Computational Complexity of Boolean Formulas with Query Symbols

Toshio SUZUKI

1999

1

Computational Complexity of Boolean Formulas with Query Symbols

Toshio SUZUKI

A dissertation submitted to the Doctoral Program
in Mathematics, the University of Tsukuba
in partial fulfillment of the requirements for the
degree of Doctor of Philosophy (Science)

January, 1999

Author: Toshio Suzuki

Department of Mathematics and Information Sciences

Osaka Prefecture University, Sakai, Osaka 599-8531, Japan.

suzuki@mi.cias.osakafu-u.ac.jp

Title: Computational Complexity of Boolean Formulas with Query Symbols

Year: 1999.

Mathematics Subject Classification:

03D15 (Mathematical logic and foundations / Recursion theory / Complexity of computation),

03E40 (Mathematical logic and foundations / Set theory / Other aspects of forcing and Boolean-valued models),

68Q15 (Computer science / Theory of computing / Complexity classes).

Key words: Structural complexity, Generic oracle, Random oracle.

Chapters 1, 3, 4 and 5 are based on the author's articles submitted to
Notre Dame Journal of Formal Logic.
(publisher: Notre Dame University, P.O. Box 5, IN 46556-0005, U.S.A.)
Chapter 7 is based on the author's article submitted to
Kobe Journal of Mathematics.
(publisher: Kobe University, Kobe 657-8501, Japan.)

A dissertation submitted to the Doctoral Program in Mathematics, the University of Tsukuba in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Science).
(Institute of Mathematics, University of Tsukuba, Ibaraki 305-8571, Japan.)

Contents

Preface

An oracle Turing machine is an algorithm that can utilize external information. Intuitively speaking, an oracle Turing machine is obtained by allowing a programmer to use special flow control statements of the following form.

```
if $u$ belongs to the oracle /* ⇐ this line is called a "query." */
    then ... ; else ... ;
end-if        /* $u$ is a bit string. */
```

A set of bit strings, called an oracle, is fixed previously to the computation of a given oracle Turing machine.

In this thesis, by extending the work of Dowd and that of Poizat, we study computational complexity of Boolean formulas with query symbols.

Main methodological features of this thesis are the following two points. First, we consider not only complexity of Turing machines but also complexity of arithmetical *predicates*. For this purpose, we introduce the concept of *forcing complexity* as an extension of Dowd's concept of t-generic oracles. The forcing complexity of an arithmetical predicate for a given oracle means the minimal size of a finite portion of the oracle that forces the predicate. An oracle with small forcing complexity of a given predicate is called a *ceiling-generic oracle for the predicate.* We present applications of forcing complexity and ceiling-generic oracles to the study of complexity of Turing machines. The results (a) and (b) in the following are shown by using forcing complexity and ceiling-generic oracles. Second, we use forcing methods not only for controlling a nondeterministic Turing machine but also for controlling the execution time of a while-loop of a *deterministic* Turing machine, by which we clarify the relationship between the r-query tautologies and fundamental problems in the theory of computational complexity. The result (c) in the following is shown by using this method.

(a) Clear-cut new proofs of previous results. We present an explicit example of a $coNP[X]$-predicate $\varphi(X)(y)$ which has exponential forcing complexity for any oracle. By this example, we give a simpler alternative proof of Dowd's result about t-generic oracles. We also present a clear-cut proof of Dowd's result about r-generic oracles, by investigating oracles' hierarchy with respect to forcing complexity.

(b) Results on Cohen-Feferman generic oracles. By investigating how existence of ceiling-generic oracles affects behavior of a Cohen-Feferman generic oracle, we show, for each positive integer r, that the following set is comeager in the Cantor space.

$$\{X : coNP[X] \not\subseteq NP[rTAUT[X]]\}.$$

(c) Control of while-loops by a forcing method. Bennet and Gill showed: "If A is a random oracle then $TAUT[A] \notin P[A]$ with probability 1." We consider the problem whether the statement of the above fact remains true when we substitute $rTAUT[A]$ for $TAUT[A]$. For each positive integer r and for each r-generic oracle A (in Dowd's sense), we show the following formula:

$$rTAUT[A] \equiv_T^P TAUT \oplus A.$$

The above formula is shown by constructing a deterministic algorithm whose while-loop's execution time is controlled by a forcing method. Consequently, we have that the following two assertions are equivalent.

(1) If A is a random oracle then $rTAUT[A] \notin P[A]$ with probability 1.

(2) The unrelativized classes R and NP are not identical.

Acknowledgement

The author would like to thank Prof. Hisao Tanaka at Hosei University for useful discussions. The author would also like to thank Prof. Nobuyoshi Motohashi at University of Tsukuba, Prof. Katsuya Eda at Waseda University and Prof. Shizuo Kamo at Osaka Prefecture University for their encouragement.

This research was partially supported by Grant-in-Aid for Scientific Research (No. 09440072, 09440078), Ministry of Education, Science, Sports and Culture of Japan.

Chapter 1

Introduction

1.1 Forcing complexity and historical background

Various researchers have suggested concepts that measures computational complexity of mathematical objects; for example, polynomial time Turing reducibility, Kolmogorov complexity, circuit complexity and Karchmer-Wigderson's communication complexity, and so on. However, we still have a variety of mathematical objects whose nature of computational complexity is unknown.

Arithmetical forcing was introduced by Feferman [14] soon after Cohen's independence proofs in set theory [10, 11]. Since Hinman's work [16], arithmetical forcing has been studied in recursion theory. For example, see Jockusch [18] or Odifreddi [24]. Later, arithmetical forcing and its variations were used as tools to study $P = ?NP$ question by some people. Typical examples are Dowd [12, 13], Ambos-Spies et al. [2], Poizat [26] and Blum and Impagliazzo [7]. Among them, Dowd investigated the relationship between uniform machines and $NP = ?coNP$ question. For this purpose, he studied the relativized propositional calculus by investigating the minimal size of the forcing condition that forces a given formula. More precisely, he investigated an extension of the propositional calculus that has query symbols for a given oracle. Then, he studied the concept of an oracle such that if a given formula of the relativized propositional calculus is a tautology with respect to the oracle then (under certain assumption) there exists a small finite portion of the oracle that forces the formula, where "small" means polynomial size and "forces" means that the formula is a tautology with respect to any oracle extending the finite portion. We may call such an approach initiated by him *"forcing complexity."*

More formally speaking, we have the following definition, where we identify an oracle with its characteristic function; thus, a finite portion of a given oracle means a function obtained from the characteristic function by restricting the domain to a finite set of bit strings. A finite portion of an oracle is often called a forcing condition, or a condition. In the following, for a set X, $\mathrm{Card}(X)$ denotes its cardinality.

Definition 1.1 *Let X be a unary predicate symbol denoting membership to a given oracle and y be a variable for a bit string. Assume that $\varphi(X)(y)$ is an arithmetical predicate (or, a functional).*

1. *Suppose that S is a forcing condition and that A is either a forcing condition or an oracle. We say "A is an extension of S" and write "$S \sqsubseteq A$" if we have $\mathrm{dom}(S) \subseteq \mathrm{dom}(A)$ and $S(u) = A(u)$ for all $u \in \mathrm{dom}(S)$.*

2. ([26]. See also Tanaka and Kudoh [31]) *$\varphi(X)(y)$ is* finitely testable *(or,* test fini*) if there exists a function $f : \mathbb{N} \to \mathbb{N}$ such that for every*

oracle A, every natural number n and every bit string u of length n, $\varphi(A)(u)$ holds if and only if $\varphi(A|f(n))(u)$ holds, where we temporarily define the oracle $A|f(n)$ as follows: we have $(A|f(n))(u) = A(u)$ for each bit string u of length at most $f(n)$, and $(A|f(n))(u) = 0$ for all u such that $|u| > f(n)$.

3. *We say "a condition S forces $\varphi(X)(u)$," where u is a given bit string, if $\varphi(A)(u)$ holds for any oracle A such that $S \sqsubseteq A$.*

4. (Suzuki [30]) *Assume that $\varphi(X)(y)$ is finitely testable. Assume that A is an oracle.* The forcing complexity of $\varphi(X)(y)$ relative to A *is a function $f : \mathbb{N} \to \mathbb{N}$ such that for each natural number n, $f(n)$ is the least number $k \in \mathbb{N}$ of the following property: for any bit string u of length n, if $\varphi(A)(u)$ is true then A has a finite portion S of size at most k such that S forces $\varphi(X)(u)$: in other words, the cardinality of $\mathrm{dom}(S)$ is at most k and for any oracle B extending S, $\varphi(B)(u)$ is true. If f is the forcing complexity of $\varphi(X)(y)$ relative to A and n is a natural number, the value $f(n)$ is called* the forcing complexity of $\varphi(X)(y)$ relative to A at n, *which is denoted by the following:*

$$\mathrm{FC}(\varphi(X)(y), A, n). \quad \square$$

The author believes that the study of forcing complexity by Dowd is important; it is an interesting example of a study that bridges over computational complexity and set theory. However, the author also feels that Dowd's proofs [13, §3, §4] are not necessarily easy to understand. Among them, the difficulty of Dowd's proof of nonexistence of t-generic oracles is pointed out in the introduction of Suzuki [28]. We discuss this matter in the next section.

1.2 Motive for the study: the existential problem of t-generic oracles

Although we shall present precise definitions in the next chapter, let us review some of them in an informal manner. The relativized propositional calculus is an extension of the propositional calculus. We get the former by adding a countable set $\{\xi^n(q_1,\ldots,q_n) : n \geq 1\}$ of connectives to the latter. Roughly speaking, $\xi^n(q_1,\ldots,q_n)$ asserts that a certain binary sequence, of length less than n, associated to the given bit string $q_1\cdots q_n$ belongs to the oracle that we are considering. Suppose that r is a positive integer. A relativized formula is called an r-query formula if it has just r-many occurrences of additional connectives. For each oracle A, $TAUT[A]$ denotes the collection of all (binary representations of) relativized formulas that are tautologies with respect to A. For each positive integer r, $rTAUT[A]$ denotes the collection of all (binary representations of) r-query formulas that are tautologies with respect to A. An oracle G is called *t-generic* [13] if every relativized tautology with respect to G is forced by a polynomial sized portion of G.

Dowd's Lemma 7 ([13, Lemma 7]) *t-generic oracles do not exist.*

Dowd proved the above lemma by using the following lemma. His expression M^X is, in our notation, $M[X]$. Similarly, $\mathcal{N}$ is $\{0,1\}^*$: we denote the collection of all bit strings of finite length by $\{0,1\}^*$, as in the textbook on computational complexity by Balcázar et al. [5]. (On the other hand, Kunen's textbook on set theory [20] denotes this collection by ${}^{<\omega}2$.) For each natural number n, $\{0,1\}^n$ $(= {}^n2)$ and $\{0,1\}^{\leq n}$ $(= {}^{\leq n}2)$ are similarly defined. It is easily verified that the cardinality of $\{0,1\}^{\leq n}$ is $2^{n+1}-1$ for each natural number n. Recall that a language A is called *sparse* if there exists a polynomial p such that for each natural number n, $\mathrm{Card}(A \cap \{0,1\}^{\leq n}) \leq p(n)$.

