\exists x_n)(\exists B_0, B_1 \subseteq D'(n))[B_0 \cap B_1 = \emptyset$, $T\lceil_{\rho_{B_0,B_1}} \neq *$, and $C_0\lceil_{\rho_{B_0,B_1}} = C_x\lceil_{\rho_{B_0,B_1}} = *$ for all x such that $m \leq 2|x|+1 \leq t(n)]$.

To see that requirement R'_n can be satisfied, we let π be the fixed computation path in T in which all queries are answered *no*. Let $B_0 = \{y|\ y$ is queried in path $\pi\}$ and $B_1 = \emptyset$. Then, obviously, $T\lceil_{\rho_{B_0,B_1}} \neq *$. We note that m is chosen such that $2^{m/2} > p_n(m)$, and so $\|B_0\| \leq p_n(m) < 2^{m/2}$. Therefore, ρ_{B_0,B_1} cannot determine circuit C_0, nor any C_x, $m \leq 2|x|+1 \leq t(n)$, since each of these circuits needs at least one positive value or at least $2^{m/2}$ negative values to be completely determined.

Now, from requirement R'_n, if $T\lceil_{\rho_{B_0,B_1}} = 0$, then we choose one variable v_z in C_0 such that $z \notin B_0$, let $X(m-1) = X(m-1) \cup \{z\}$, and if $T\lceil_{\rho_{B_0,B_1}} = 1$ then do nothing. This satisfies requirement $R_{n,0}$ with respect to set $X(m-1)$.

Finally, in Step 4, we satisfy requirements $R_{i,1}$ for all i such that $m \leq 2i+1 \leq t(n)$. Again, we divide this step into $t(n)-m+1$ many substeps. In each Substep j, $j = m, \cdots, t(n)$, we consider all strings x of length $(j-1)/2$ to see whether $x \in K(X(j-1))$. If $x \in K(X(j-1))$, then let $Y_x = Y'_x = \emptyset$; otherwise, find a string $z = 1xy$, $|y| = |x|$, such that $z \notin B_0$ and let $Y_x = \{z\}$ and $Y'_x = \emptyset$ (such a string z must exist as guaranteed by condition (c) of Step 3). Define $X(j) = X(j-1) \cup (\cup_{|x|=(j-1)/2} Y_x)$ and $X'(j) = X'(j-1) \cup (\cup_{|x|=(j-1)/2} Y'_x)$. Let $A(n) = X(t(n))$ and $A'(n) = X'(t(n))$. This completes stage n.

The above stage n defined set $A(n)$, or $A^{\leq t(n)}$ if we let $A = \cup_{n=1}^{\infty} A(n)$. Note that we have, in Step 3 of stage n, satisfied requirement $R_{n,0}$ with respect to set $X(m-1)$. Note that our construction of A always keep A to be an extension of $X(j)$ and $X'(j)$ within stage n in the sense that $X(j) \subseteq A$ and $X'(j) \subseteq \overline{A}$. Therefore, requirement $R_{n,0}$ is also satisfied with respect to set A. Similarly, requirements $R_{i,1}$, for all i such that $t(n-1) < 2i+1 \leq t(n)$, are satisfied in stage n with respect to $X(2i+1)$ and $X'(2i+1)$, and so they are satisfied with respect to set A.

In summary, our construction of set A which collapses class $\mathcal{C}_3$ to class $\mathcal{C}_1$ but still separates $\mathcal{C}_1$ from $\mathcal{C}_2$ uses the following general setting:

(1) The diagonalization setting for $\mathcal{C}_1$ against $\mathcal{C}_2$ includes (a) the enumeration of predicates $\sigma_n(A;x)$ in $\mathcal{C}_2$, (b) defining a fixed set $L_A \in \mathcal{C}_1(A)$ such that the question of whether $x \in L_A$ depends only on set $A \cap W(x)$, where $W(x)$ is a window usually contained in $\{0,1\}^{|x|}$, (c) setting the requirement

$$R_{n,0}\text{: } (\exists x_n)[x_n \in L_A \iff \textit{not } \sigma_n(A;x_n)],$$

and (d) defining $t(n)$ such that diagonalization occurs in the region $D'(n) = \{y| \, |x_n| \leq |y| \leq t(n)\}$.

(2) To collapse the class $\mathcal{C}_3$ to $\mathcal{C}_1$, we need a complete set $K(A)$ for $\mathcal{C}_3(A)$, which has the following property: whether a string x is in $K(A)$ depends only on set $A^{<|x|}$; or, equivalently, $A^{<|x|} = B^{<|x|}$ implies that $x \in K(A) \iff x \in K(B)$. Then we need a fixed a predicate $\tau(A;x)$ in $\mathcal{C}_1$ which has the property that whether $\tau(A;x)$ holds or not depends only on set $A \cap W'(x)$, where $W'(x)$ is a window such that all windows $W'(x)$ and $W(y)$ are pairwisely disjoint. $W'(x)$ is often defined to be a subset of $\{0,1\}^{g(|x|)}$ for some function g. (In the above example, we had $W'(x) = \{1xy| \, |y| = |x|\}$, and $g(n) = 2n+1$.) The new requirements are

$$R_{n,1}\text{: } (\forall x, |x| = n)[x \in K(A) \iff \tau(A;x)].$$

(3) Inside the diagonalization region $D(n)$, we need to satisfy requirement $R_{n,0}$ as well as $R_{i,1}$ for all i, $t(n-1) < g(i) \leq t(n)$. Requirements $R_{i,1}$, $t(n-1) < g(i) < |x_n|$, will be done first in a straightforward way. Let $X(|x_n|-1)$ be the resulting extension of set $A(n-1)$. Then, requirement $R_{n,0}$ will be strengthened to

$$R'_n\text{: } (\exists B_0, B_1 \subseteq D'(n))[B_0 \cap B_1 = \emptyset \text{ and } C\lceil_{\rho_{B_0,B_1}} \neq * \text{ and } C_0\lceil_{\rho_{B_0,B_1}} = C_x\lceil_{\rho_{B_0,B_1}} \neq *, \text{ for all } x, \, |x_n| \leq g(|x|) \leq t(n)],$$

where C_0 is the circuit corresponding to the predicate "$x_n \in L_A$", C_x is the circuit corresponding to the predicate $\tau(A;x)$, and C is the circuit corresponding to the predicate $\sigma_n(A;x_n)$, with the variables v_y replaced by the value $\chi_{X(|x_n|-1)}(y)$, if $|y| < |x_n|$, and ρ_{B_0,B_1} is the restriction defined above. Observe that the variables in C_0 are those v_y's such that $y \in W(x_n)$, and the variables in C_x are those v_y's such that $y \in W'(x)$. So, all circuits except circuit C have pairwisely disjoint variables. This verifies that R'_n implies $R_{n,0}$. Also note that now $R_{n,0}$ is, again, reduced to a lower bound requirement on circuits—this time a little more complex than the lower bound requirements defined in Sections 3 and 4.

Finally, when R'_n is satisfied, requirements $R_{i,1}$, $|x_n| \leq g(i) \leq t(n)$, can be satisfied since we have left $C_x\lceil_{\rho_{B_0,B_1}}$ undertimed for all x of length i.

6. Collapsing Results

In this section, we demonstrate how to apply the general setting of Section 5 to collapse some hierarchy to a fixed level, while keeping other classes separated. This task is generally more complex than the separation results. In particular, when the classes to be separated are not of an apparently simpler structure than the classes to be collapsed, some different types of encoding techniques must be used. These special techniques are presented in Sections 6.2 and 6.3.

6.1. Collapsing PSPACE to PH

We first construct an oracle A such that $PSPACE(A) = \Sigma^P_k(A) \neq \Sigma^P_{k-1}(A)$, where k is an arbitrary but fixed integer greater than 0.

Following the setting given in Section 5, we let

$$\begin{aligned} L_A =& \{0^{(k+1)n} |\ (\exists y_1, |y_1| = n) \cdots (Q_k y_k, |y_k| = n)\ 0^n y_1 \cdots y_k \in A\}, \\ \tau(A; x) \equiv& (\exists y_1, |y_1| = |x|) \cdots (Q_k y_k, |y_k| = |x|)\ [1x y_1 \cdots y_k \in A], \text{ and} \\ Q(A) =& \{\langle 0^i, a, 0^j \rangle | \text{ the } i\text{th TM } M_i \text{ accepts } a \text{ using } \leq j \text{ cells}\}. \end{aligned}$$

In the above definition for $Q(A)$, the space used by the query tape of machine M_i is included in the space measure. Therefore, set $Q(A)$ is complete for $PSPACE(A)$, and whether x is in $Q(A)$ depends only on set $A^{<|x|}$. In addition, let $W(0^m) = \{0^{m/(k+1)}z|\ |z| = km/(k+1)\}$ and $W'(x) = \{1xz|\ |z| = k|x|\}$. Then, the question of whether $0^m \in L_A$ depends only on set $A \cap W(0^m)$ and the predicate $\tau(A; x)$ depends only on set $A \cap W'(x)$. Also, all windows $W(0^m)$ and $W'(x)$ are pairwisely disjoint. Let $g(i) = (k+1)i + 1$.

In stage n, we choose a sufficiently large m such that m is a multiple of $k+1$ and is greater than $t(n-1)$. Let $x_n = 0^m$ and $t(n) = p_n(m)$. As in Section 5, we first perform Steps 1 and 2 and define set $X(n-1)$. Then, let C be the circuit corresponding to the nth $\Sigma^{P,1}_{k-1}$-predicate $\sigma_n(A; x_n)$, with the variables v_y replaced by the value $\chi_{X(m-1)}(y)$, if $|y| < m$, C_0 be the circuit corresponding to the predicate "$0^m \in L_A$", and C_x be the circuit corresponding to the predicate $\tau(A; x)$. From

the discussion in Section 5, our construction in stage n is reduced to the following requirement (if $k \geq 2$):

R'_n: $(\exists B_0, B_1 \subseteq D'(n))[B_0 \cap B_1 = \emptyset$ and $C\lceil_{\rho_{B_0,B_1}} \neq *$ and $C_0\lceil_{\rho_{B_0,B_1}} = C_x\lceil_{\rho_{B_0,B_1}} \neq *$, for all x, $m \leq g(|x|) \leq t(n)]$,

where ρ_{B_0,B_1} is the restriction defined by making all v_z to be 1 if $z \in B_1$ and all v_z to be 0 if $z \in B_0$. (Note that when $k = 1$, the construction is almost identical to that given in Section 5, with the *NP*-complete set $K(A)$ replaced by the *PSPACE*-complete set $Q(A)$.)