Citation 1.1 ([13, Lemma 6])

> **Lemma** *If a deterministic polynomial time oracle machine M^X accepts all its inputs with respect to a t-generic oracle G, then it is forced to do so by a sparse set of queries. That is, there is a partial function Y from $\mathcal{N}$ to $\{0,1\}$ satisfying $Y \sqsubseteq G$ whose domain is sparse, which forces $\forall x M^X(x)$.*
>
> Proof. The relativized formula asserting that "on all inputs of length $\leq n$ the machine M accepts" is a tautology with respect to the oracle G for every n, and its length is bounded by a polynomial in n. Therefore the nth is forced by a set W_n of queries to G of size polynomial in n. Let $W = \bigcup\{W_n : n \text{ is a power of } 2\}$. Then W is sparse, and forces the statement. □

Careful readers may hesitate, because the following assertion is false, by a counter example below.

Assertion 1.1(false) Suppose that p is a polynomial, and that for each positive integer n, D_n is a subset of $\{0,1\}^{\leq p(n)}$ such that $\mathrm{Card}(D_n) \leq p(n)$. Let $D = \bigcup\{D_n : n \text{ is a power of } 2\}$. Then D is sparse. □

Example 1.1 [28] For each natural number $n \geq 2$, let $k(n)$ be the largest natural number k such that $2^{k+1} - 1 \leq n$. For each n, let $D_n = \{0,1\}^{\leq k(n)}$.

Let $D = \bigcup\{D_n : n \text{ is a power of } 2\}$. Then, for each $n \geq 2$, D_n is a subset of $\{0,1\}^{\leq n}$ and $\mathrm{Card}(D_n)$ is at most n. However, we have $D = \{0,1\}^*$. □

Nevertheless, **Dowd's Lemma 7** is right. An alternative proof of this lemma is given in Chapter 3 of this thesis.

Remark. One significance of **Dowd's Lemma 7** is that it implies the following result about uniform machines.

Dowd's Theorem 8 ([13, Theorem 8]) *If $M[X]$ is a nondeterministic oracle machine uniformly accepting $TAUT[X]$ (i.e. if for every oracle A, we have $\mathrm{Lang}(M[A]) = TAUT[A]$), then for every oracle A, $M[A]$ does not halt in polynomial time.*

1.3 Aim of this thesis

In this thesis, by extending techniques in [13] and those in [26], we study computational complexity of $TAUT[A]$ and $rTAUT[A]$. In particular, we investigate the case where A is a Cohen-Feferman generic oracle and the case where A is a random oracle.

- **Clear-cut new proofs of previous results.** We begin our study by giving explicit alternative proofs of Dowd's results.

In [28], forcing complexity of an arbitrary finitely testable arithmetical predicate was investigated, and the concept of *ceiling-generic oracles* was introduced. An oracle is called ceiling-generic with respect to a given arithmetical predicate if the forcing complexity of the arithmetical predicate relative to the oracle is at most polynomial. In [28, §3], by using the concept of ceiling-generic oracles, an alternative proof of a weak version of **Dowd's Lemma 7** was given. In [30], an explicit example of a $coNP[X]$-predicate $\varphi(X)(y)$ with the following property was given: for each oracle A

and for each positive integer n, a lower bound for its forcing complexity is (uniformly) given as follows.

$$\mathrm{FC}(\varphi(X)(y), A, n) \geq \frac{2^{n-1} - n + 1}{n}.$$

By this example and the argument in [28, §3], an alternative proof of **Dowd's Lemma 7** is immediately derived. We present this example in Chapter 3.

By the way, Dowd also introduced weak versions of the notion of t-generic oracles. Suppose that r is a positive integer. An oracle G is called an *r-generic oracle (in Dowd's sense)*, if it satisfies the definition of a t-generic oracle with r-query tautology in place of tautology.

Dowd's Theorem 10 ([13, Theorem 10]) *Suppose that r is a positive integer. Then, r-generic oracles in the sense of Dowd form a measure one set in the Cantor space.*

The following was also shown in [13].

Fact 1.1 (section 4 of [13]). *The class of all r-generic oracles (in Dowd's sense) is meager in the Cantor space. Further, this class is closed under finite changes i.e. if A is r-generic and $B(u) = A(u)$ for all but finitely many bit strings u then B is also r-generic.* □

In Chapter 4, we explain what is difficult in Dowd's proof of his **Theorem 10**. By investigating oracles' hierarchy with respect to forcing complexity, we present a modified, clear-cut proof of **Dowd's Theorem 10**. We hope our exposition help the reader understand the pioneering work of Dowd. The contents of Chapter 4 is based on a section of [30].

- **Results on Cohen-Feferman generic oracles.** Recall the well-known result (Mehlhorn [23], [26] and [13]) that the following class of oracles is comeager (i.e. the class contains almost all oracles in topological sense):

$$\{X : P[X] \neq NP[X]\}.$$

The above result was strengthened in [28] ; we discuss this matter in Chapter 5. That is, we investigate how existence of ceiling-generic oracles affects behavior of a Cohen-Feferman generic oracle, by which we show that the following is comeager, where r is an arbitrary positive integer:

$$\{X : coNP[X] \not\subseteq NP[rTAUT[X]]\}.$$

Chapter 6 and Chapter 7 are devoted to studying random oracles.

- **One-query versus one-question.** In Chapter 6, we investigate, among others, whether for a random oracle A, the set of one-query tautologies relative to A is one-question truth-table reducible to A. We present negative answer, i.e. it is not one-question truth-table reducible to A with probability one.

- **Control of while-loops by a forcing method.** In Chapter 7, we investigate the relationship between the r-query tautologies and fundamental questions in the theory of computational complexity. In [6], Bennet and Gill showed that if A is a random oracle then $P[A] \neq NP[A]$ with probability 1. Since $TAUT[A]$ is a $coNP[A]$-complete set for an arbitrary oracle A, we obtain the following as its direct corollary.

If A is a random oracle then $TAUT[A] \notin P[A]$ with probability 1.

The above statement means that the class of all A such that $TAUT[A] \notin P[A]$ has Lebesgue measure one in the Cantor space. We consider the problem whether the statement of the above fact remains true when we

substitute $rTAUT[A]$ for $TAUT[A]$. In Suzuki [29], by extending Dowd's work about r-generic oracles, it was shown that for each positive integer r and for each r-generic oracle A (in Dowd's sense), we have the following:

$$rTAUT[A] \equiv_T^P TAUT \oplus A.$$

The above formula was shown by constructing a deterministic algorithm whose while-loop's execution time is controlled by a forcing method. By using the above formula, it was also shown in [29] that the following two assertions are equivalent.

(1) If A is a random oracle then $rTAUT[A] \notin P[A]$ with probability 1.

(2) The unrelativized classes R and NP are not identical.

Recall that $P \subseteq R \subseteq NP$. In Chapter 7, we present proofs of the above results.

Chapter 2

Preliminaries

In this chapter, we review basic concepts relating to the following subjects.

- Computational complexity.
- Random oracles and Cohen-Feferman generic oracles.
- The relativized propositional calculus.

2.1 Guide to this chapter

An oracle Turing machine is an algorithm that can utilize external information. Intuitively speaking, an oracle Turing machine is obtained by allowing a programmer to use special flow control statements of the following form.

```
/* $u$ is a bit string. */
if $u$ belongs to the oracle /* ⇐ this line is called a "query." */
    then ... ; else ... ;
end-if
```

A set of bit strings, called an oracle, is fixed previously to the computation of a given oracle Turing machine. We assume an oracle Turing machine can get the correct answer for each query. More formally, an oracle Turing machine is defined as a multitape Turing machine M with the following three properties [5]. First, M has a distinguished work tape which is called a query tape. Second, M has three distinguished states: $QUERY$, YES and NO. Third, for a given oracle A and a given input, M works as follows: at some steps of computation, M may transfer into the state $QUERY$; let u be the bit string currently appearing on the query tape; if u belongs to the oracle A then M transfers into the state YES; otherwise, M transfers into the state NO.

Each recursive function is computed by a Turing machine; likewise, for a given oracle A, each function recursive in A is computed by an oracle Turing machine with the oracle A.

By substituting oracle Turing machines for usual Turing machines in the definition of the computational complexity class P, we get the relativized computational complexity class $P[X]$. That is, $P[X]$ denotes the set of all languages recognized by a polynomial time-bounded deterministic oracle Turing machine with the oracle X. Likewise, we get the relativized classes $NP[X]$ and $coNP[X]$. Although $P = ?NP$ problem is a very hard problem, many results about the relativized version of this problem are known. For example, it is well-known that (Baker et al. [3]) there exist oracles A and B such that $P[A] = NP[A]$ and $P[B] \neq NP[B]$. There are important relationships between unrelativized complexity classes (i.e. P, NP, etc.) and relativized complexity classes (i.e. $P[X]$, $NP[X]$, etc.) relative to "the majority" of oracles.

There are two well-known approach to formalize the concept of "the majority" of oracles, i.e. "almost all" oracles. The one is a probabilistic formalization in which, for a given class $\mathcal{K}$ of oracles, we consider $\mathcal{K}$ contains almost all oracles if $\mathcal{K}$ has Lebesgue measure one. The other is a topological formalization in which we consider a given class $\mathcal{K}$ contains almost all oracles if $\mathcal{K}$ is comeager. While the concept of random oracles plays an important role in the probabilistic approach, the concept of Cohen-Feferman generic oracles plays an important role in the topological approach.

As we stated in the previous paragraph, there are important relationships between unrelativized complexity classes and relativized complexity classes relative to "the majority" of oracles. For example, if the set of all oracles X such that $P[X] \neq NP[X] \cap coNP[X]$ is comeager, then there exists a Cohen-Feferman generic oracle G such that $P[G] \neq NP[G] \cap coNP[G]$, and hence, by [7, Theorem 2.3], we have $P \neq NP$ (unrelativized). Moreover, in Chapter 7, we show the relationship between random oracles and unrelativized complexity classes.

By the way, recall that many assertions about graphs and Turing machines can be interpreted as assertions about the propositional calculus. In particular, SAT, the set of satisfiable formulas of the propositional calculus, is an important example of a NP-complete set.

2.2 Computational complexity

Sets and functions. The set of all natural numbers is denoted by $\mathbb{N} = \{0, 1, 2, \dots\}$. For a function f and a set $D \subseteq \mathrm{dom}(f)$, $f \upharpoonright D$ denotes the restriction of f to D. For a set A, $\mathrm{Card}(A)$ denotes its cardinality. If two sets A and B are disjoint (i.e. $A \cap B = \emptyset$), we often write "$A + B$" to denote their union instead of "$A \cup B$," and we call $A + B$ the *disjoint* *union* of A and B. The union of two given functions means the function whose graph is the union of the graphs of the two functions. If f and g are functions such that $\mathrm{dom}(f) \cap \mathrm{dom}(g) = \emptyset$, then we often write "$f + g$" instead of "$f \cup g$" to denote their union.