It is clear that C is a $\Sigma_{k-1}(p_n(m))$-circuit, while each of C_0 and C_x contains a subcircuit computing a function $f_k^{2^{m/(k+1)}}$. This allows us to show that requirement R'_n can be satisfied by showing the following generalization of the lower bound results of Theorem 4.6.

Theorem 6.1. Let $\{C_i\}_{i=1}^t$ be t circuits, each computing a function f_k^n, with pairwisely disjoint variables. Let C be a Σ_{k-1}-circuit having size $\leq r$ and bottom fanin $\leq \log r$. If $t \leq 2^{\delta n^{1/6}}$ and $r \leq 2^{\delta n^{1/3}}$, with $\delta = 1/12$, then for sufficiently large n, there exists a restriction ρ such that $C\lceil_\rho \neq *$ and $C_i\lceil_\rho = *$ for all $i = 1, \cdots, t$.

Proof. The proof is very similar to that of Theorem 4.6. First, for the purpose of induction, we change the induction statement to a stronger form that the theorem holds if C satisfies the weaker constraint that it has at most r gates not at the bottom level.

The base step of the induction proof, with $k = 2$, is trivial. We leave it to the reader. For the inductive step, we define probability spaces $R^+_{q,\mathcal{B}}$ and $R^-_{q,\mathcal{B}}$ as in the proof of Theorem 4.6, where $\mathcal{B}$ is the partition of variables such that all variables leading to a bottom gate in any C_i form a block. We need to show

(a) for a random restriction ρ, the probability that $C\lceil_\rho$ is equivalent to a Σ_{k-2}-circuit having $\leq r$ gates not at the bottom level and having bottom fanin $\leq \log r$ is big, and

(b) for a random restriction ρ, the probability that each $C_i\lceil_\rho$, $1 \leq i \leq t$, contains a subcircuit computing a function f_{k-1}^n is big.

Part (a) follows exactly from the proof of Theorem 4.6. Part (b) can be proved by a slight modification of the proof of Lemma 4.8. Namely, we let q_1 be the probability that bottom AND gates $H_j\lceil_{\rho g(\rho)}$ of $C_i\lceil_{\rho g(\rho)}$, $1 \leq i \leq t$, takes the value s_j, and let q_2 be the probability that all OR gates at level 2 from bottom of all

$C_i\lceil_{\rho g(\rho)}$, $1 \le i \le t$, have $\ge \sqrt{n}$ many child nodes $H_j\lceil_{\rho g(\rho)}$ having value $s_j = *$. Note that in the proof of Lemma 4.8, the estimation for q_1 was

$$q_1 \ge (1 - e^{-n^{1/6}})^{n^{k-1}},$$

because there were n^{k-1} many bottom gates in C_0. Here, we have t such circuits, each having n^{k-1} many bottom gates, so

$$q_1 \ge (1 - e^{-n^{1/6}})^{n^{k-1} \cdot t} \ge 1 - e^{-n^{1/6}} \cdot n^{k-1} \cdot 2^{\delta n^{1/6}} \ge 5/6,$$

if n is sufficiently large.

Similarly, we modify the estimation for probability q_2 and obtain

$$q_2 \ge (1 - 2^{-n^{1/2}})^{n^{k-2} \cdot t} > 5/6,$$

for sufficiently large n. □

Theorem 6.2. For every $k \ge 1$, there exists a set A such that $PSPACE(A) = \Sigma_k^P(A) \ne \Sigma_{k-1}^P(A)$.

Proof. There are $\le 2^{t(n)} = 2^{p_n(m)}$ many circuits C_x, each having a subcircuit computing a function $f_k^{2^{m/(k+1)}}$. To satisfy requirement R'_n, all we need is to choose m so large that Theorem 6.1 holds with respect to the function $f_k^{2^{m/(k+1)}}$ and that $(1/12)2^{m/6(k+1)} > k \cdot p_n(m)$. □

6.2. Collapsing PH but Keeping PSPACE Seperated from PH

In this section, we construct an oracle A such that $PSPACE(A) \ne \Sigma_{k+1}^P(A) = \Sigma_k^P(A) \ne \Sigma_{k-1}^P(A)$ for an arbitrary $k \ge 1$.

The separation part consists of two requirements: $PSPACE(A) \ne \Sigma_k^P(A)$ and $\Sigma_k^P(A) \ne \Sigma_{k-1}^P(A)$. Let

$$L_A = \{0^m | \ \|A^{=n}\| \text{ is odd}\} \in PSPACE(A),$$

and

$$L'_A = \{0^{(k+1)n} | \ (\exists y_1, |y_1| = n) \cdots (Q_k y_k, |y_k| = n) 0^n y_1 \cdots y_k \in A\} \in \Sigma_k^P(A).$$

These requirements may be divided into an infinite number of requirements:

$R_{2n,0}$: $(\exists x_{2n})[x_{2n} \in L_A \iff \textit{not } \sigma_n^{(k)}(A; x_{2n})]$,

$R_{2n+1,0}$: $(\exists x_{2n+1})[x_{2n+1} \in L'_A \iff \textit{not } \sigma_n^{(k-1)}(A; x_{2n+1})]$,

where $\sigma_n^{(h)}$ is the nth $\Sigma_h^{P,1}$-predicate.

The collapsing part requires that $\Sigma^P_{k+1}(A) = \Sigma^P_k(A)$. We assume the existence of a $\Sigma^P_{k+1}(A)$-complete set $K^{k+1}(A)$ which has the property that the question of whether $x \in K^{k+1}(A)$ depends only on set $A^{<|x|}$. Next we let $\tau(A;x)$ be a fixed $\Sigma^{P,1}_k$-predicate:

$$\tau(A;x) \equiv (\exists y_1, |y_1| = |x|) \cdots (Q_k y_k, |y_k| = |x|)\ 1xy_1 \cdots y_k \in A.$$

We divide the requirement of $\Sigma^P_{k+1}(A) = \Sigma^P_k(A)$ into the following requirements:

$R_{i,1}$: $(\forall x, |x| = i)\ [x \in K^{k+1}(A) \iff \tau(A;x)]$.

We now describe the construction of set A. In stage $2n+1$, we will satisfy requirements $R_{2n+1,0}$, as well as requirements $R_{i,1}$, for all i such that $t(n-1) < g(i) \leq t(n)$, where $t(n-1)$ and $t(n)$ are the bounds for the diagonalization region described in Sections 3 and 5, and $g(i) = (k+1)i + 1$. The construction is almost identical to the stage n of the construction in Section 6.1. The only difference is that we now are working with the complete set $K^{k+1}(A)$ instead of the $PSPACE(A)$-complete set $Q(A)$.

In stage $2n$, we will satisfy requirement $R_{2n,0}$, as well as requirements $R_{i,1}$, for all i such that $t(n-1) < g(i) \leq t(n)$. Following the setting of Section 5, we let $x_{2n} = 0^m$, with m being a multiple of $k+1$ and is sufficiently large. We now consider the following circuits:

(a) the circuit C corresponding to the predicate $\sigma^{(k)}_n(A; 0^m)$, with all variables v_y replaced by $\chi_{A(n-1)}(y)$ if $|y| < t(n-1)$,

(b) for each x, $t(n-1) < g(|x|) \leq t(n)$, the circuit C_x corresponding to the predicate $\tau(A;x)$.

We need a restriction ρ on variables such that $C\lceil_\rho \neq *$, $C_x\lceil_\rho = *$ for all x, $t(n-1) < g(|x|) \leq t(n)$, and the set $\{y|\ |y| = m, \rho(v_y) = *\}$ is nonempty. (Note that none of the variables v_y in C_x's is of length m.) Inspecting the structure of these circuits, we learn that C is a $\Sigma_k(p_n(m))$-circuit and each C_x is a $\Sigma_k(|x|)$-circuit. Thus, such a restriction ρ does not seem to exist (e.g., circuit C may simulate a particular C_x). Our solution to this problem is to modify these requirements so that the requirements $R_{i,1}$, even for $g(i) \geq m$, can be satisfied *before* the requirement $R_{2n,0}$ is satisfied. The price we pay is that the circuit C would become more complicated—though still not complicated enough to be able to compute the parity of $A^{=m}$.

We now describe how this tradeoff between the structure of circuits C and C_x's is done. First, we replace each requirement $R_{i,1}$ by a simpler requirement $R'_{i,1}$

(only for those i such that $t(n-1) < g(i) \leq t(n)$):

$R'_{i,1}$: $(\forall x, |x| = i)[x \in K^{k+1}(A) \Rightarrow (\forall z, |z| = ki) 1xz \in A]$ and $[x \notin K^{k+1}(A) \Rightarrow (\forall z, |z| = ki) 1xz \notin A]$.

Using these simpler requirements, we can modify circuit C to depend only on variables v_y, $|y| = m$, and so the requirement $R_{2n,0}$ may be satisfied without worrying about requirements $R_{i,1}$. The modification will make circuit C having depth greater than $k+1$, but within the acceptable size. Let $V = \{v_y |\ v_y$ occurs in C and $y = 1xz$ for some x and z, $|z| = k|x|$, and $|y| > t(n-1)\}$. For each x of length $> (t(n-1)-1)/(k+1)$, let D_x be the circuit corresponding to the predicate $x \in K^{k+1}(A)$, with its variables v_y replaced by $\chi_{A(n-1)}(y)$ if $|y| < t(n-1)$, and replaced by 0 if $|y| \neq m$ and $v_y \notin V$. Note that D_x is a Σ_{k+1}-circuit such that all its variables have the form $v_{x'}$, with $|x'| < |x|$, because the question of "$x \in ?K^{k+1}(A)$" depends only on set $A^{<|x|}$. In addition, we note that for every set A, if A satisfies

$$(*) \qquad A^{\leq t(n-1)} = A(n-1), A \cap \{y |\ |y| > t(n-1), |y| \neq m, v_y \notin V\} = \emptyset$$

then $D_x\lceil_{\rho_A} = 1$ iff $x \in K^{k+1}(A)$.

Now we modify circuit C as follows. First, for all variables v_y with $|y| \leq t(n-1)$, replace v_y by $\chi_{A(n-1)}(y)$. Then, for all variables $v_y \notin V$ such that $|y| \neq m$, replace v_y by constant 0. Second, for all variables $v_y \in V$, if $y = 1xz$ for some x and z, $|z| = k|x|$, then we replace it by circuit D_x (and replace $\overline{v}_y$ by the dual circuit of D_x). Note that after the modification, we obtain a circuit C_1 of depth $\leq 2(k+1)$ but each of its variables v_y is either of length $|y| = m$ or is in V and of length $|y| < p_n(m)/(k+1)$. Also note that if we apply a restriction ρ_A to circuit C_1, then $C_1\lceil_{\rho_A} = 1$ iff $\sigma_n(A; 0^m)$, provided that requirements $R'_{i,1}$ are satisfied by A for all i such that $t(n-1) < g(i) \leq t(i)$, and that A satisfies $(*)$. This is true because requirements $R'_{i,1}$ imply that for each $y = 1xz$, $|z| = k|x|$, $x \in K^{k+1}(A) \iff y \in A$ and so $D_x\lceil_{\rho_A} = 1 \iff y \in A$.

Repeat the above process until the circuit no longer has any variable in V (thus, all variables v_y have length $|y| = m$). To obtain such a circuit C', we need only to repeat the above modification for at most $\log(p_n(m))$ times. Therefore, the resulting circuit C' has depth $\leq \log(p_n(m)) \cdot (k+1)$, and has fanins $\leq 2^{q(p_n(m))}$, where q is a polynomial depending on the set $K^{k+1}(A)$ (i.e., $2^{q(|x|)}$ bounds the fanins of circuit D_x). Also, if A satisfies $(*)$ and requirements $R'_{i,1}$ for all i such that $t(n-1) < g(i) \leq t(i)$, then $C'\lceil_{\rho_A} = 1$ iff $\sigma_n(A; 0^m)$. Since C' does not have common

variables with C_x's, we need only to show that C' does not compute the parity of 2^m variables. This, again, reduces our diagonalization problem to the lower bound problem about parity circuits.

Once it is proved that this modified circuit C' does not compute the parity of set $A^{=m}$, we can complete the stage $2n$ by finding a subset $B \subseteq A^{=m}$ such that ρ_B completely determines C' but $C'\lceil_{\rho_B} = 1$ iff $\|B\|$ is even. Then, we satisfy requirements $R'_{i,1}$ for each i, $t(n-1) < g(i) \leq t(n)$, by letting each $y = 1xz$, $|z| = k|x|$, be in $A(n)$ iff $x \in K^{k+1}(A)$ for all x, $|x| = i$. (More precisely, we do this in $t(n) - t(n-1)$ substeps. We let $X(t(n-1)) = A(n-1)$ and, in each Substep j, $t(n-1) < j \leq t(n)$, we let strings $y = 1xz$ be in $X(j)$ iff $x \in K^{k+1}(X(j-1))$ for all x, $g(|x|) = j$. Finally, let $A(n) = X(t(n))$.) Note that set $A(n)$ satisfies both $(*)$ and $R'_{i,1}$ for all i, $t(n-1) < g(i) \leq t(n)$. It is left to show that circuit C' does not compute the parity of set $A^{=m}$.

Theorem 6.3. Let $s(n)$ be the minimum size of circuit C of depth $k = c \log\log n$ for some $c > 0$ such that C computes the parity of n variables. Then, for sufficiently large n, $s(n) \geq 2^{\epsilon n^{1/(k-1)}}$ for some $\epsilon > 0$.

Proof. We proved in Theorem 4.1 and Remark 4.3 that for some $\epsilon > 0$, $s_k(n) \geq 2^{\epsilon n^{1/(k-1)}}$ for depth-k parity circuits, if $n > n_k$ for some constant n_k depending only on k. In fact, the constant n_k can be taken as $(n_0)^k$ for some absolute constant n_0 which does not depend on k (see Hastad [1987] for details). So, if $n > (n_0)^{c \log\log n}$, then $s(n) \geq 2^{\epsilon n^{1/(k-1)}}$. □

Theorem 6.4. For each $k \geq 1$, there exists an oracle A such that $PSPACE(A) \neq \Sigma^P_{k+1}(A) = \Sigma^P_k(A) \neq \Sigma^P_{k-1}(A)$. Also, there exists an oracle B such that $PSPACE(B) \neq NP(B) = P(B)$.

The above proof actually showed that the set $L_A = \{0^m | \ \|A^{=m}\| \text{ is odd}\}$ is not in $PH(A)$ even if $PH(A)$ is finite. Therefore, for this set A, $\oplus P(A) \not\subseteq PH(A)$. In fact, all we need above is that L_A is not in $\Sigma^P_{c \log n}(A)$ for all $c > 0$. Thus, the same proof established the following more general result.

Corollary 6.5. If $\mathcal{C}$ is a complexity class such that $\mathcal{C}(A) \not\subseteq \Sigma^P_{c \log n}(A)$ for some oracle A, then for every $k > 0$, there exists an oracle B such that $\mathcal{C}(B) \not\subseteq \Sigma^P_{k+1}(B) = \Sigma^P_k(B) \neq \Sigma^P_{k-1}(B)$.

It is an interesting question whether there exists an oracle A such that the class $\Sigma^P_{\log n}(A)$ is outside the polynomial hierarchy $PH(A)$, and $PH(A)$ collapses to the kth level for some fixed but arbitrary k.

6.3. Collapsing BPH to PH but Keeping PH Infinite

In Section 4.3, we have shown that there exists an oracle A such that $BP\Sigma_k^P(A) \not\subseteq \Sigma_{k+1}^P(A)$ for all $k \geq 1$, and hence both hierarchies $PH(A)$ and $BPH(A)$ are infinite and the two differ at each level. What we want to show now is that relative to some oracle A the two hierarchies are identical in the sense that $BP\Sigma_k^P(A) = \Sigma_k^P(A)$ for all $k \geq 0$ and yet $PH(A)$ is infinite. Note that $\Sigma_k^P(A) \subseteq BP\Sigma_k^P(A) \subseteq \Pi_{k+1}^P(A)$ for all oracles A. Therefore our proof needs to collapse $BP\Sigma_k^P(A)$ to $\Sigma_k^P(A)$ but keeping $\Pi_{k+1}^P(A)$ separated from $\Sigma_k^P(A)$. Such a proof, like the one in Section 6.2, does not fit into the general setting of Section 5 (where the setting is designed for separating classes which lie below in the hierarchy than the classes to be collapsed).

The reader who is familiar with the theory of relativization would notice that the intended result here is a generalization of Rackoff's [1982] result that there exists an oracle A such that $P(A) = R(A) \neq NP(A)$. Rackoff's result also follows from Bennett and Gill's [1981] proof that for a random oracle A, $P(A) = R(A)$ and $P(A) \neq NP(A)$. We remark that neither of these proofs seems to work for our generalized result. Rackoff's constructive proof requires the oracle to be a sparse set, but from Balcázar, Book and Schöning [1986] and Long and Selman [1986], a sparse set does not seem to be able to separate Σ_2^P from Π_2^P (unless $\Sigma_2^P \neq \Pi_2^P$ in the unrelativized form). For the approach of using random oracles, we can see from Bennett and Gill's work that $BP\Sigma_k^P(A) = \Sigma_k^P(A)$ for random oracles A. However, it is still an important open question whether the polynomial hierarchy is infinite relative to a random oracle (cf. Cai [1986], Babai [1987] and Hastad [1987]).

First we simplify our problem to only collapsing $BP\Sigma_k^P(A)$ to $\Sigma_k^P(A)$ while keeping $\Sigma_k^P(A) \neq \Sigma_{k+1}^P(A)$, for a fixed but arbitrary $k > 0$. The separation part can be handled by a usual diagonalization setting. Let $L_A = \{0^{(k+2)n} |\ (\exists y_1, |y_1| = n) \cdots (Q_{k+1} y_{k+1}, |y_{k+1}| = n)\ 0^n y_1 \cdots y_{k+1} \in A\} \in \Sigma_{k+1}^P(A)$. Let $\sigma_n(A; x)$ be an enumeration of all $\Sigma_k^{P,1}$-predicates. Our requirements for separation are

$$R_{n,0}\colon\ (\exists x^n = 0^m)[0^m \in L_A \iff \mathit{not}\ \sigma_n(A; 0^m)].$$

For the collapsing part, first recall that $BP\Sigma_k^P(A)$ has a simple characterization (see, for example, Zachos [1986]): If $k \geq 1$ then $L \in BP\Sigma_k^P(A)$ iff there exists a $\Sigma_k^{P,1}$-predicate σ such that for all x,

$$x \in L \Rightarrow \forall_{\mathrm{p}} y\ \sigma(A; \langle x, y\rangle),$$
$$x \notin L \Rightarrow \exists_{\mathrm{p}}^{+} y\ \mathit{not}\ \sigma(A; \langle x, y\rangle).$$

We will use this characterization in the following proof.