Oracles. The set of all bit strings is denoted by $\{0,1\}^*$. Let λ be the empty string. We order all bit strings in lexicographic order:

$$\lambda, 0, 1, 00, 01, 11, 000, 001, \ldots$$

The $(n+1)$st string in this list is denoted by $z(n)$. For example, $z(0) = \lambda$ and $z(1) = 0$. A subset of $\{0,1\}^*$ is called an *oracle* or a *language*, according to the context. We identify an oracle with its characteristic function; thus, an oracle is a function from $\{0,1\}^*$ to $\{0,1\}$. The class of all oracles is denoted by $\mathcal{C}$. When we talk about Lebesgue measure and topology of $\mathcal{C}$, we consider the class $\mathcal{C}$ to be the Cantor space by identifying each natural number n with $z(n) \in \{0,1\}^*$.

Suppose that A and B are oracles. $A \oplus B$ denotes the join of A and B. The only one important property of the join is that its polynomial time many-one degree is the supremum of those of A and B. According to the textbook of complexity theory [5], we adopt the language $\{u0 : u \in A\} \cup \{v1 : v \in B\}$ as a formal definition of the join; of course, there are different ways to define the join (see e.g. Rogers [27]). $P[A]$ denotes the set of all oracles which are polynomial time Turing reducible to A. "$A \equiv_T^P B$" means that A and B are polynomial time Turing equivalent. "$A \equiv B$ (mod. finite)" means that the following set is finite : $\{x \in \{0,1\}^* : A(x) \neq B(x)\}$. Suppose that $M[X]$ is an oracle Turing machine and that A is an oracle. Then, $\mathrm{Lang}(M[A])$ denotes the language accepted by the machine $M[X]$ with the oracle A.

Complexity classes. For each oracle A, Book [8] introduced the computational complexity class $NPQUERY[A]$ as follows. A language B belongs to $NPQUERY[A]$ if $B = \mathrm{Lang}(M[A])$ for some nondeterministic oracle machine $M[X]$ such that $M[X]$ uses a polynomial amount of work space and make a polynomial number of queries to associated oracle in each computation. It was shown in Balcázar et al. [4] that for any oracle A, $NPQUERY[A] = NP[QBF \oplus A]$, where QBF is a well-known $PSPACE$-complete set.

A language L belongs to the computational complexity class R if and only if there exists a probabilistic polynomial time Turing machine M and a positive constant $\varepsilon < 1/2$ such that (1) and (2) below hold for every bit string u :

(1) $u \in L$ if and only if $\mathrm{Prob}[M \text{ accepts } u] \geq (1/2) + \varepsilon$,

(2) $u \notin L$ if and only if $\mathrm{Prob}[M \text{ accepts } u] = 0$.

It is well known that $P \subseteq R \subseteq NP$. See [5] for more about computational complexity and oracle Turing machines. For the Cantor space and arithmetical predicates, see [27].

2.3 Random oracles and generic oracles

Random oracles. Suppose that for each bit string, we decide whether the bit string belongs to an oracle A or not by tossing a fair coin. Then, A is called a *random oracle* (Bennet and Gill [6]).

The reader would think that the concept of random oracles is a little bit strange term; in fact, by the above classical definition of random oracles, we do not have defined "the set of all random oracles" as a particular subset of the Cantor space but have defined a probability distribution on the Cantor space. This probability distribution is the exactly same thing as Lebesgue measure on the Cantor space. Thus, a statement of the following form (1) means the statement (2) below it.

(1) "If A is a random oracle, then the assertion $\cdots$ holds with probability one."

(2) "The set of all oracles A for which the assertion $\cdots$ holds has Lebesgue measure one in the Cantor space."

In Bennet and Gill [6], it was shown that if A is a random oracle, then we have $P[A] \neq NP[A]$ with probability one.

Cohen-Feferman generic oracles. A function S is called a *forcing condition* (condition, for short) if $\mathrm{dom}(S)$ is a finite subset of $\{0,1\}^*$ and we have $\mathrm{ran}(S) \subseteq \{0,1\}$. A collection D of conditions is called *dense* if for every condition S, there exists a condition $T \in D$ such that $S \sqsubseteq T$. An oracle G is called a *Cohen-Feferman generic oracle* (or, a generic oracle) if for any collection D of conditions such that D is arithmetical and dense, there exists a condition S such that $S \in D$ and $S \sqsubseteq G$. Such definitions of dense sets and generic oracles appear e.g. in Definition 1.1 of [7]. It is well-known that the collection of all Cohen-Feferman generic oracles form a comeager set in the Cantor space [16, 12]. And, it is known that if G is a Cohen-Feferman generic oracle, then we have $P[G] \neq NP[G]$; for a proof of this fact, see [13, 26, 7].

For general knowledge about the theory of forcing, see standard textbooks of axiomatic set theory [17, 20].

2.4 The relativized propositional calculus

In this section, by introducing new notation, we review concepts relating to the relativized propositional calculus, such as the connective $\xi^n(q_1, \ldots, q_n)$, the Boolean function $A^n(q_1, \ldots, q_n)$, t-generic oracles and r-generic oracles.

Convention: $\mathrm{Str}(n)$. Suppose that n is a positive integer. We define a set $\mathrm{Str}(n)$ of bit strings as follows.

$$\begin{aligned}\mathrm{Str}(n) &=_{\mathrm{def.}} \{z(m) : 0 \leq m \leq 2^n - 1\} \\ &= \{0,1\}^{\leq n-1} \cup \{0^n\}.\end{aligned}$$

The relativized propositional calculus. Let m be a natural number such that $0 \leq m \leq 2^n - 1$. Note that $z(2^n - 1 + m)$ is a bit string of length n. Then, let $z_j^{(n,m)}$ be the jth bit of the string $z(2^n - 1 + m)$ (for $j = 1, \ldots, n$): that is, $z(2^n - 1 + m) = z_1^{(n,m)} \cdots z_n^{(n,m)}$. We use the auxiliary symbol $z_j^{(n,m)}$ only in this section. Let A be an oracle. Intuitively speaking, A^n denotes the n-ary Boolean function such that the following diagram commutes, where "$\simeq$" means the isomorphism with respect to lexicographic order i.e. $z(2^n - 1 + m) \mapsto z(m)$.

$$\begin{tikzcd}
\{0,1\}^n \arrow[r, "A^n"] \arrow[d, "\simeq"'] & \{0,1\} \\
\mathrm{Str}(n) \arrow[ur, "A \restriction \mathrm{Str}(n)"'] &
\end{tikzcd}$$

More formally, for each natural number m such that $0 \leq m \leq 2^n - 1$, we define $A^n(z_1^{(n,m)}, \ldots, z_n^{(n,m)})$ as to be $A(z(m))$. This rather obscure definition of A^n is forced on us in place of the more direct $A^n(u_1, \ldots, u_n) = A(u)$, because we want that the information contained in A^n be preserved in A^{n+1}, and also because a predicate in a tautology must have a definite number of arguments. The corresponding string $z(m)$ is very simply obtained from the bit string $u = z_1^{(n,m)} \cdots z_n^{(n,m)}$: if u is 0^n then $z(m) = \lambda$; otherwise, first, delete from u the first 1 from the left and all the 0's at its left, then the resulting string is $z(m-1)$, and $z(m)$ is easily obtained.

We introduce an n-ary connective $\xi^n(q_1, \ldots, q_n)$, where q_j's are propositional variables. For a given oracle A, we interpret the connective ξ^n to the Boolean function A^n. *The relativized propositional calculus* [13] is a system obtained by adding the set of connectives $\{\xi^n : n = 1, 2, 3, \ldots\}$ to the propositional calculus. A formula of the relativized propositional calculus is often called a *relativized formula*; it is called a *query free formula* if it has no occurrence of ξ^n's.

r-query tautologies. Suppose that r is a positive integer. We consider a relativized formula with r occurrences of additional connectives; at the expense of adding dummy variables, it can be put in the form:

$$[(a_1 \Leftrightarrow \xi^{n_1}(q_{1,1}, \ldots, q_{1,n_1})) \wedge \cdots \wedge (a_r \Leftrightarrow \xi^{n_r}(q_{r,1}, \ldots, q_{r,n_r}))] \Rightarrow H,$$

where H is a query free formula and a_i's and $q_{i,j}$'s are propositional variables. According to the terminology of [13], we call a relativized formula of the above form an *r-query formula*. Note that it is only the number r of queries which is relevant to this definition, not their length $n_1, \ldots, n_r$.

For an oracle A, $TAUT[A]$ is the set of all (binary representations of) relativized formulas that are tautologies with respect to A. $rTAUT[A]$ is the set of all (binary representations of) r-query formulas that are tautologies with respect to A. Each member of $rTAUT[A]$ is called an *r-query tautology* with respect to A. Moreover, by $TAUT$, we denote the collection of all (binary representations of) tautologies of usual propositional calculus. Let X be a unary predicate symbol denoting membership to a given oracle and y be a variable for a bit string. Membership to the set $TAUT[X]$ is expressed by an arithmetical predicate, that we denote $TAUT(X)(y)$. For each r, a predicate $rTAUT(X)(y)$ is similarly defined. As is well-known, $TAUT[X]$ is uniformly $coNP[X]$-complete [13, p.68]: that is, for any polynomial time-bounded nondeterministic oracle Turing machine $M[X]$, there exists a function f such that f is polynomial time computable (without an oracle) and for any oracle A and for any bit string u, $M[A]$ rejects u if and only if $f(u)$ belongs to $TAUT[A]$.

r-generic oracles. Suppose F is a relativized formula. We say "a forcing condition S *forces that F is a tautology*" if F is a tautology with respect to any oracle A that is an extension of S. We also say "S forces $F \in TAUT[X]$," or more simply, we say "S forces F." An oracle A is called a *t-generic oracle* [13] if there exists a polynomial p such that for any formula $F \in TAUT[A]$, there exists a finite portion $S \sqsubseteq A$ such that the size of (the domain of) S is at most $p(|F|)$ and S forces F. Suppose r is a positive integer. An oracle A is called an *r-generic oracle* (in Dowd's sense) [13] if it satisfies the definition of a t-generic oracle with $rTAUT[A]$ in place of $TAUT[A]$. Of course, this concept of r-generic oracles is completely different from the concept of Cohen-Feferman generic oracles. The relationship of these two concepts of generic oracles is studied in [13, Theorem 12] and in [28, §4]. That is, [13, Theorem 12] says that any Cohen-Feferman generic oracle is not 1-generic in Dowd's sense. The contents of [28, §4] is discussed in Chapter 5 of this thesis. Throughout this article, an "r-generic oracle" means that in Dowd's sense.