One of the difficulty in setting the requirements for the collapsing result is that the class $BP\Sigma_k^P(A)$ is not known to possess a complete set. Therefore, the encoding of information becomes more complicated. Fortunately, we can find a pair of *pseudo-complete* sets for $BP\Sigma_k^P(A)$:

$$J_1(A) = \{\langle 0^i, a, 0^j\rangle |\ j \geq p_i(|a|), (\forall b, |b| = j)\ \sigma_i(A; \langle a, b\rangle)\},$$
$$J_0(A) = \{\langle 0^i, a, 0^j\rangle |\ j \geq p_i(|a|), (\exists\ (3/4)\cdot 2^j \text{ many } b, |b| = j)\ \textit{not}\ \sigma_i(A; \langle a, b\rangle)\},$$

where σ_i is the ith $\Sigma_k^{P,1}$-predicate. (We assume further that if $\sigma_i(A; \langle a, b\rangle)$ and b is an initial segment of b' then $\sigma_i(A; \langle a, b'\rangle)$.) From the extra condition that $j \geq p_i(|a|)$, we can see that the question of whether $x \in J_1(A)$ or $x \in J_0(A)$ depends only on set $A^{<|x|}$. The "reduction" from a set $L \in BP\Sigma_k^P(A)$ to the pair $(J_1(A), J_0(A))$ is easy to see from the characterization for $BP\Sigma_k^P(A)$: For each $L \in BP\Sigma_k^P(A)$, there exists an i such that $x \in L \Rightarrow \langle 0^i, x, 0^{p_i(|x|)}\rangle \in J_1(A)$ and $x \notin L \Rightarrow \langle 0^i, x, 0^{p_i(|x|)}\rangle \in J_0(A)$. This allows us to set up our requirements as

$$R_{n,1}\text{: } (\forall x = \langle 0^i, a, 0^j\rangle, |x| = n)[x \in J_1(A) \Rightarrow \tau(A; x) \text{ and } x \in J_0(A) \Rightarrow \textit{not}\ \tau(A; x),$$

where τ is a fixed $\Sigma_k^{P,1}$-predicate to be defined as follows.

It is natural to try to use an arbitrary $\Sigma_k^{P,1}$-predicate for $\tau(A; x)$; e.g., the one used in Section 6.1: $\tau(A; x) \equiv (\exists y_1, |y_1| = |x|)\cdots(Q_k y_k, |y_k| = |x|)\ 1xy_1\cdots y_k \in A$. Unfortunately, if we use such a predicate τ, then the conflict between the separating requirements and the encoding requirements would be too much to overcome. Instead, we let $\tau(A; x)$ be a simulation of $\sigma_i(A; \langle a, b\rangle)$ for "random" choices of b (if $x = \langle 0^i, a, 0^j\rangle$). Namely, for each n and i, $0 \leq i \leq 2^n - 1$, let $s_{n,i}$ be the ith string in $\{0,1\}^n$, under the lexicographic order, and for each n and r, $1 \leq r \leq n$, let $w_{n,r}^A = \chi_A(s_{n,(r-1)n})\cdots\chi_A(s_{n,rn-1})$, i.e., $w_{n,1}^A, \cdots, w_{n,n}^A$ are n n-bit strings determined by set $A^{=n}$. Now let $\tau_i(A; x)$ be true iff $\sigma_i(A; \langle a, w_{n,r}^A\rangle)$ are true for all r, $1 \leq r \leq n$, where $n = 2|x|$. It is clear that for all x, $x \in J_1(A) \Rightarrow \tau(A; x)$; that is, requirement $R_{n,1}$ is simplified to $(\forall x, |x| = n)\ [x \in J_0(A) \Rightarrow \textit{not}\ \tau(A; x)]$.

At stage n, we try to satisfy requirement $R_{n,0}$, as well as requirements $R_{j,1}$ for j such that $t(n-1) < 2j \leq t(n)$. The critical step is, of course, to satisfy $R_{n,0}$ and still keeps predicates $\tau(A; x)$, $t(n-1) < 2|x| \leq t(n)$, undetermined. We choose $x_n = 0^m$, where m is a sufficiently large odd integer greater than $t(n-1)$. Also, we let $t(n) = p_n(m)$. Then, we satisfy all requirements $R_{j,1}$ for all j such that

$t(n-1) < 2j < m$, by set $X(m-1)$, which is an extension of $A(n-1)$. The existence of such an extension is nontrivial, but will become clear later (see Remark 6.8).

Next, let C_0 be the Σ_{k+1}-circuit corresponding to the predicate "$0^m \in L_A$". Then, C_0 has a subcircuit computing a function $f_{k+1}^{2^{m/(k+2)}}$. Let C be the $\Sigma_k(p_n(m))$-predicate corresponding to the predicate $\sigma_n(A; 0^m)$, with all variables v_y, $|y| < m$, replaced by $\chi_{X(m-1)}(y)$. In order to perform diagonalization while leaving predicates $\tau(A; x)$ undetermined, we must modify circuit C as we did in Section 6.2. However, we cannot increase its depth here, because C_0 is barely one level deeper than C. What we will do is to simulate *all* possible answers for queries "$y \in ?A$" asked by $\sigma_n(A; 0^m)$ if $y = s_{i,j}$, for some even integer i, $m \leq i \leq p_n(m)$, and for some j, $0 \leq j \leq i^2 - 1$.

Let $h(\ell) = \sum\{i^2 |\ \ell \leq i \leq t(n), i \text{ even}\}$. We identify each string α of length $h(\ell)$ with a subset B_α of $E_\ell = \{s_{i,j} |\ \ell \leq i \leq t(n), i \text{ even}, 0 \leq j \leq i^2 - 1\}$ such that $\alpha = \chi_{B_\alpha}(s_{\ell,0}) \cdots \chi_{B_\alpha}(s_{\ell,\ell^2-1}) \chi_{B_\alpha}(s_{\ell+1,0}) \cdots \chi_{B_\alpha}(s_{t(n),t(n)^2-1})$ (assuming that both ℓ and $t(n)$ are even). Now, for each string α of length $h(m+1)$, let C_α be the circuit modified from C by (a) replacing all variables v_y, $y = s_{i,j} \in E_{m+1}$, by $\chi_{B_\alpha}(s_{i,j})$, and (b) replacing all variables v_y, $|y| \neq m$ and $y \notin E_{m+1}$, by constant 0. Then, each C_α contains only variables v_y of length $|y| = m$. We are ready to reduce our requirements $R_{n,0}$ and $R_{j,1}$'s to the lower bound problem on circuits C_0 versus C_α's. More precisely, we need the following lemma (but postpone the proof).

Lemma 6.6. Let $k \geq 1$ and h and q be two polynomial functions. Let C_α, $1 \leq \alpha \leq 2^{h(m+1)}$, be $2^{h(m+1)}$ many $\Sigma_k(q(m))$-circuits. Let C_0 be a circuit computing a function $f_{k+1}^{2^{m/(k+2)}}$. Then, for sufficiently large m, there exists an assignment ρ on variables such that $\|\{\alpha |\ C_\alpha\lceil_\rho \neq C_0\lceil_\rho\}\| \geq 2^{h(m+1)-(m+1)}$.

To see why this lemma is sufficient for our requirements, let us assume that such an assignment ρ has been chosen , and we let $X(m) = X(m-1) \cup \{y |\ |y| = m, \rho(v_y) = 1\}$. Also let $T = \{\alpha |\ C_\alpha\lceil_\rho \neq C_0\lceil_\rho\}$. Note that $\|T\| \geq 2^{h(m+1)-(m+1)}$. We claim that we can find a subset $B_{m+1} \subseteq \{s_{m+1,0}, \cdots, s_{m+1,(m+1)^2-1}\}$ such that

(a) $R_{(m+1)/2,1}$ is satisfied by $X(m) \cup B_{m+1}$, and

(b) the corresponding string $\beta_{m+1} = \chi_{B_{m+1}}(s_{m+1,0}) \cdots \chi_{B_{m+1}}(s_{m+1,(m+1)^2-1})$ has the property that the set $T_{\beta_{m+1}} =_{\text{defn}} \{\gamma \in \{0,1\}^{h(m+3)} |\ \beta_{m+1}\gamma \in T\}$ has size $\|T_{\beta_{m+1}}\| \geq 2^{h(m+3)-(m+2)}$.

To prove this claim, we first state a combinatorial lemma.

Lemma 6.7. Let $\ell < 2^{(m+1)/2}$. Let D be a $\ell \times 2^{m+1}$ boolean matrix such that each row of D has $\geq (3/4) \cdot 2^{m+1}$ many 1's. Then, for a randomly chosen $(m+1)$-tuple $(j_1, \cdots, j_{m+1})$, $1 \leq j_r \leq 2^{m+1}$,

$$\Pr[(\forall i, 1 \leq i \leq \ell)(\exists r, 1 \leq r \leq m+1) D[i, j_r] = 1] \geq 1 - 2^{-(m+3)}.$$

Now we form the matrix D as follows: each row is labeled by a string x of length $(m+1)/2$ and $x \in J_0(X(m))$, and each column is labeled by a string $z \in \{0,1\}^{m+1}$. For each x and z, let $D[x,z] = 1$ iff $[x = \langle 0^i, a, 0^j \rangle$ and $\sigma_i(X(m); \langle a, z \rangle)$ is false]. Then, D satisfies the hypothesis of Lemma 6.7 and hence the conclusion. That is, the set $S_{m+1} = \{z_1 \cdots z_{m+1} |\ |z_1| = \cdots = |z_{m+1}| = m+1, (\forall x \in J_0(X(m)) \cap \{0,1\}^{(m+1)/2})(\exists j, 1 \leq j \leq m+1) D[x, z_j] = 1\}$ has size $\|S_{m+1}\| \geq (1 - 2^{-(m+3)}) 2^{(m+1)^2}$. Thus, for any $\beta_{m+1} \in S_{m+1}$, $R_{(m+1)/2,1}$ is satisfied by set $X(m) \cup B_{m+1}$ for each corresponding set B_{m+1} (i.e., $s_{m+1,j} \in B_{m+1}$ iff the jth bit of β_{m+1} is 1).

Now, it takes a simple counting argument to see that there exists a $\beta_{m+1} \in S_{m+1}$ such that $\|T_{\beta_{m+1}}\| \geq 2^{h(m+3)-(m+2)}$. This proves the claim. □

We let $X(m+1) = X(m) \cup B_{m+1}$. The above showed that we can satisfy requirement $R_{m+1,1}$ and yet keep many "good" strings α. It can be checked that the above process can be repeated for all the requirements $R_{j,1}$, $m+1 \leq 2j \leq t(n)$, and keeping size $\|T_{\beta_{2j}}\| \geq 2^{h(2j+2)-(2j+1)}$. In particular, $\|T_{\beta_{t(n)-2}}\| \geq 2^{t(n)^2-(t(n)-1)}$, and we only need to choose a string $\beta_{t(n)} \in T_{\beta_{t(n)-2}} \cap S_{t(n)}$ (where $S_{t(n)}$ is defined similar to S_{m+1}, having size $\|S_{t(n)}\| \geq (1 - 2^{-(t(n)+2)}) 2^{t(n)^2}$). Let $\beta = \beta_{m+1}\beta_{m+3} \cdots \beta_{t(n)}$ and define $X(t(n))$ accordingly. We note that $\beta \in T$ implies that $C_0\lceil_\rho \neq C_\beta\lceil_\rho$ for some assignment ρ on variables v_y, $|y| = m$. Since C_β is the circuit C with all variables v_y, $|y| \neq m$, replaced by $\chi_{X(t(n))}(y)$, we see that requirement $R_{n,0}$ is satisfied by set $X(t(n))$. Let $A(n) = X(t(n))$. This completes the stage n.

Remark 6.8. At this moment, we observe that earlier in stage n, the construction of set $X(m-1)$ can be done just like the above construction of set $X(t(n))$. Actually, we don't need part (b) of the claim, and hence it is easier.

The only thing left to show is Lemma 6.6.

Proof of Lemma 6.6. We prove it by induction on k.

First consider the case $k = 1$. We show that the lemma holds even if C_0 is an AND of $2^{m/2(k+2)}$ variables. Let $r = m/2(k+2)$. We let the 2^r variables be $v_1, \cdots, v_{2^r}$, and define $2^r + 1$ assignments ρ_i, $0 \leq i \leq 2^r$, as follows: $\rho_0(v_j) = 1$ for all

j; and for each $i \geq 1$, $\rho_i(v_j) = 1$ if $j \neq i$ and $\rho_i(v_j) = 0$ if $j = i$. Note that $C_0\lceil_{\rho_0} = 1$ and $C_0\lceil_{\rho_i} = 0$ for all $i \geq 1$. For each i, $0 \leq i \leq 2^r$, let

$$E_i = \{\alpha |\ 1 \leq \alpha \leq 2^{h(m+1)}, C_\alpha\lceil_{\rho_i} = C_0\lceil_{\rho_i}\}.$$

Suppose, by way of contradiction, that the lemma does not hold for a specific C_0. Then each E_i has size $\|E_i\| \geq (1 - 2^{-(m+1)})2^{h(m+1)}$ and hence the intersection of all E_i's must be nonempty. Let β be a specific string in the intersection of E_i's. We have $C_\beta\lceil_{\rho_i} = C_0\lceil_{\rho_i}$ for all i, $0 \leq i \leq 2^r$. Note that $C_\beta\lceil_{\rho_0} = 1$ implies that C_β has a subcircuit D having $D\lceil_{\rho_0} = 1$. However, this subcircuit D is an AND of only $q(m)$ many inputs. Assume that m is so large that $2^r > q(m)$. There must be some v_j such that neither v_j nor $\overline{v}_j$ occurs in D, and hence

$$D\lceil_{\rho_0} = 1 \Rightarrow D\lceil_{\rho_j} = 1 \Rightarrow C_\beta\lceil_{\rho_j} = 1,$$

which is a contradiction.

For the inductive step, let $k > 1$. From the proof of Theorem 6.1, we can find a restriction ρ such that $C_0\lceil_\rho$ contains a subcircuit computing a $f_k^{2^{m/(k+2)}}$ function, and for each α, $C_\alpha\lceil_\rho$ is a $\Sigma_{k-1}(q(m))$-circuit. By the inductive hypothesis, there exists an assignment ρ' such that $\|\{\alpha |\ C_\alpha\lceil_{\rho\rho'} \neq C_0\lceil_{\rho\rho'}\}\| \geq 2^{h(m+1)-(m+1)}$. The combined restriction $\rho\rho'$ satisfies our requirement. □

Theorem 6.9. For every $k > 0$, there exists an oracle A such that $\Sigma_k^P(A) = BP\Sigma_k^P(A) \neq \Sigma_{k+1}^P(A)$.

We observe that in the above proof, the collapsing requirement $\Sigma_k^P(A) = BP\Sigma_k^P(A)$ can be satisfied when we diagonalize for separating requirements $\Sigma_h^P(A) \neq \Sigma_{h+1}^P(A)$ for different h, even if $h < k$. By a careful dovetailing of requirements $\Sigma_h^P(A) \neq \Sigma_{h+1}^P(A)$ for all $h > 0$, together with a complete encoding for $\Sigma_k^P(A) = BP\Sigma_k^P(A)$ for all $k > 0$, we obtain the following result:

Theorem 6.10. There exists an oracle A such that for every $k > 0$, $\Sigma_k^P(A) = BP\Sigma_k^P(A) \neq \Sigma_{k+1}^P(A)$.

Corollary 6.11. There exists an oracle A such that $co\text{-}NP(A) \not\subseteq AM_2(A)$.

There remains an interesting question of whether the above encoding technique still works when the polynomial hierarchy collapses to the $(k+1)$st level $\Sigma_{k+1}^P(A)$. A straightforward way of combining the encoding technique here with the diagonalization technique of Section 4.1 does not seem to work.

7. Other Hierarchies

In this section, we discuss briefly two other separation results on hierarchies which use similar proof techniques.

7.1. Generalized AM Hierarchy

In Section 1, we defined generalized polynomial hierarchy $\Sigma^P_{f(n)}$ by $f(n)$ levels of alternating $\exists_{\rm p}$- and $\forall_{\rm p}$-quantifiers. Instead of $\exists_{\rm p}$- and $\forall_{\rm p}$-quantifiers, we can also define a generalized AM-hierarchy by alternating $\exists^+_{\rm p}$- and $\exists_{\rm p}$-quantifiers. However, it is not clear that the complexity classes $AM_{f(n)}$ defined this way are equivalent to the languages defined by the $f(n)$-*round AM-game* as a generalization of the Arthur-Merlin game of Babai [1985]. The main difference is that in the Arthur-Merlin game, it is required that the total accepting probability be either $\geq 3/4$ (when Merlin wins) or $\leq 1/4$ (when Merlin loses), while in the definition by $\exists^+_{\rm p}$- and $\exists_{\rm p}$-quantifiers, it is only required that each individual quantifier $\exists^+_{\rm p}$ has a fixed probability bound. For a constant function $f(n) = k$, this difference is not substantial as the repetition of the same probabilistic computation can reduce the error probability (see, for instance, Zachos [1982]). However, the price to be paid for this reduction of error probability is the increased message length exchanged between Arthur and Merlin (i.e., the length of variables quantified by $\exists^+_{\rm p}$- and $\exists_{\rm p}$- quantifiers). When $f(n)$ is, for example, a linear function, the total length increase becomes intolerable (cf. Aiello, Goldwasser and Hastad [1986]).

Therefore, we define the generalized AM-hierarchy as follows:

$$\begin{aligned} AM_{f(n)}(A) = \{L|\ & (\exists P^1\text{-predicate } \sigma)(\forall x) \\ & [x \in L \Rightarrow (\exists^{++}_{\rm p} y_1)(\exists_{\rm p} y_2)\cdots(Q'_{f(|x|)} y_{f(|x|)})\ \sigma(A; \langle x, y_1, \cdots, y_{f(|x|)}\rangle) \text{ and} \\ & [x \notin L \Rightarrow (\exists^{++}_{\rm p} y_1)(\forall_{\rm p} y_2)\cdots(Q''_{f(|x|)} y_{f(|x|)})\ \sigma(A; \langle x, y_1, \cdots, y_{f(|x|)}\rangle)\}, \end{aligned}$$

where $\exists^{++}_{\rm p} y$ denotes "for more than $(1-2^{-n})2^{q(n)}$ many $y \in \{0,1\}^{q(n)}$", $Q'_m = \exists^{++}_{\rm p}$ if m is odd, and $= \exists_{\rm p}$ if m is even, and $Q''_m = \exists^{++}_{\rm p}$ if m is odd and $= \forall_{\rm p}$ if m is even. It is easy to see that for all functions $f(n)$ which are bounded by polynomial functions, $AM_{f(n)}(A) \subseteq \Sigma^P_{f(n)}(A)$ for all sets A. On the other hand, the exact relation between the generalized AM-hierarchy and the polynomial hierarchy and the generalized polynomial hierarchy is not known. What we do know is that there exist oracles relative to which (a) the generalized AM-hierarchy does not contain

the polynomial hierarchy (in fact, it does not even contain the class *co-NP*) and (b) each class $AM_{f(n)}$ in the generalized AM-hierarchy is not contained in $\Sigma^P_{g(n)}$ if $g(n) = o(f(n))$. In this section, we give brief outline of these proofs.

Theorem 7.1. Let f and g be two functions such that both are bounded by some polynomial function $q(n)$, and $g(n) = o(f(n))$. Then, there exists an oracle A such that $co\text{-}NP(A) \not\subseteq AM_{f(n)}(A) \not\subseteq \Sigma_{g(n)}(A)$.

Corollary 7.2. Let f_i be an infinite sequence of functions such that $f_i(n) = o(f_{i+1}(n))$ for all i and that each f_i is bounded by a polynomial function. Then, there exists an oracle A such that the classes $\Sigma^P_{f_i(n)}(A)$ form a proper infinite hierarchy between polynomial hierarchy $PH(A)$ and the class $PSPACE(A)$.

For the first part of Theorem 7.1, we need to show that there exists a set A such that $co\text{-}NP(A) \not\subseteq AM_{f(n)}(A)$. We let $L_A = \{0^n |\ (\forall y, |y| = n)\ 0^n y \in A\} \in co\text{-}NP(A)$, and show that L_A is not in any class $AM_{f(n)}(A)$.

First, following the approach of Section 2, we describe circuits corresponding to complexity classes $AM_{f(n)}(A)$. We define a new type of gates called MAJ$^+$ gates which operate as follows: a MAJ$^+$ gate outputs 1 (or 0) if more than $(1 - 2^{-n})\%$ of its inputs have value 1 (or 0, respectively), and it outputs ? otherwise. Then, define a $AM_{f(n)}(m)$-*circuit* to be a circuit of depth $f(n) + 2$ having the following structure: the top $f(n)$ levels of the circuit are alternating MAJ$^+$ and OR gates beginning with a top MAJ$^+$ gate, and the bottom two levels are OR of ANDs, and it has fanins $\leq 2^m$ and bottom fanin $\leq m$.