Convention: $\triangle^{(n,i)}[S]$. For each natural numbers i and j, we reserve the propositional variables $a^{(i)}$ and $q_j^{(i)}$ for special usage. Suppose that S is a forcing condition whose domain is a subset of $\mathrm{Str}(n)$. For each bit string $z(m) \in \mathrm{dom}(S)$, let $c_S^{(n,m)}$ be the value of $S(z(m))$; in other words, $c_S^{(n,m)}$ is the value of $S^n(z_1^{(n,m)}, \ldots, z_n^{(n,m)})$. We use the auxiliary symbol $c_S^{(n,m)}$ in this paragraph only. Then, for each natural number i, we define a query free formula $\triangle^{(n,i)}[S]$ as follows, where m varies over all m such that $z(m) \in \mathrm{dom}(S)$. (cf. $\hat{S}_1$ in [13, p.71].)

$$\triangle^{(n,i)}[S] \equiv_{\mathrm{def.}}$$
$$\bigwedge\nolimits_m \Big([(q_1^{(i)} \Leftrightarrow z_1^{(n,m)}) \wedge \cdots \wedge (q_n^{(i)} \Leftrightarrow z_n^{(n,m)})] \Rightarrow (a^{(i)} \Leftrightarrow c_S^{(n,m)}) \Big).$$

Then, for any oracle A, the necessary and sufficient condition for "$S \sqsubseteq A$" is that the following relativized formula is a tautology with respect to A.

$$(a^{(i)} \Leftrightarrow \xi^n(q_1^{(i)}, \ldots, q_n^{(i)})) \Rightarrow \triangle^{(n,i)}[S].$$

In particular, if the domain of S is $\{0,1\}^n$, then S forces the following formula.

$$(a^{(i)} \Leftrightarrow \xi^n(q_1^{(i)}, \ldots, q_n^{(i)})) \Leftrightarrow \triangle^{(n,i)}[S].$$

Chapter 3

t-generic Oracles

In this chapter, we present an explicit example of a $coNP[X]$-predicate $\varphi(X)(y)$ with the following property: for each oracle A and for each positive integer n, a lower bound for its forcing complexity is (uniformly) given as follows.

$$\mathrm{FC}(\varphi(X)(y), A, n) \geq \frac{2^{n-1} - n + 1}{n}.$$

By this example, an alternative proof of **Dowd's Lemma 7** is immediately derived.

Dowd's Lemma 7 ([13, Lemma 7]) *t-generic oracles do not exist.*

3.1 Ceiling-generic oracles

To explain our motive for the argument, we first show a weak version of **Dowd's Lemma 7**.

Definition 3.1 *Suppose that $\varphi(X)(y)$ is an arithmetical predicate, where X is a unary symbol denoting membership to a given oracle, and y is a variable for a bit string.*

1. ([26]. See also [31]) *For each oracle A, $\varphi[A]$ denotes the set $\{u \in \{0,1\}^* : \varphi(A)(u)\}$.*

2. ([28]) *Suppose that $\varphi(X)(y)$ is finitely testable, G is an oracle, and f is a function from $\mathbb{N}$ to $\mathbb{N}$. G is* f-ceiling-generic *for $\varphi(X)(y)$ (f-c-generic for $\varphi(X)(y)$, for short), if for each natural number n, we have*

   $$\mathrm{FC}(\varphi(X)(y), G, n) \leq f(n).$$

   G is ceiling-generic *for $\varphi(X)(y)$ (c-generic for $\varphi(X)(y)$, for short), if there exists a polynomial p such that G is p-ceiling-generic for $\varphi(X)(y)$.* □

The following is a weak version of **Dowd's Lemma 7**.

Lemma 3.1 *The set of all t-generic oracles has Lebesgue measure zero in the Cantor space.*

Proof: ([28]) The oracle-dependent language $CORANGE[X]$ is well-known among the reader of Bennet and Gill [6]. We express membership to this language by a predicate $CORANGE(X)(y)$. More precisely, the predicate $CORANGE(X)(y)$ is defined as the following assertion:

$$\text{“}\neg\exists u \text{ such that } y = X(u1)X(u10)X(u100)\cdots X(u10^{|u|-1}).\text{”}$$

Note that y and u in the above assertion have the same length, and hence the above assertion is finitely testable. Recall that $TAUT[X]$ is uniformly $coNP[X]$-complete. Thus, there exists a function f such that f is computable (without an oracle) in polynomial time, and for any oracle A and any bit string w, $CORANGE(A)(w)$ holds if and only if we have $f(w) \in TAUT[A]$. Therefore, if A is a t-generic oracle, then A is ceiling-generic for the predicate $CORANGE(X)(y)$. Hence $CORANGE[A]$ is a finite set; indeed, letting p be a polynomial for which A is p-ceiling-generic, whenever 2^n is sufficiently larger than $p(n)$, $CORANGE[A]$ does not contain any y of length n, since a condition of size $p(n)$ cannot force all the u's of size n so that $y \neq X(u1)X(u10)X(u100)\cdots X(u10^{|u|-1})$. Thus, all t-generic oracles belong to the following class: $\{X : CORANGE[X] \in NP[X]\}$. However, by [6], this class has Lebesgue measure zero. □

3.2 Exponential lower bound

In this section, we state and prove our main theorem of this chapter (Theorem 3.2). We introduce a predicate $NotCNum(X)(y)$. Let X be a unary predicate symbol for membership to an oracle and let y be a variable for a bit string. We define an arithmetical predicate $ConsecNum(X)(y)$ as the following assertion: "y is not the empty bit string and, letting n be the natural number such that $|y| = n+1$, there exists a natural number j such that we have $|z(j)| = n$ and $y = X(z(j))X(z(j{+}1))\cdots X(z(j{+}n))$." "$ConsecNum$" is an abbreviation of "consecutive numbers." Then, we define a predicate $NotCNum(X)(y)$ as to be the *negation* of $ConsecNum(X)(y)$.

$$NotCNum(X)(y) \equiv_{\mathrm{def.}} \neg ConsecNum(X)(y).$$

Clearly, $NotCNum(X)(y)$ is a finitely testable $coNP[X]$-predicate.

Theorem 3.2 ([30]) *Assume that A is an arbitrary oracle and n is a positive integer. Then, a lower bound for the forcing complexity of the predicate $NotCNum(X)(y)$ is (uniformly) given as follows.*

$$\mathrm{FC}(NotCNum(X)(y), A, n) \geq \frac{2^{n-1} - n + 1}{n}.$$

Proof: Assume for a contradiction that for an oracle A and for a natural number $n+1$, we have the following.

$$\mathrm{FC}(NotCNum(X)(y), A, n+1) < \frac{2^n - n}{n+1}.$$

Among bit strings of length $n+1$, at most 2^n bit strings v's make the assertion $ConsecNum(A)(v)$ true. Hence, there exists at least one bit string u such that $|u| = n+1$ and $NotCNum(A)(u)$ is true. We fix such a u. Then, there exists a finite portion $S \sqsubseteq A$ such that S forces $NotCNum(X)(u)$ and, letting

$$d =_{\text{def.}} \mathrm{Card}(\mathrm{dom}(S)),$$

we have $d \leq \mathrm{FC}(NotCNum(X)(y), A, n+1)$. By our assumption for a contradiction, we have the following.

$$d < \frac{2^n - n}{n+1}.$$

Hence, we have $n \cdot (d+1) < 2^n - d$. Therefore, the cardinality of the set

$$Remainder =_{\text{def.}} \{0,1\}^n \setminus \mathrm{dom}(S)$$

is greater than $n \cdot (d+1)$. However, *Remainder* is a disjoint union of at most $d+1$ closed intervals (with respect to lexicographic order), and hence at least one of them, say

$$I =_{\text{def.}} [z(j), z(j+k)] \subseteq Remainder,$$

must have cardinality strictly greater than n. Hence, there exists an oracle B extending S such that $ConsecNum(B)(u)$ is true: in other words, S does not force $NotCNum(X)(u)$, and we get a contradiction. □

In our proof of **Lemma 3.1**, the predicate $CORANGE(X)(y)$ played an important role. By using the predicate $NotCNum(X)(y)$ in place of

$CORANGE(X)(y)$, we immediately get an alternative proof of **Dowd's Lemma 7**.

Proof of **Dowd's Lemma 7**: ([30]) Assume for a contradiction that an oracle A is t-generic. Then, A is ceiling-generic for $NotCNum(X)(y)$, in other words, the forcing complexity of $NotCNum(X)(y)$ relative to A is at most polynomial; the argument is similar to **Lemma 3.1**. By **Theorem 3.2**, we have a contradiction. □

Chapter 4

r-generic Oracles

In this chapter, we reconstitute a proof of the following fact by investigating oracles' hierarchy with respect to forcing complexity.

Dowd's Theorem 10 ([13, Theorem 10]) *Suppose that r is a positive integer. Then, r-generic oracles in the sense of Dowd form a measure one set in the Cantor space.*

The contents of this chapter is based on [30].

4.1 What is difficult in the original proof?

In this section, we explain what is difficult in the original proof. Dowd proved his **Theorem 10** by using the following result.

Dowd's Lemma 9 ([13, Lemma 9]) *If F is a 1-query formula which is a tautology with respect to some X then there is a unique minimal set S of queries which force F to be a tautology (we say F specifies S).*

That is, F specifies the unique forcing condition S of the following property: if T is a forcing condition then the necessary and sufficient condition for "T forces F" is "$S \sqsubseteq T$."

Note that, in the above **Lemma**, the assumption of 1-query is critical. For example, let $G_\star$ be the following formula:

$$\Big((a^{(2)} \Leftrightarrow \xi^4(q_1^{(2)}, 0, 1, q_4^{(2)})) \wedge (a^{(3)} \Leftrightarrow \xi^4(q_1^{(3)}, 1, 1, q_4^{(3)})) \wedge$$
$$(q_1^{(1)} \Leftrightarrow q_1^{(3)}) \wedge (q_1^{(3)} \Leftrightarrow 1) \wedge (q_4^{(1)} \Leftrightarrow q_4^{(3)}) \wedge (q_4^{(3)} \Leftrightarrow 1)\Big)$$
$$\Rightarrow (a^{(1)} \Leftrightarrow \neg a^{(3)}).$$

Next, let $F_\star$ be the following "3-query formula."

$$(a^{(1)} \Leftrightarrow \xi^4(q_1^{(1)}, 1, 0, q_4^{(1)})) \Rightarrow G_\star.$$

Observe that $F_\star$ does not specify the unique minimal set of queries in the following sense. There are two minimal sets of queries that force F, one is {"$0 \Leftrightarrow \xi^4(1,1,0,1)$," "$1 \Leftrightarrow \xi^4(1,1,1,1)$"}, and the other is {"$1 \Leftrightarrow \xi^4(1,1,0,1)$," "$0 \Leftrightarrow \xi^4(1,1,1,1)$"}. So, we must be careful when we deal with r-query formulas for r strictly greater than 1.

By the way, Dowd began the proof of [13, Theorem 10] by showing the existence of 1-generic oracles. More precisely, let k be a sufficiently large natural number, p be the polynomial $p(\ell) = \ell^k$ (in his notation, roughly speaking, the variable ℓ denotes the length of a given relativized formula) and let A^n be an n-ary Boolean function which is 1-generic with respect to the polynomial p ; he counted the number of $(n+1)$-ary Boolean function A^{n+1} which is compatible with A^n and is not 1-generic with respect to the polynomial p. Let $N = 2^n$. Then, he showed, by using [13, Lemma 9], among the 2^N Boolean function A^{n+1} compatible with A^n, the number of Boolean function satisfying the above requirement is $o(2^N)$. After that, he wrote as follows, where his symbol "X^n" means, in our notation, the connective "ξ^n" or the n-ary Boolean function "X^n" with respect to an oracle X depending on the context.