The proof of the following lemma is similar to that of Lemma 2.3. Call predicates (τ_1, τ_2) a pair of $AM_{f(n)}$-predicates if

$$\tau_1(A; x) \equiv (\exists^{++}_{\rm p} y_1)(\exists_{\rm p} y_2) \cdots (Q'_{f(n)} y_{f(n)})\sigma(A; \langle x, y_1, \cdots, y_{f(n)}\rangle),$$

and

$$\tau_2(A; x) \equiv (\exists^{++}_{\rm p} y_1)(\forall_{\rm p} y_2) \cdots (Q''_{f(n)} y_{f(n)})\sigma(A; \langle x, y_1, \cdots, y_{f(n)}\rangle),$$

for some P^1-predicate σ, where $n = |x|$, $Q'_m = \exists^{++}_{\rm p}$ if m is odd and $Q'_m = \exists_{\rm p}$ if m is even, and $Q'_m = \exists^{++}_{\rm p}$ if m is odd and $Q'_m = \forall_{\rm p}$ if m is even. Note that a pair of $AM_{f(n)}$-predicates (τ_1, τ_2) define a set in $AM_{f(n)}(A)$.

Lemma 7.3. Let $f(n)$ be a polynomially bounded function. For every pair of $AM_{f(n)}$-predicates (τ_1, τ_2) there is a polynomial q such that for every x, there exists a $AM_{f(n)}(q(|x|))$-circuit C, having the property that for any set A, $C\lceil_{\rho_A} = 1$ if $\tau_1(A; x)$ holds, and $C\lceil_{\rho_A} = 0$ if $\tau_2(A; x)$.

Following the diagonalization setting of Section 3, we see that the critical part of the proof is to show that for any set $B \in AM_{f(n)}(A)$, there exists a sufficiently large integer m such that $0^m \in L_A$ iff $0^m \notin B$. In other words, the following lemma on circuits suffices.

Lemma 7.4. Let f and q be two polynomial functions. Let C_0 be a depth-1 circuit which is the AND of $2^{n/2}$ many variables, and let C be a $AM_{f(n)}(q(n))$-circuit. Then, for sufficiently large n, there exists an assignment ρ such that $C\lceil_\rho \neq C_0\lceil_\rho$.

Sketch of proof. The proof is similar to the proof of Lemma 6.6. Let $v_1, \cdots, v_{2^{n/2}}$ be the variables of circuit C_0. Define assignments ρ_i, $0 \leq i \leq 2^{n/2}$, as follows: $\rho_0(v_j) = 1$ for all j, and for each $i \geq 1$, let $\rho_i(v_j) = 0$ iff $i = j$. We prove by induction on $m = f(n)$ that there must exist an i, $0 \leq i \leq 2^{n/2}$, such that $C\lceil_{\rho_i} \neq C_0\lceil_{\rho_i}$.

First, we consider the case when C has only two levels; i.e., C is the OR of $\leq 2^{q(n)}$ many ANDs, each having $\leq q(n)$ many inputs. Assume, by way of contradiction, that $C\lceil_{\rho_i} = C_0\lceil_{\rho_i}$ for all $i = 0, \cdots, 2^{n/2}$. Then, $C_0\lceil_{\rho_0} = 1$ implies $C\lceil_{\rho_0} = 1$ amd that in turn implies that there is an AND gate D of C having $D\lceil_{\rho_0} = 1$. However, D has only $\leq q(n)$ imputs and so, for sufficiently large n, there is at least one v_j such that neither v_j nor its negation $\overline{v}_j$ occurs in D. Therefore, $D\lceil_{\rho_j} = D\lceil_{\rho_0} = 1$. However, $D\lceil_{\rho_j} = 1$ implies $C\lceil_{\rho_j} = 1$ and this provides a contradiction.

Now, assume that C has $m > 2$ levels with the top gate being a MAJ^+ gate. By way of contradiction, suppose that $C\lceil_{\rho_i} = C_0\lceil_{\rho_i}$ for all i, $0 \leq i \leq 2^{n/2}$. Then, for each i, there are at most $(2^{-n})\%$ of the subcircuits D of C having $D\lceil_{\rho_i} \neq C_0\lceil_{\rho_i}$. Altogether, there are at most $(2^{-n} \cdot 2^{n/2})\%$ subcircuits computing differently from C_0 on at least one assignment ρ_i. That means that there exists at least one subcircuit D of C having $D\lceil_{\rho_i} = C_0\lceil_{\rho_i}$ for all i, $0 \leq i \leq 2^{n/2}$. Note that this subcircuit D has a top OR gate and that $D\lceil_{\rho_0} = 1$. Therefore, there exists at least one subcircuit G of D such that $G\lceil_{\rho_0} = 1$. Also, $G\lceil_{\rho_i} = 0$ for all $i \geq 1$, because $D\lceil_{\rho_i} = 0$ for all $i \geq 1$. So, we have shown that there is an $AM_{m-1}(q(n))$-circuit G such that $G\lceil_{\rho_i} = C_0\lceil_{\rho_i}$ for all i, $0 \leq i \leq 2^{n/2}$. This contradicts to the inductive hypothesis. □

For the second part of Theorem 7.1, we define $L_A = \{0^n | \ (\exists^{++}_{\mathrm{p}} y_1, |y_1| = n)\ (\exists_{\mathrm{p}} y_2, |y_2| = n) \cdots (Q'_{f(n)} y_{f(n)}, |y_{f(n)}| = n)\ [0^n y_1 \cdots y_{f(n)} \in A]\}$, where $Q'_m = \exists^{++}_{\mathrm{p}}$ if m is odd and $Q'_m = \exists_{\mathrm{p}}$ if m is even. From this definition, we do not know that $L_A \in AM_{f(n)}(A)$. What we need to do is to construct A to satisfy the additional condition that $x \notin L_A \iff \tau(A; x) \equiv (\exists^{++}_{\mathrm{p}} y_1)(\forall_{\mathrm{p}} y_2) \cdots (Q''_{f(n)} y_{f(n)})\ 0^n y_1 \cdots y_{f(n)} \notin A$, where $Q''_m = \exists^{++}_{\mathrm{p}}$ if m is odd and $Q''_m = \forall_{\mathrm{p}}$ if m is even. In addition, we need to satisfy the

requirements

$$R_i\text{: } (\exists x_i = 0^m)[0^m \in L_A \iff \textit{not } \sigma_i(A; 0^m)],$$

where σ_i is the ith $\Sigma^{P,1}_{g(m)}$-predicate. For each m, define the following circuits: (a) C is the $\Sigma_{g(m)}(p_i(m))$-circuit corresponding to the predicate $\sigma_i(A; 0^m)$, and (b) C_0 is the $AM_{f(m)}$-circuit corresponding to the pair of predicates "$0^m \in L_A$" and $\tau(A; 0^m)$. Following the general diagonalization setting of Section 3, the separation problem is reduced to the following theorem on circuits.

Theorem 7.5. Let C and C_0 be defined as above. For sufficiently large m, there exists an assignment ρ such that $C\lceil_\rho \neq C_0\lceil_\rho \neq ?$.

The proof of the theorem again uses the technique of applying random restrictions ρ to circuits C and C_0 to simplify them. Two new issues arise in this application. First, a random restriction ρ is not likely to shrink circuit C_0 by only one level (like we had in Section 4.2), because we do not allow the MAJ$^+$ gates in C_0 to output ?. This is resolved by allowing ρ to shrink C_0 by k levels, where k is a constant depending on the degree of the polynomial $p_i(m)$ that bounds the bottom fanins of circuit C. Therefore, after applying random restrictions to these circuits for $g(m)$ times, circuit C is simplified into a simple depth-2 circuit but circuit C_0 still has $f(m) - k \cdot g(m)$ levels to perform diagonalization.

The second issue is how to define a probability space R of these restriction ρ such that (a) the Switching Lemma still holds with respect to space R, and (b) with a high probability, $C_0\lceil_\rho$ has depth at least $f(m) - k$. Since the circuit C_0 contains the MAJ$^+$ gates and since it is allowed to be shrunk by k levels, the probability space R is necessarily very complicated. Due to the space limit, we will omit the definition of the space R, as well as how it can satisfy the above two conditions. The interested reader should read Aiello, Goldwasser and Hastad [1986] for details.

7.2. The Low Hierarchy in NP

Schöning [1983] defined the high and low hierarchies within *NP*. It is natural to generalize it to the following relativized hierarchies. For $k \geq 0$, define

$$H^P_k(A) = \{L \in NP(A) |\ \Sigma^P_k(L \oplus A) = \Sigma^P_{k+1}(A)\},$$
$$L^P_k(A) = \{L \in NP(A) |\ \Sigma^P_k(L \oplus A) = \Sigma^P_k(A)\},$$

where $\oplus$ is the join operator on sets: $B \oplus C = \{0x |\ x \in B\} \cup \{1y |\ y \in C\}$. Let $HH(A) = \cup_{k\geq 0} H^P_k(A)$ and $LH(A) = \cup_{k\geq 0} L^P_k(A)$. It is not hard to see that

$H_k^P(A) \subseteq H_{k+1}^P(A)$ and $L_k^P(A) \subseteq L_{k+1}^P(A)$ for all $k \geq 0$ and for all A. It is however not known whether these hierarchies collapse or intersect each other. What we do know is that for all $k \geq 0$, $H_k^P(A) \cap L_k^P(A) \neq \emptyset$ iff $\Sigma_k^P(A) = \Pi_k^P(A)$ iff $NP(A) = H_k^P(A) = L_k^P(A)$. Other known relations about the hierarchies include: $L_0^P(A) = P(A)$, $L_1^P(A) = NP(A) \cup co\text{-}NP(A)$, $H_0^P(A) = \{L|\ L$ is $\leq_T^P$-complete for $NP(A)\}$. The interested reader is referred to Schöning [1983] and Ko and Schöning [1985] for more information about these hierarchies.

In this section, we show how to construct an oracle A such that $L_k^P(A) \neq L_{k+1}^P(A)$ for all $k \geq 0$. Similar separation result holds for the high hierarchy $HH(A)$. It is also possible to collapse both hierarchies to the kth level for any fixed $k \geq 0$. All these results are proven in Ko [1988b]. We only sketch the proof for the separation of the low hierarchy $LH(A)$.

In order to separate $L_{k+1}^P(A)$ from $L_k^P(A)$ for all $k \geq 1$, we need to find, for each $k \geq 0$, a set $B_k \in NP(A)$ satisfying the following two conditions:

(a_k) $\Sigma_{k+1}^P(B_k \oplus A) = \Sigma_{k+1}^P(A)$ and

(b_k) $\Sigma_k^P(B_k \oplus A) \neq \Sigma_k^P(A)$.

Note that each B_k is in $NP(A)$ and hence $\Sigma_k^P(B_k \oplus A) \subseteq \Sigma_{k+1}^P(A)$ and $\Sigma_{k+1}^P(B_k \oplus A) \subseteq \Sigma_{k+2}^P(A)$. This suggests that for condition (b_k) we need to separate the $(k+1)$st level of the polynomial hierarchy from the kth level, and for condition (a_k) we need to *partially* collapse the $(k+2)$nd level of the polynomial hierarchy to the $(k+1)$st level. The collapsing part is only a partial collapsing because condition (b_{k+1}) implies the separation of the $(k+2)$nd level of the polynomial hierarchy from the $(k+1)$st level. It is interesting to point out that our goal is only a separation result but our proof technique is more like the one used in Section 6.2 for collapsing results.

How do we *partially* collapse the polynomial hierarchy? This involves a careful choice of diagonalization regions and the witness sets B_k. To satisfy condition (a_k), we would like to make the set B_k to behave like an empty set as much as possible, and, on the other hand, to satisfy condition (b_k), we need to make the set B_k to be similar to, for instance, the set L_A used in Section 3. More precisely, we define it as follows. First, let $e(0) = 1$, and $e(n+1) = 2^{2^{e(n)}}$ for all $n > 0$. Then, for each $k \geq 0$, let

$$B_k = \{x|\ |x| = e(\langle k, m\rangle) \text{ for some } m, \text{ and } (\exists y, |y| = |x|)\ 0xy \in A\}.$$

That is, we predetermine the diagonalization regions for all diagonalization processes

for condition (b_k) for all k (namely, for any k, the corresponding regions locate close to $e(\langle k, m\rangle)$ for some m), and make B_k to be identical to the empty set outside these regions.

This definition achieves two important subgoals for our construction. First, it separates the diagonalization regions for conditions (b_k) from that for conditions (b_h) if $k \neq h$. Second, it satisfies condition (a_k) immediately outside the diagonalization regions for (b_k). This allows us to divide and conquer the numerous seemingly contradictory requirements.

Now we state our requirements as follows. For the separation part, we need

$R_{k,n,0}$: $(\exists x_n)[x_n \in E_A^{(k)} \iff \sigma_n^{(k)}(A; x_n)$ is false],

for some set $E_A^{(k)} \in \Sigma_k^P(B_k \oplus A)$, where $\sigma_n^{(k)}$ is the nth $\Sigma_k^{P,1}$-predicate. We simply let $E_A^{(k)} = \{0^{e(\langle k,m\rangle)} |\ (\exists y_1, |y_1| = r) \cdots (Q_k y_k, |y_k| = r)\ y_1 \cdots y_k 0^t \in B_k, 0 \leq t < k, rk + t = e(\langle k, m\rangle)\}$.

For the collapsing part, we need

$R_{k,n,1}$: $(\forall x, |x| = n)[x \in K^{k+1}(B_k \oplus A) \iff \tau_k(A; x)]$,

for some $\Sigma_{k+1}^{P,1}$-predicate τ_k. (Recall that $K^n(A)$ is a standard $\Sigma_k^P(A)$-complete set.)

Note that for each x such that $2^{e(\langle k,m\rangle)} \leq |x| < e(\langle k, m+1\rangle)$ for some m, the question of whether $x \in K^{k+1}(B_k \oplus A)$ can be simulated by a Σ_{k+1}^P-machine using only A as the oracle: It simulates the computation of $x \in K^{k+1}(B_k \oplus A)$ and answer each query "$y \in ?B_k$" as follows: if $|y| \neq e(\langle k, r\rangle)$ for any $r \leq m$, then answer NO, else answer YES iff $(\exists z, |z| = |y|)0yz \in A$. So, requirements $R_{k,n,1}$ need to be satisfied only if $e(\langle k, m\rangle) \leq n < 2^{e(\langle k,m\rangle)}$. We let $\tau_k(A; x)$ be the following predicate:

$$\tau_k(A; x) \equiv (\exists u_1, |u_1| = |x|) \cdots (Q_{k+1} u_{k+1}, |u_{k+1}| = |x|)\ 10^k 1 x u_1 \cdots u_{k+1} \in A.$$

(The heading of $10^k 1$ is used to distinguish between τ_k and τ_h when $h \neq k$.)

Then, in stage $n = \langle k, m\rangle$, we find the least i such that requirement $R_{k,i,0}$ is not yet satisfied and try to satisfy it by witness $x_i = 0^{e(n)}$ and the diagonalization region $D_k(i) = \{y |\ e(n) \leq |y| < 2^{e(n)}\}$. In the meantime, we need to satisfy requirements $R_{h,j,1}$ for all $h \geq 0$ and all j, $e(n) \leq j < 2^{e(n)}$. Observe that if $h \neq k$ then, $2^{e(\langle h,t\rangle)} \leq j < e(\langle h, t+1\rangle)$ for some t if $e(n) \leq j < 2^{e(n)}$. Therefore, we need only to worry about requirements $R_{k,j,1}$ for $e(n) \leq j < 2^{e(n)}$.

It is easy to check now that all these preparations lead us to a familiar diagonalization setup of Section 6.1: diagonalizing against a $\Sigma_k^{P,1}$-predicate while keeping

some set in $\Sigma_{k+2}^P(A)$ encoded by a $\Sigma_{k+1}^{P,1}$-predicate. The separation of $LH(A)$ follows immediately.

Theorem 7.6. There exists an oracle A such that for all $k \geq 0$ $L_{k+1}^P(A) \neq L_k^P(A)$.

8. Conclusion

In this paper, we have presented a general method of separating or collapsing hierarchies by oracles. The construction of the oracles usually involves two different types of proof techniques: the recursion-theoretic one and the combinatorial one. The simple relations between the computation trees generated by oracle machines and the circuits with unbounded fanins provide nice reduction of recursion-theoretic problems to combinatorial problems. For the pure separation results, the recursion-theoretic part is usually a simple diagonalization, and the main difficulty arises from finding good lower bounds for circuit complexity. For the collapsing results, both the recursion-theoretic setup and the combinatorial techniques become more complicated. It is often necessary, like in Sections 6.2 and 6.3, to use more ad hoc tricks to obtain the required lower bound results.

Following this point of view, we may continue this research in two directions. First, we need to make a deeper investigation into the diagonalization and encoding techniques, particularly how two techniques can be combined to satisfy more seemingly contradictory requirements. Do more powerful recursion-theoretic techniques, such as the finite-injury method, possibly have interesting applications in this type of proofs? Are there better forms of encoding of information to provide more free space in diagonalization regions? Although the diagonalization and encoding techniques have been examined by many people, many ad hoc techniques still seem beyond our understanding. (One example is the question posed in Section 6.3: to construct an oracle A such that $BPH(A)$ collapses to $PH(A)$ and $PH(A)$ collapses to the kth level.)

Second, in the combinatorial side, we would like to see how far the proof technique of Yao and Hastad can be stretched. Can we find general conditions on the probability spaces which allow the Switching Lemma to hold? Are there totally different approaches (such as the one of Smolensky [1987]) that make lower bound proofs easier or give better lower bounds? Even more, what is the limit of this types

of combinatorial arguments? For instance, the question of whether the polynomial hierarchy is infinite relative to a random oracle is still open. Can we sharpen this proof technique to solve this problem? or do we really need new ideas?

9. References

9.1. Bibliographic Notes

Section 1. Hopcroft and Ullman [1979] and Garey and Johnson [1979] contain introductory materials on complexity classes *P, NP, PSPACE* and *PH*. They also include formal models for oracle machines. A more recent and more complete textbook on complexity classes is Balcázar, Diaz and Gabarró [1988], which also contains the formal definitions of probabilistic classes R and BPP. The AM hierarchy was introduced by Babai [1985] and the interactive proof systems by Goldwasser, Micali and Rackoff [1985]. Their equivalence was proved by Goldwasser and Sipser [1986]. The BP operator, the probabilistic polynomial hierarchy and its relation to the class AM were given by Schöning [1987]. The nice layout of Figure 1 is from Tang and Watanabe [1988]. The notation $\exists^+_{\mathrm{p}}$ is due to Zachos [1986], which contains a survey on the relations between complexity classes definable by the $\exists^+_{\mathrm{p}}$-quantifiers over polynomial-time predicate. The relativization of these complexity classes is most often done by adding oracles to corresponding machine models; e.g., class $\Sigma^P_k(A)$ is defined by alternating machines with oracles which can make at most k alternations [Chandra, Kozen and Stockmeyer, 1981]. Our approach essentially cleans up the computation of those oracle machines and pushes the queries down to the bottom level (cf. Furst, Saxe and Sipser [1984]).

Section 2. The idea of using circuits to represent oracle computation trees (Lemma 2.3) is originated from Furst, Saxe and Sipser [1984]. They also introduced the concept of random restrictions and gave the first super-polynomial (but still sub-exponential) lower bound for constant-depth parity circuits. Sipser [1983] pointed out that the relation given by Lemma 2.3 may be applied to the separation of $PH(A)$. Majority gate MAJ, as well as similar threshold gates, have been considered in circuit complexity theory. See, for example, Hajnai et al [1987].

Section 3. The first application of the diagonalization technique to relativization was by Baker, Gill and Solovay [1975]. Many people discussed this application,

including Angluin [1980], Bennett and Gill [1981], Kozen [1978], and Torenvliet and van Amde Boas [1986].