Citation 4.1 ([13, p.70 line 33 – p.71 line 6])

> For the induction step, define $k_1 = k$, $k_{s+1} = k \cdot k_s + 1$, and suppose X^n has been constructed deciding the s-query formulas involving X^i with $i \leq n$, with polynomial ℓ^{k_s} for $1 \leq s \leq r$. Proceeding as above, the number of X^{n+1} ruled out by 1-query formulas is determined as above. Now suppose F is $(p \Leftrightarrow X^{n+1}(q_1, \ldots, q_{n+1})) \Rightarrow G$, where G is an $(s-1)$-query formula. We may further suppose that if G involves X^i then $i \leq n$, so that with respect to X^n, F is either never a tautology or specifies some set S of queries.

Why may we suppose so? How to deal with formulas such as the above $F_\star$ would be not so clear from the argument in **Citation 4.1**. May we suppose "if $G_\star$ involves ξ^i then $i \leq 3$"? How should we proceed induction? Thus, in the following sections, we would like to present some facts implicitly shown in [13, pp.70-71], and we would like to describe the structure of induction explicitly. We will see that Dowd's idea is completely right.

4.2 Disentangled r-query tautologies

Now, we may think that it is a formula of the following form that appeared as the formula F in **Citation 4.1**.

$$\Big((a^{(1)} \Leftrightarrow \xi^4(1, q_2^{(1)}, q_3^{(1)}, q_4^{(1)})) \wedge$$
$$(a^{(2)} \Leftrightarrow \xi^4(0, q_2^{(2)}, q_3^{(2)}, q_4^{(2)})) \wedge$$
$$(a^{(3)} \Leftrightarrow \xi^4(0, q_2^{(3)}, q_3^{(3)}, q_4^{(3)}))\Big) \Rightarrow H,$$

where H is a query free formula. We shall consider not only formulas of the above form but also formulas of the following form.

$$\Big((a^{(1)} \Leftrightarrow \xi^4(q_1^{(1)}, 1, 0, q_4^{(1)})) \wedge \\ (a^{(2)} \Leftrightarrow \xi^4(q_1^{(2)}, 0, 1, q_4^{(2)})) \wedge \\ (a^{(3)} \Leftrightarrow \xi^4(q_1^{(3)}, 1, 1, q_4^{(3)}))\Big) \Rightarrow H.$$

In the following **Definition**, note that $r(r-1)/2$ is the binomial coefficient:

$$\frac{r(r-1)}{2} = \binom{r}{2} = {}_rC_2.$$

Definition 4.1 *Suppose that r and n are positive integers such that we have $r \geq 2$. A triple $\delta = (t, B, f)$ is called an* (r, n)-disentangled matrix *if the following three requirements are satisfied. The set of all (r, n)-disentangled matrices is denoted by* $\mathrm{DEM}(r, n)$.

1. *t is a positive integer satisfying the following inequality.*

   $$\log_2 r \leq t \leq \min\{n, r(r-1)/2\}.$$

2. *B is a matrix of type (r, t) such that each element is 0 or 1 and such that rows are pairwise different.*

3. *$f : \{1, \cdots, t\} \to \{1, \ldots, n\}$ is an order preserving mapping; we shall often denote f by the sequence $\langle f(1), \ldots, f(t)\rangle$.*

In the following four **Examples**, we explain how we shall use the concept of (r, n)-disentangled matrices. To clarify our idea, we observe the particular case where $r = 3$ and $n = 4$. Throughout these **Examples**, we assume that

$\delta = (2, B, f)$ is a $(3, 4)$-disentangled matrix such that $f = \langle 2, 3\rangle$ and

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

Example 4.1 Suppose that Q is a matrix of type $(3, 4)$ such that each component is 0 or 1 or a propositional variable. Then, we say "*δ expresses a submatrix of Q*" if the jth column of B is identical with the $f(j)$th column of Q (for $j = 1, 2$), i.e. if Q is of the following form.

$$Q = \begin{bmatrix} * & 1 & 0 & * \\ * & 0 & 1 & * \\ * & 1 & 1 & * \end{bmatrix}.$$

Thus, we consider the sequence $f = \langle 2, 3\rangle$ to be the list of names of the columns. □

Example 4.2 We regard the disentangled matrix δ as a partition of the set $\mathrm{Str}(4)$. For each $i \in \{1, 2, 3\}$, we define a set $\mathrm{Str}(4, \delta, i)$ of bit strings as follows.

$$\mathrm{Str}(4, \delta, i) =_{\mathrm{def.}} \{z(m) \in \mathrm{Str}(4) :\ \text{The string } z(2^4 - 1 + m) \text{ is of the form} \\ u_1 b_1^{(i)} b_2^{(i)} u_4,\ \text{where } b_1^{(i)} b_2^{(i)} \text{ is the } i\text{th row of } B \text{ and } \{u_1, u_4\} \subseteq \{0, 1\}\}.$$

Then, we get the following partition of the set $\mathrm{Str}(4)$, where $+$ means disjoint union.

$$\mathrm{Str}(4) = \mathrm{Str}(4, \delta, 1) + \mathrm{Str}(4, \delta, 2) + \mathrm{Str}(4, \delta, 3) + (\mathrm{Str}(4) \setminus \textstyle\bigcup_{1 \leq i \leq 3} \mathrm{Str}(4, \delta, i)).$$

For any oracle A, the partial function $A \restriction \mathrm{Str}(4, \delta, 1)$ determines queries of the form "$a \Leftrightarrow \xi^4(q_1, 1, 0, q_4)$." Similar fact holds for $\mathrm{Str}(4, \delta, 2)$ and for $\mathrm{Str}(4, \delta, 3)$. □

Example 4.3 We identify the disentangled matrix δ to be the following operation on arbitrary relativized formula F.

$$\begin{aligned}\delta(F)\equiv_{\text{def.}}F[1/q_2^{(1)}][0/q_3^{(1)}]\\ [0/q_2^{(2)}][1/q_3^{(2)}]\\ [1/q_2^{(3)}][1/q_3^{(3)}].\end{aligned}$$

The right-hand side denotes substitution. For example,

$$\delta(q_2^{(1)}\vee(q_1^{(1)}\wedge q_2^{(2)}))\equiv(q_2^{(1)}\vee(q_1^{(1)}\wedge q_2^{(2)}))[1/q_2^{(1)}][0/q_2^{(2)}]\equiv 1\vee(q_1^{(1)}\wedge 0).$$

We often write δF instead of $\delta(F)$. If F is a tautology with respect to an oracle A, then δF is also a tautology with respect to A.

In particular, for a query free formula H, we define a relativized formula $\delta\langle 3,4,H\rangle$ as follows.

$$\begin{aligned}\delta\langle 3,4,H\rangle\equiv_{\text{def.}}\text{“}\Big((a^{(1)}\Leftrightarrow\xi^4(q_1^{(1)},1,0,q_4^{(1)}))\wedge\\ (a^{(2)}\Leftrightarrow\xi^4(q_1^{(2)},0,1,q_4^{(2)}))\wedge\\ (a^{(3)}\Leftrightarrow\xi^4(q_1^{(3)},1,1,q_4^{(3)}))\Big)\Rightarrow\delta H.\text{”}\end{aligned}$$

In the following, each expression of the form $\delta\triangle^{(n,i)}[S]$ denotes $\delta(\triangle^{(n,i)}[S])$. For the symbol of the form $\triangle^{(n,i)}[S]$, see the last paragraph of Chapter 2 of this thesis. Suppose that each S_i $(i=1,2,3)$ is a forcing condition such that $\mathrm{dom}(S_i)\subseteq\mathrm{Str}(4,\delta,i)$. Let F_3 be the formula obtained from $\delta\langle 3,4,H\rangle$ by substituting $\delta\triangle^{(4,3)}[S_3]$ for "$a^{(3)}\Leftrightarrow\xi^4(q_1^{(3)},1,1,q_4^{(3)})$." Let $F_{2,3}$ be the formula obtained from F_3 by substituting $\delta\triangle^{(4,2)}[S_2]$ for "$a^{(2)}\Leftrightarrow\xi^4(q_1^{(2)},0,1,q_4^{(2)})$." Moreover, let $F_{1,2,3}$ be the query free formula obtained from $F_{2,3}$ by substituting $\delta\triangle^{(4,1)}[S_1]$ for "$a^{(1)}\Leftrightarrow\xi^4(q_1^{(1)},1,0,q_4^{(1)})$." We consider the following four assertions.

(1) The query free formula $F_{1,2,3}$ is a tautology.

(2) The forcing condition S_1 forces the formula $F_{2,3}$.

(3) The forcing condition S_1+S_2 forces the formula F_3.

(4) The forcing condition $S_1+S_2+S_3$ forces the formula $\delta\langle 3,4,H\rangle$.

Suppose that a truth assignment for propositional variables is given such that the three query free formulas $\delta\triangle^{(4,1)}[S_1]$, $\delta\triangle^{(4,2)}[S_2]$ and $\delta\triangle^{(4,3)}[S_3]$ are true with respect to this truth assignment. Then, it is not hard to see that we can extend the forcing condition $S_1+S_2+S_3$ to an oracle A (i.e. $S_1+S_2+S_3\sqsubseteq A$) so that the three requirements "$a^{(1)}\Leftrightarrow A^4(q_1^{(1)},1,0,q_4^{(1)})$," "$a^{(2)}\Leftrightarrow A^4(q_1^{(2)},0,1,q_4^{(2)})$" and "$a^{(3)}\Leftrightarrow A^4(q_1^{(3)},1,1,q_4^{(3)})$" are satisfied with respect to the truth assignment. Hence, the assertion (4) implies the assertion (1). On the other hand, S_1 forces the following formula.

$$(a^{(1)}\Leftrightarrow\xi^4(q_1^{(1)},1,0,q_4^{(1)}))\Rightarrow\delta\triangle^{(4,1)}[S_1].$$

Similar facts hold for S_2 and S_3. Therefore, the above four assertions are equivalent. □

Example 4.4 We observe that a weak version of Dowd's Lemma 9 [13, Lemma 9] holds for tautologies of the form $\delta\langle 3,4,H\rangle$.

Now, suppose that H is a query free formula and E is a set of bit strings satisfying the following inclusion.

$$\mathrm{Str}(4,\delta,2)+\mathrm{Str}(4,\delta,3)\subseteq E\subseteq\mathrm{Str}(4)\setminus\mathrm{Str}(4,\delta,1).$$

Suppose that A_0 is a forcing condition whose domain is E and that the formula $F\equiv_{\text{def.}}\delta\langle 3,4,H\rangle$ is a tautology with respect to some oracle A extending A_0. In general, a minimal set of queries that forces F is *not unique*; recall the example of the formula $F_\star$ in the previous section. However, letting $S_i=_{\text{def.}}A_0\restriction\mathrm{Str}(4,\delta,i)$ for each $i\in\{2,3\}$, the "1-query formula"

$F_{2,3}$ defined as in **Example 4.3** is a tautology with respect to A. Hence, by **Dowd's Lemma 9** [13, Lemma 9], there exists the unique forcing condition S_1 satisfying the following two requirements.

- For any forcing condition T_1, the necessary and sufficient condition for "T_1 forces $F_{2,3}$" is "$S_1 \sqsubseteq T_1$."
- $\mathrm{dom}(S_1)$ is a subset of $\mathrm{Str}(4, \delta, 1)$.

In particular, by considering the case where $T_1 = A \upharpoonright \mathrm{Str}(4)$, we have $S_1 \sqsubseteq A$. Note the following two facts. First, for any forcing condition T_1 such that $\mathrm{dom}(T_1) \subseteq \mathrm{Str}(4, \delta, 1)$, the following three assertions are equivalent (see **Example 4.3**).

- T_1 forces $F_{2,3}$.
- $T_1 + A_0$ forces F.
- A_0 forces F_1, where F_1 is the formula obtained from F by substituting $\delta\triangle^{(4,1)}[T_1]$ for "$a^{(1)} \Leftrightarrow \xi^4(q_1^{(1)}, 1, 0, q_4^{(1)})$."

Second, S_1 is uniquely determined only by F and A_0: that is, determined without using information on the remaining part $A \setminus A_0$. Because, F and A_0 determine the formula $F_{2,3}$, and $F_{2,3}$ specifies S_1. □

More generally, we define as follows.

Definition 4.2 *Suppose that r is a positive integer such that $r \geq 2$.*

1. *Suppose that n is a positive integer and $\delta = (t, B, f)$ is an (r, n)-disentangled matrix. For a matrix Q of type (r, n), we say "δ* expresses a submatrix *of Q" if the jth column of B is identical with the $f(j)$th column of Q for each $j = 1, \ldots, t$. For each integer $i = 1, \ldots, r$, we define a set $\mathrm{Str}(n, \delta, i)$ as follows. $\mathrm{Str}(n, \delta, i)$ is the collection of all bit strings $z(m) \in \mathrm{Str}(n)$ such that, letting $u_1 \cdots u_n$ be the bit string $z(2^n - 1 + m) \in \{0, 1\}^n$, the bit string $u_{f(1)}u_{f(2)} \cdots u_{f(t)}$ is identical with the ith row of the matrix B. Moreover, from a given relativized formula F, we make a formula $\delta(F)$ by the following substitution, where each $b_j^{(i)}$ denotes the (i, j)-component of the matrix B.*

$$\begin{aligned}\delta(F) \equiv_{\text{def.}} F[b_1^{(1)}/q_{f(1)}^{(1)}] \cdots [b_t^{(1)}/q_{f(t)}^{(1)}] \\ \cdots [b_j^{(i)}/q_{f(j)}^{(i)}] \cdots \\ [b_1^{(r)}/q_{f(1)}^{(r)}] \cdots [b_t^{(r)}/q_{f(t)}^{(r)}].\end{aligned}$$

We often write δF instead of $\delta(F)$. In particular, for a given query free formula H, we abbreviate "$\delta \natural\langle r, n, H\rangle$" to "$\delta\langle r, n, H\rangle$," where $\natural\langle r, n, H\rangle$ is the r-query formula defined as follows.

$$\natural\langle r, n, H\rangle \equiv_{\text{def.}} \text{“}\Big(\bigwedge_{i=1}^{r}(a^{(i)} \Leftrightarrow \xi^n(q_1^{(i)}, \ldots, q_n^{(i)}))\Big) \Rightarrow H.\text{”}$$

2. *Assume that F is a relativized formula. We call F a* disentangled *r-query formula if F is of the form $\delta\langle r, n, H\rangle$, where n is a natural number, δ is an (r, n)-disentangled matrix and H is a query free formula. If F is a disentangled r-query formula and F is a tautology with respect to an oracle A, then we call F a* disentangled *r-query* tautology with respect to *A.* □

The following is a crucial property of the disentangled r-query tautologies.

Revised Dowd's Lemma 9 (Implicitly shown in [13, Lemma 9]) *Suppose that r and n are positive integers such that $r \geq 2$. Suppose $\delta \in \mathrm{DEM}(r, n)$*

and suppose that H is a query free formula and E is a set of bit strings satisfying the following inclusion.

$$\mathrm{Str}(n,\delta,2)+\cdots+\mathrm{Str}(n,\delta,r)\subseteq E\subseteq \mathrm{Str}(n)\setminus\mathrm{Str}(n,\delta,1).$$

Let A_0 be a forcing condition whose domain is E. Assume that $F \equiv_{\mathrm{def.}} \delta\langle r,n,H\rangle$ is a disentangled r-query tautology with respect to some oracle extending A_0. Then, there exists a forcing condition S_1 such that $\mathrm{dom}(S_1)\subseteq \mathrm{Str}(n,\delta,1)$ *and such that for any forcing condition T_1 whose domain is a subset of* $\mathrm{Str}(n,\delta,1)$*, the following three assertions are equivalent.*

1. *$S_1 \sqsubseteq T_1$.*

2. *$T_1 + A_0$ forces F.*

3. *A_0 forces F_1, where F_1 is the formula obtained from F by substituting $\delta\Delta^{(n,1)}[T_1]$ for "$\delta(a^{(1)} \Leftrightarrow \xi^n(q_1^{(1)},\ldots,q_n^{(1)}))$."*

Such a forcing condition S_1 is uniquely determined only by the disentangled r-query formula F and the forcing condition A_0. And, for any oracle A extending A_0, if F is a tautology with respect to A then A is an extension of S_1. (Because, letting $T_1 \equiv_{\mathrm{def.}} A \restriction \mathrm{Str}(n,\delta,1)$, $T_1 + A_0$ forces F.)

Proof: Proceed as in **Example 4.4.** □

Now, suppose that Q is a matrix of type (r,n) such that each element is 0 or 1. Observe that either Q has a pair of identical rows or there exists $\delta \in \mathrm{DEM}(r,n)$ such that δ expresses a submatrix of Q. If r is fixed, then the cardinality of $\mathrm{DEM}(r,n)$ is at most polynomial of n ; indeed, we have the following, where $R = r(r-1)/2$.

$$\mathrm{Card}(\mathrm{DEM}(r,n)) \le R\cdot\max(\{2^{rt}\cdot\binom{n}{t} : t\le R\}) \le R\cdot 2^{rR}\cdot n^R.$$

Thus, we can reduce a problem of an r-query tautology to polynomial many problems of disentangled r-query tautologies and to polynomial many problems of $(r-1)$-query tautologies; we shall discuss this matter more precisely in the next section. Also, we shall see that by using **Revised Dowd's Lemma 9**, we can apply the argument in [13, pp.70–71] to disentangled r-query tautologies.

4.3 Hierarchy

Our next task is to describe the structure of induction explicitly.

For this purpose, we investigate oracles' hierarchy with respect to forcing complexity. The following Diagram 4.1 illustrates the relationship of oracle classes $r\mathrm{GEN}_i$'s that we are going to define. It is easily seen that $r\mathrm{GEN}_3$ is exactly the collection of all r-generic oracles in Dowd's sense (for $r = 1,2,\cdots$). For each r, the class $r\mathrm{GEN}_2$ is defined as the collection of oracles for which (the binary codes of) disentangled r-query tautologies have forcing complexity at most polynomial. The class $r\mathrm{GEN}_1$ is defined as the collection of oracles for which (the binary codes of) disentangled r-query tautologies have "partial forcing complexity" at most polynomial, that is, the collection of oracles for which "S_1" in **Revised Dowd's Lemma 9** has size at most polynomial.

Diagram 4.1.

$$\begin{tikzcd}
 & 2\mathrm{GEN}_1 \arrow[r, phantom, "\supseteq"] \arrow[d, phantom, sloped, "\supseteq"] & 3\mathrm{GEN}_1 \arrow[r, phantom, "\supseteq"] \arrow[d, phantom, sloped, "\supseteq"] & \cdots \\
 & 2\mathrm{GEN}_2 \arrow[r, phantom, "\supseteq"] \arrow[d, phantom, sloped, "\supseteq"] & 3\mathrm{GEN}_2 \arrow[r, phantom, "\supseteq"] \arrow[d, phantom, sloped, "\supseteq"] & \cdots \\
1\mathrm{GEN}_3 \arrow[r, phantom, "\supseteq"] & 2\mathrm{GEN}_3 \arrow[r, phantom, "\supseteq"] & 3\mathrm{GEN}_3 \arrow[r, phantom, "\supseteq"] & \cdots
\end{tikzcd}$$

In the following, for a relativized formula F, $\ell(F)$ denotes the length of F as a formula: that is, $\ell(F)$ is the number of occurrences of propositional variables, constants (0 and 1), logical connectives ($\neg$, $\wedge$, $\vee$, $\Rightarrow$ and $\Leftrightarrow$), additional connectives (ξ^1, ξ^2, ξ^3, $\ldots$) and punctuation marks (parenthesisses and commas). We do not rigorously formalize syntax of the relativized propositional calculus here. However, we do not admit gates of unbounded fan-in, that is, the following symbols are not our formal symbols.

$$\bigwedge_{i=1}^{r}, \bigvee_{i=1}^{r}.$$

Definition 4.3 *Suppose that r, k, C and n are positive integers. Let* $\mathrm{Func}(n)$ *be the collection of all forcing conditions whose domains are* $\mathrm{Str}(n)$*. In the following, for each $i \in \{1,2\}$, $r\mathrm{GEN}_i(k,C,n)$ is defined as a subset of* $\mathrm{Func}(n)$*.*

1. *For each oracle A, $rTAUT[A;C]$ denotes the set of all formulas F such that we have $F \in rTAUT[A]$ and $\ell(F) \geq C$.*

2. *Assume $r \geq 2$. A forcing condition $X \in \mathrm{Func}(n)$ belongs to the set $r\mathrm{GEN}_1(k,C,n)$ if for every $\delta \in \mathrm{DEM}(r,n)$ and for every query free formula H, letting F be the disentangled r-query formula $\delta\langle r,n,H\rangle$, the following assertion (A1) holds:*

   (A1) "If we have $F \in rTAUT[X;C]$ then there exists a forcing condition S_1 such that $\mathrm{Card}(\mathrm{dom}(S_1)) \leq \ell(F)^k$, $S_1 \sqsubseteq X \upharpoonright \mathrm{Str}(n,\delta,1)$ and $S_1 + (X \upharpoonright E)$ forces F, where $E = \mathrm{Str}(n) \setminus \mathrm{Str}(n,\delta,1)$."

3. *Assume $r \geq 2$. A forcing condition $X \in \mathrm{Func}(n)$ belongs to the set $r\mathrm{GEN}_2(k,C,n)$ if it satisfies the requirement for $r\mathrm{GEN}_1(k,C,n)$ with the following assertion (A2) in place of (A1).*

   (A2) "If we have $F \in rTAUT[X;C]$ then there exists a forcing condition S such that $\mathrm{Card}(\mathrm{dom}(S)) \leq \ell(F)^k$, $S \sqsubseteq X$ and S forces F."

4. *For each $i \in \{1,2\}$, we define a class $r\mathrm{GEN}_i(k,C)$ as the collection of all oracles X such that for each natural number m, the partial function $X \upharpoonright \mathrm{Str}(m)$ belongs to the set $r\mathrm{GEN}_i(k,C,m)$.*

5. *We define $r\mathrm{GEN}_3(k,C)$ as the class of all oracles X such that for all $F \in rTAUT[X;C]$, F is forced by some finite portion $S \sqsubseteq X$ such that $\mathrm{Card}(\mathrm{dom}(S)) \leq \ell(F)^k$.*

6. *For each $i \in \{1,2,3\}$, $r\mathrm{GEN}_i$ denotes the collection of all oracles X such that for some natural numbers k' and C', X belongs to the class $r\mathrm{GEN}_i(k',C')$.* □

We paraphrase the argument in [13, pp.70–71] as follows.