Section 4. Before the breakthrough of Yao [1985], $PH(A)$ has been known to extend to at least $\Sigma_3^P(A)$. These results are due to Baker, Gill and Solovay [1975], Baker and Selman [1979] and Heller [1984]. Angluin [1980] also showed that $P(\#P(A)) \not\subseteq \Sigma_2^P(A)$ (a special case of Corollary 4.5). The first exponential lower bound for constant-depth parity circuit was proved by Yao [1985]. He also claimed, without a proof, a similar exponential lower bound on depth-k circuit for function f_{k+1}^n. Hastad [1986, 1987] gave a simpler proof for parity circuit, and achieved the almost optimal bound of Theorem 4.1. He also proved Yao's claim using the same proof technique (Theorem 4.6). Smolensky [1987] used an algebraic method to give a much shorter proof for the exponential lower bound for parity circuit, but his method does not seem to work for f_{k+1}^n functions. More recently, Du [1988] found, based on Hastad's proofs, simpler proofs of the Switching Lemmas (Lemmas 4.2 and 4.7), which also yield a slightly better lower bound for parity circuits. Our proofs in Sections 4.1 and 4.2 are based on Hastad's proofs. The class $\oplus P(A)$ is first defined in Papadimitriou and Zachos [1983]. Recently, Toran [1988] has proved that $NP(A) \not\subseteq \oplus P(A)$ relative to some oracle A. Lemma 4.10 was first proved by Baker and Selman [1979] in a different form. Theorem 4.11 was proved in Ko [1988a]. Corollary 4.12 has been observed independently by Watanabe [1987].

Section 5. Our example is one of the first application of the encoding technique to relativization appeared in Baker, Gill and Solovay [1975].

Section 6. The main results in Sections 6.1 and 6.2 are from Ko [1989] and the ones in Section 6.3 are from Ko [1988a]. Weaker collapsing results before include Baker, Gill and Solovay [1975] (collapsing *PSPACE* to P, and collapsing Σ_2^P to NP), Rackoff [1982] and Bennett and Gill [1981] (collapsing R to P but keep $P \neq NP$). The combinatorial lemma Lemma 6.7 has been known to many researchers, including Adleman [1978], Bennett and Gill [1981], Ko [1982] and Zachos [1982].

Section 7. The AM hierarchy was introduced by Babai [1985], who showed that the bounded AM hierarchy collapses to AM_2 and conjectured that the generalized one also collapses to AM_2. The use of notation $\exists_{\mathrm{p}}^{++}$ and the MAJ^+ gate is new. First part of Theorem 7.1 is due to Fortnow and Sipser [1988] and the second part due to Aiello, Goldwasser and Hastad [1986]. The high and low hierarchies in NP was introduced by Schöning [1983], who proved some basic properties of these hier-

archies. Ko and Schöning [1985] contains more classification of low sets by structural properties. The proofs in Section 7.2 are from Ko [1988b].

Section 8. Cai [1986] and Babai [1987] proved that *PSPACE* is not in *PH* relative to a random oracle. Other separation results by random oracles are in Bennett and Gill [1981]. Babai [1985] and Hastad [1987] have pointed out the difficulty of using Hastad's technique to prove that *PH* is infinite relative to a random oracle.

9.2. Bibliography

Adleman, L. [1978], Two theorems on random polynomial time, *Proc. 19th IEEE Symp. on Foundations of Computer Science*, 75–83.

Aiello, W., Goldwasser, S. and Hastad, J. [1986], On the power of interaction, *Proc. 27th IEEE Symp. on Foundations of Computer Science*, 368–379.

Angluin, D. [1980], On counting problems and the polynomial-time hierarchy, *Theoret. Comput. Sci.* **12**, 161–173.

Babai, L. [1985], Trading group theory for randomness, *Proc. 17th ACM Symp. on Theory of Computing*, 421–429.

Babai, L. [1987], Random oracle separates PSPACE from the polynomial-time hierarchy, *Inform. Process. Lett.* **26**, 51–53.

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

Baker, T. and Selman, A. [1979], A second step toward the polynomial hierarchy, *Theoret. Comput. Sci.* **8**, 177–187.

Balcázar, J. [1985], Simplicity, relativizations, and nondeterminism, *SIAM J. Comput.* **14**, 148–157.

Balcázar, J., Book, R. and Schöning, U. [1986], The polynomial-time hierarchy and sparse oracles, *J. Assoc. Comput. Mach.* **33**, 603–617.

Balcázar, J., Diaz, J. and Gabarró, J. [1988], *Structural Complexity I*, Springer-Verlag, Berlin.

Balcázar, J. and Russo, D. [1988], Immunity and simplicity in relativizations of probabilistic complexity classes, *RAIRO Theoret. Inform. and Appl.* **22**, 227–244.

Bennett, C. and Gill, J. [1981], Relative to a random oracle, $P^A \neq NP^A \neq coNP^A$ with probability 1, *SIAM J. Comput.* **10**, 96–113.

Boppana, R., Hastad, J. and Zachos, S. [1987], Does co-NP have short interactive proofs?, *Inform. Process. Lett.* **25**, 127–132.

Buss, J. [1986], Relativized alternation, *Proc. Structure in Complexity Theory Conf.*, Lecture Notes in Computer Science, **223**, Springer, 66–76.

Cai, J. [1986], With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy, *Proc. 18th ACM Symp. on Theory of Computing*, 21–29.

Chandra, A., Kozen, D. and Stockmeyer, L. [1981], Alternation, *J. Assoc. Comput. Mach.* **28**, 114–133.

Du, D. [1988], personal communication.

Fortnow, L. and Sipser, M. [1988], Are there interactive protocols for *co-NP* languages?, *Inform. Process. Lett.* **28**, 249–252.

Furst, M., Saxe, J. and Sipser, M. [1984], Parity, circuits, and the polynomial time hierarchy, *Math. Systems Theory* **17**, 13–27.

Garey, M. and Johnson, D. [1979], *Computers and Intractability, a Guide to the Theory of NP-Completeness*, Freeman, San Francisco.

Goldwasser, S., Micali, S. and Rackoff, C. [1985], The knowledge complexity of interactive proof systems, *Proc. 17th ACM Symp. on Theory of Computing*, 291–304.

Goldwasser, S. and Sipser, M. [1986], Private coins versus public coins in interactive proof systems, *Proc. 18th ACM Symp. on Theory of Computing*, 59–68.

Hajnai, A., Maass, W., Pudlak, P., Szegedy, M. and Turan, G. [1987], Threshold circuits of bounded depth, *Proc. 28th IEEE Symp. on Foundations of Computer Science*, 99–110.

Hastad, J. [1986], Almost optimal lower bounds for small depth circuits, *Proc. of 18th ACM Symp. on Theory of Computing*, 6–20.

Hastad, J. [1987], *Computational Limitations for Small-Depth Circuits*, (Ph.D. Dissertation, MIT), MIT Press, Cambridge.

Heller, H. [1984], Relativized polynomial hierarchies extending two levels, *Math. Systems Theory* **17**, 71–84.

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

Ko, K. [1982], Some observations on probabilistic algorithms and NP-hard problems, *Inform. Process. Lett.* **14**, 39–43.

Ko, K. [1988a], Separating and collapsing results on the relativized probabilistic polynomial time hierarchy, preprint.

Ko, K. [1988b], Separating the low and high hierarchies by oracles, preprint.

Ko, K. [1989], Relativized polynomial time hierarchies having exactly k levels, *SIAM J. Comput.*, in press; also in *Proc. 20th ACM Symposium on Theory of Computing* [1988], 245–253.

Ko, K. and Schöning, U. [1985], On circuit-size complexity and the low hierarchy in *NP*, *SIAM J. Comput.* **14**, 41–51.

Kozen, D. [1978], Indexing of subrecursive classes, *Proc. 10th ACM symp. on Theory of Computing*, 89–97.

Long, T. and Selman, A. [1986], Relativizing complexity classes with sparse oracles, *J. Assoc. Comput. Mach.* **33**, 618-627.

Papadimitriou, C. and Zachos, S. [1983], Two remarks on the power of counting, *Proc. 6th GI Conf. on Theoretical Computer Science*, Lecture Notes in Computer Science **145**, 269–276.

Rackoff, C. [1982], Relativized questions involving probabilistic algorithms, *J. Assoc. Comput. Mach.* **29**, 261–268.

Schöning, U. [1983], A low and a high hierarchy within *NP*, *J. Comput. System Sci.* **27**, 14–28.

Schöning, U. [1987], Probabilistic complexity classes and lowness, *Proc. 2nd IEEE Structure in Complexity Theory Conf.*, 2–8.

Sipser, M. [1983], Borel sets and circuit complexity, *Proc. of 15th ACM Symp. on Theory of Computing*, 61–69.

Smolensky, R. [1987], Algebraic methods in the theory of lower bounds for boolean circuit complexity, *Proc. 19th ACM Symp. on Theory of Computing*, 77–82.

Torán, J. [1988], *Structural Properties of the Counting Hierarchies*, Doctoral Dissertation, Facultat d'Informatica, UPC Barcelona.

Tang, S. and Watanabe, O. [1988], On tally relativizations of BP-complexity classes, *Proc. 3rd IEEE Structure in Complexity Theory Conf.*, 10–18.

Torenvliet, L. and van Emde Boas, P. [1986], Diagonalization methods in a polynomial setting, *Proc. Structure in Complexity Theory Conf.*, Lecture Notes in Computer Science **223**, 330–346.

Watanabe, O. [1987], Personal communication.

Wilson, C. [1988], A measure of relativized space which is faithful with respect to depth, *J. Comput. System Sci.* **36**, 303–312.

Yao, A. [1985], Separating the polynomial-time hierarchy by oracles, *Proc. of 26th IEEE Symp. on Foundations of Computer Science*, 1–10.

Zachos, S. [1982], Robustness of probabilistic computational complexity classes under definitional perturbations, *Inform. Contr.* **54**, 143–154.

Zachos, S. [1986], Probabilistic quantifiers, adversaries, and complexity classes, *Proc. of Structure in Complexity Theory Conf.*, 383–400.