Proposition 4.1 (Implicitly shown in [13, p.70]) *Suppose we have $r \geq 2$. Then, for sufficiently large k and C, we have the following.*

$$\mu(r\mathrm{GEN}_1(k,C)) > 0,$$

where μ denotes Lebesgue measure. Since $r\mathrm{GEN}_1$ is closed under finite changes, we also have $\mu(r\mathrm{GEN}_1) = 1$. In other words, since membership to $r\mathrm{GEN}_1$ is a tail event, by zero-or-one law, we have $\mu(r\mathrm{GEN}_1) = 1$. (For tail events and zero-or-one law, see e.g. Feller [15].)

Proof: The proof is a paraphrase of Dowd's proof of the existence of 1-generic oracles [13, p.70]. Instead of **Dowd's Lemma 9**, we use **Revised Dowd's Lemma 9**. Suppose that k, C and n are positive integers.

We observe the nature of a forcing condition $X \in \mathrm{Func}(n)$ ruled out by a particular formula F, where "ruled out" means that X does not belong to

$r\mathrm{GEN}_1(k, C, n)$. Assume that X is a member of $\mathrm{Func}(n)$ and F is a formula of the form $\delta\langle r, n, H\rangle$, where $\delta = (t, B, f)$ is an (r, n)-disentangled matrix and H is a query free formula. Note that δ and H are uniquely determined by F. Thus, the set $\mathrm{Str}(n, \delta, 1)$ is also uniquely determined by F.

Then, we assume the formula F witnesses that the forcing condition X belongs to the set $\mathrm{Func}(n) \setminus r\mathrm{GEN}_1(k, C, n)$. We state this assumption more precisely in the following. We assume that F is a member of $rTAUT[X; C]$ and X_0 is the partial function defined as follows.

$$X_0 \equiv_{\mathrm{def.}} X \restriction (\mathrm{Str}(n) \setminus \mathrm{Str}(n, \delta, 1)). \tag{4.1}$$

The domain of X_0 is uniquely determined by F. By **Revised Dowd's Lemma 9**, there exists the minimal forcing condition S_1 such that $\mathrm{dom}(S_1) \subseteq \mathrm{Str}(n, \delta, 1)$ and $S_1 + X_0$ forces F. We have $S_1 \sqsubseteq X$ and S_1 is uniquely determined by F and X_0. The assumption of "F witnesses $X \in \mathrm{Func}(n) \setminus r\mathrm{GEN}_1(k, C, n)$ " means that we assume the following.

$$\ell(F)^k < \mathrm{Card}(\mathrm{dom}(S_1)). \tag{4.2}$$

Thus, in particular, we have $\ell(F) \leq 2^{n/k}$.

Now, X is uniquely determined by F, X_0 and by the following function X_1.

$$X_1 \equiv_{\mathrm{def.}} X \restriction (\mathrm{Str}(n, \delta, 1) \setminus \mathrm{dom}(S_1)). \tag{4.3}$$

Of course, the domain of X_1 is uniquely determined by F and X_0. Thus, letting $V(F)$ be the number of members of $\mathrm{Func}(n) \setminus r\mathrm{GEN}_1(k, C, n)$ witnessed by the particular formula F, we have the following.

$$V(F) \leq (\text{the number of } X_0\text{'s}) \cdot \sup(\text{the number of } X_1\text{'s}),$$

where the second factor of the right-hand side denotes the supremum (in fact, maximum) of the number of X_1's for various X_0's. Therefore, by the above formulas (1), (2) and (3), we have the following.

$$\log_2 V(F) \leq (2^n - 2^{n-t}) + (2^{n-t} - \ell(F)^k) = 2^n - \ell(F)^k. \tag{4.4}$$

For each natural number ℓ, let $W(\ell)$ be the number of formulas F such that its length is ℓ and we have to pay attention to F to count the cardinality of the set $\mathrm{Func}(n) \setminus r\mathrm{GEN}_1(k, C, n)$. If ℓ is sufficiently large, then we may assume $W(\ell) \leq (2\ell)^\ell$. Because, we array at most ℓ variables and at most ℓ other symbols (constants, logical connectives, etc.) to make a formula of length ℓ. Thus, we may assume the following:

$$\log_2 W(\ell) \leq \ell^2. \tag{4.5}$$

Let $N =_{\mathrm{def.}} 2^n$ and $L =_{\mathrm{def.}} \max\{C, r(n+1)+1\}$. If k and C are sufficiently large, then by the above formulas (4) and (5), we have the following.

$$\mathrm{Card}(\mathrm{Func}(n) \setminus r\mathrm{GEN}_1(k, C, n)) \leq \sum_{\ell} 2^{N - f(\ell)},$$

where ℓ varies from L to $2^{n/k}$ and $f(x) =_{\mathrm{def.}} x^k - x^2$. Then, the right-hand side is at most the following:

$$2^N \cdot \{2^{n - f(L)}\} \leq 2^N \cdot \{1/2^{f(L) - L}\} \leq 2^N \cdot (1/2^L) \leq 2^N \cdot (1/2^{n+2}).$$

Therefore, for sufficiently large natural numbers k and C, we have the following:

$$\mu(r\mathrm{GEN}_1(k, C)) > 0. \ \square$$

Proposition 4.2 *We have* $2\mathrm{GEN}_1 \subseteq 1\mathrm{GEN}_3$.

Proof: This is shown by adding dummy symbols. Suppose F is the following 1-query formula.

$$F \equiv_{\mathrm{def.}} \text{“}(a^{(1)} \Leftrightarrow \xi^n(q_1^{(1)}, \ldots, q_n^{(1)})) \Rightarrow H,\text{”}$$

where H is query free. Then, for any oracle A, the necessary and sufficient condition for "$F \in 1TAUT[A]$" is that the following formula belongs to $2TAUT[A]$, where we assume H has no occurrence of the variables $a^{(2)}$ and $q_j^{(2)}$ (for $j = 1, \cdots, n$).

$$F' \equiv_{\text{def.}} \text{``}\Big((a^{(1)} \Leftrightarrow \xi^{n+1}(0, q_1^{(1)}, \ldots, q_n^{(1)})) \wedge \\ (a^{(2)} \Leftrightarrow \xi^{n+1}(1, q_1^{(2)}, \ldots, q_n^{(2)}))\Big) \Rightarrow H.\text{''}$$

Then, we may consider F' to be a disentangled 2-query formula. Therefore, we have $2\mathrm{GEN}_1 \subseteq 1\mathrm{GEN}_3$. □

Proposition 4.3 (Implicitly shown in [13, p.71]) *Suppose that r is a positive integer. Then, we have $r\mathrm{GEN}_3 \cap (r+1)\mathrm{GEN}_1 \subseteq (r+1)\mathrm{GEN}_2$.*

Proof: The proof is a paraphrase of [13, p.71, line 6–13]. We show the **Proposition** by interpolating a formula of the form $\delta\triangle^{(n,1)}[S_1]$. Let A be an oracle such that $A \in r\mathrm{GEN}_3(k, C) \cap (r+1)\mathrm{GEN}_1(k, C)$, where k and C are positive integers. Let $F \equiv_{\text{def.}} \delta\langle r, n, H\rangle$, where we have $\delta = (t, B, f) \in \mathrm{DEM}(r, n)$ and H is a query free formula. Now, suppose $F \in (r+1)TAUT[A; C]$. Let G be the following formula.

$$G \equiv_{\text{def.}} \text{``}\Big(\delta(a^{(2)} \Leftrightarrow \xi^{n}(q_1^{(2)}, \ldots, q_n^{(2)})) \wedge \cdots \wedge \\ \delta(a^{(r+1)} \Leftrightarrow \xi^{n}(q_1^{(r+1)}, \ldots, q_n^{(r+1)}))\Big) \Rightarrow \delta H.\text{''}$$

Then, F is equivalent to $\delta(a^{(1)} \Leftrightarrow \xi^{n}(q_1^{(1)}, \ldots, q_n^{(1)})) \Rightarrow G$. Since A belongs to the class $(r+1)\mathrm{GEN}_1(k, C)$, there exists a forcing condition $S_1 \sqsubseteq A \upharpoonright \mathrm{Str}(n, \delta, 1)$ such that the size of the domain of S_1 is sufficiently small and A_0 forces the formula $(\delta\triangle^{(n,1)}[S_1]) \Rightarrow G$, where $A_0 =_{\text{def.}} A \upharpoonright (\mathrm{Str}(n) \setminus \mathrm{Str}(n, \delta, 1))$ (see **Revised Dowd's Lemma 9**). Although the size of A_0 is big, by the hypothesis of $A \in r\mathrm{GEN}_3(k, C)$ and by the fact that the formula $(\delta\triangle^{(n,1)}[S_1]) \Rightarrow G$ is a tautology with respect to A, there exists a forcing condition $T \sqsubseteq A$ such that T forces $(\delta\triangle^{(n,1)}[S_1]) \Rightarrow G$ and the size of the domain of T is sufficiently small. Recall that S_1 forces the following formula.

$$\delta(a^{(1)} \Leftrightarrow \xi^{n}(q_1^{(1)}, \ldots, q_n^{(1)})) \Rightarrow \delta\triangle^{(n,1)}[S_1].$$

Therefore, $S_1 \cup T$ forces F, and the size of the domain of $S_1 \cup T$ is sufficiently small. Hence, for sufficiently large k' and sufficiently large C', A belongs to $(r+1)\mathrm{GEN}_2(k', C')$. □

Proposition 4.4 *Suppose that r is a positive integer. Then, we have the following.*

$$r\mathrm{GEN}_3 \cap (r+1)\mathrm{GEN}_2 \subseteq (r+1)\mathrm{GEN}_3.$$

Proof: Assume that k and C are natural numbers and A is an oracle such that $A \in r\mathrm{GEN}_3(k, C) \cap (r+1)\mathrm{GEN}_2(k, C)$. Suppose that n is a positive integer and F is the following $(r+1)$-query formula.

$$F \equiv_{\text{def.}} \text{``}\Big((a^{(1)} \Leftrightarrow \xi^{n}(q_1^{(1)}, \ldots, q_n^{(1)})) \wedge \cdots \wedge \\ (a^{(r+1)} \Leftrightarrow \xi^{n}(q_1^{(r+1)}, \ldots, q_n^{(r+1)}))\Big) \Rightarrow H,\text{''}$$

where H is query free. Let $\ell =_{\text{def.}} \ell(F)$.

For each pair (i, j) of integers such that $1 \leq i < j \leq r+1$, let $G_{i,j}$ be the following formula obtained from F by substitution.

$$G_{i,j} \equiv_{\text{def.}} F[q_1^{(i)}/q_1^{(j)}] \cdots [q_n^{(i)}/q_n^{(j)}].$$

Each $G_{i,j}$ is essentially not an $(r+1)$-query formula of length ℓ but an r-query formula of shorter length. Now, for any oracle X, the necessary and sufficient condition for the assertion "$F \in TAUT[X]$" is the conjunction of the following two assertions; recall the argument in the last paragraph of the previous section.

- For every (i, j) such that $1 \leq i < j \leq r+1$, $G_{i,j}$ is a tautology with respect to X.
- For every $\delta \in \mathrm{DEM}(r, n)$, $\delta\langle r+1, n, H\rangle$ is a tautology with respect to X.

Thus, by our assumption of $A \in r\mathrm{GEN}_3(k, C) \cap (r+1)\mathrm{GEN}_2(k, C)$, if F is a tautology with respect to A and ℓ is sufficiently large, then F is forced by a finite portion $S \sqsubseteq A$ of the following size.

$$\mathrm{Card}(\mathrm{dom}(S)) \leq \binom{r+1}{2} \cdot \ell^k + \mathrm{Card}(\mathrm{DEM}(r+1, n)) \cdot \ell^k.$$

Since r is a fixed constant, it is easily verified that the right-hand side is bounded by a polynomial of ℓ and n. And, obviously we have $n \leq \ell$. Thus, the right-hand side is bounded by a polynomial of ℓ. Hence, for sufficiently large numbers k' and C', the oracle A belongs to $(r+1)\mathrm{GEN}_3(k', C')$. □

Theorem 4.5 *In Diagram 4.1, each vertical hierarchy collapses. That is, for each positive integer r, we have $(r+1)\mathrm{GEN}_1 = (r+1)\mathrm{GEN}_2 = (r+1)\mathrm{GEN}_3$. Thus, we get the relation of oracle classes illustrated by the following Diagram 4.2.*

Diagram 4.2.

$$\begin{array}{ccccccc}
 & & 2\mathrm{GEN}_1 & & 3\mathrm{GEN}_1 & & \\
 & & \| & & \| & & \\
 & & 2\mathrm{GEN}_2 & & 3\mathrm{GEN}_2 & & \\
 & & \| & & \| & & \\
1\mathrm{GEN}_3 & \supseteq & 2\mathrm{GEN}_3 & \supseteq & 3\mathrm{GEN}_3 & \supseteq & \cdots
\end{array}$$

Proof: By induction on r with **Propositions 4.2**, **4.3** and **4.4**, we have $(r+1)\mathrm{GEN}_1 \subseteq r\mathrm{GEN}_3$ and $(r+1)\mathrm{GEN}_1 = (r+1)\mathrm{GEN}_2 = (r+1)\mathrm{GEN}_3$, for each positive integer r. □

Dowd's Theorem 10 ([13, Theorem 10]) *For each positive integer r, we have $\mu(r\mathrm{GEN}_3) = 1$.*

Proof: Immediately from **Proposition 4.1** and **Theorem 4.5**. □

Chapter 5

Cohen-Feferman Generic Oracles

In this chapter, we study how existence of ceiling-generic oracles affects behavior of a generic oracle, by which we strengthen the well-known result that the following class of oracles is comeager: $\{X : P[X] \neq NP[X]\}$. The contents of this chapter is based on [28].

5.1 Separation of complexity classes

Recall that an oracle A is called ceiling-generic for a predicate $\varphi(X)(y)$ if the forcing complexity of $\varphi(X)(y)$ relative to A is bounded by a polynomial. And, recall that $\varphi[A]$ denotes the set of all bit strings u such that $\varphi(A)(u)$ holds. For these concepts, see **Definition 3.1**.

Theorem 5.1 *Suppose $\varphi(X)(y)$ and $\psi(X)(y)$ are finitely testable arithmetical predicates, G_1 is an oracle, and suppose that the following three hypothesisses hold for every oracle A such that $A \equiv G_1$ (mod. finite).*

(H. 1) A is c-generic for $\varphi(X)(y)$.

(H. 2) A is c-generic for $\neg\varphi(X)(y)$.

(H. 3) A is not c-generic for $\psi(X)(y)$.

Then, for every Cohen-Feferman generic oracle G_2, *we have*

$$\psi[G_2] \notin NP[\varphi[G_2]].$$

Proof: Suppose that $M[X]$ is a polynomial time-bounded nondeterministic oracle Turing machine, and suppose that S_0 is an arbitrary condition. We shall show existence of a condition T such that T is an extension of S_0, and T forces $\psi[X] \neq \mathrm{Lang}(M[\varphi[X]])$. Let A be an oracle such that $A \equiv G_1$ (mod. finite) and A is an extension of S_0 (i.e. $S_0 \sqsubseteq A$). Assume that p is a polynomial such that A is p-c-generic for $\varphi(X)(y)$ and A is p-c-generic for $\neg\varphi(X)(y)$. By the hypothesisses (H. 1) and (H. 2), such a p surely exists. Let t be a polynomial that is a time-bounding function of $M[X]$. We may assume $n \leq t(n) < t(n+1)$, for all natural numbers n, and may assume that the same thing holds with p in place of t. Let us define a polynomial q as follows.

$$q(x) = t(x) \cdot p(t(x)) + \mathrm{Card}(\mathrm{dom}(S_0)).$$

By the hypothesis (H.3), A is not q-c-generic for $\psi(X)(y)$. Moreover, the predicate $\psi(X)(y)$ is finitely testable. Hence, there exists a bit string u for which the following holds: "$\psi(A)(u)$ is true, and for each condition S such that $S \sqsubseteq A$ and $\mathrm{Card}(\mathrm{dom}(S)) \leq q(|u|)$, there exists a condition T such that $S \sqsubseteq T$ and T forces $\neg\psi(X)(u)$." We fix such a u.

In the case where $M[\varphi[A]]$ accepts u. We consider a fixed accepting computation of M. Since in course of the computation M asks at most $t(|u|)$ questions of size at most $t(|u|)$ to the oracle, there exists a condition S_1 such that $S_1 \sqsubseteq A$, $\mathrm{Card}(\mathrm{dom}(S_1)) \leq t(|u|) \cdot p(t(|u|))$ and S_1 forces that $M[\varphi[X]]$ accepts the bit string u. Since S_0 and S_1 are compatible, there exists a condition S_2 such that S_2 is a common extension of them (i.e. $S_0 \sqsubseteq S_2$ and $S_1 \sqsubseteq S_2$) and $\mathrm{Card}(\mathrm{dom}(S_2)) \leq q(|u|)$. Hence, by our choice of the bit string u, there exists a condition T such that $S_2 \sqsubseteq T$ and T forces $\neg\psi(X)(u)$. We fix such a T.

Otherwise. We consider the arithmetical predicate $\psi_0(X)(y)$ defined by the following assertion: "$\psi(X)(y)$ is true, and $M[\varphi[X]]$ rejects y." Since the predicate $\psi_0(X)(y)$ is finitely testable and $\psi_0(A)(u)$ is true, there exists a condition $S_3 \sqsubseteq A$ such that S_3 forces $\psi_0(X)(u)$. Let T be a common extension of S_0 and S_3.

In either case, $S_0 \sqsubseteq T$, and T forces $\psi[X] \neq \mathrm{Lang}(M[\varphi[X]])$. □

Corollary 5.2 *Suppose that r is a positive integer. Then, the following class of oracles is comeager:*

$$\{X : coNP[X] \not\subseteq NP[rTAUT[X]]\}.$$

Proof: Note that one counter-example is sufficient to refute a tautology. Thus, *any* oracle is p-c-generic for $\neg rTAUT(X)(y)$ with $p(n) =_{\mathrm{def.}} r$ (for each $n \in \mathbb{N}$). Let G_1 be an r-generic oracle in Dowd's sense such that G_1 is not t-generic. By **Dowd's Lemma 7**, **Dowd's Theorem 10** and **Fact 1.1** (see Chapter 1 of this thesis), we know that such a G_1 surely exists, and that the following triple satisfies the three hypothesis (H. 1), (H. 2) and (H. 3):

$$(rTAUT(X)(y), TAUT(X)(y), G_1).$$

Hence, by **Theorem 5.1**, for each Cohen-Feferman generic oracle G_2, we have $TAUT[G_2] \notin NP[rTAUT[G_2]]$. □

Remark: Since $NPQUERY[A] = NP[QBF \oplus A]$ for any oracle A, it is easily seen that the statements of **Theorem 5.1** and **Corollary 5.2** hold with $NPQUERY[]$ in place of $NP[]$. □

5.2 Examples

Let $L_{\rm BGS}[X]$ be the oracle-dependent tally set defined as follows.

$$L_{\rm BGS}[X] = \{0^n : \neg\exists y \in A(|y| = n)\}.$$

It is well-known that Baker et al. used (the complement of) the above tally set in [3] to show existence of an oracle A such that $P[A] \neq NP[A]$. Later, some people interpreted the method of Baker et al. as a forcing method, and they showed that $P[G_2] \neq NP[G_2]$ for every Cohen-Feferman generic oracle G_2 (e.g. [12, 13], [7]). However, the polynomial time many-one degree of the tally set $L_{\rm BGS}[X]$ is so low that $L_{\rm BGS}[X]$ is useless to separate $TAUT[X]$ from $rTAUT[X]$ i.e. useless to show **Corollary 5.2**. To see this, let us prove an example by using $L_{\rm BGS}[X]$. Suppose that G_2 is a Cohen-Feferman generic oracle. Then, the following holds:

(4.1) $\quad 1TAUT[G_2] \notin NP[QBF \oplus G_2].$

Moreover, as a special case of (4.1), we have the following:

(4.2) $\quad 1TAUT[G_2] \notin P[TAUT \oplus G_2].$

A proof of (4.1) by using $L_{\rm BGS}[X]$ is as follows. Suppose that $M[X]$ is a polynomial time-bounded nondeterministic oracle Turing machine. Let D_M be the set of all conditions that force the following assertion (4.3).

(4.3) $\quad L_{\rm BGS}[X] \neq {\rm Lang}(M[QBF \oplus X])$

By the method of the proofs of Theorem 3 and 4 of [3], it is verified that D_M is dense, and hence every Cohen-Feferman generic oracle X satisfies (4.3). Therefore, for each Cohen-Feferman generic oracle G_2, we have $L_{\rm BGS}[G_2] \notin NP[QBF \oplus G_2]$. Since the above tally set $L_{\rm BGS}[A]$ is polynomial time many-one reducible to $1TAUT[A]$ for each oracle A, we have (4.1).

Of course, we can show (4.1) without using $L_{\rm BGS}[X]$. First, note the following.

Proposition 5.3 *Suppose $\psi(X)(y)$ is a finitely testable arithmetical predicate and G_1 is an oracle. And, suppose that the hypothesis (H. 3) holds for every oracle A such that $A \equiv G_1$ (mod. finite). Then, for every Cohen-Feferman generic oracle G_2, we have*

$$\psi[G_2] \notin NP[QBF \oplus G_2].$$

Proof: We consider the predicate $\varphi(X)(y)$ defined by "$y \in X$." Clearly, any oracle is c-generic for $\varphi(X)(y)$ and c-generic for $\neg\varphi(X)(y)$. And, for any oracle A, the language $\varphi[A]$ is just A itself. Hence, by **Theorem 5.1** and **Remark** after the proof of **Corollary 5.2**, we have **Proposition 5.3**. □

Let $\mathcal{F}$ be the class of all oracles which are *not* 1-generic in Dowd's sense. By **Fact 1.1**, $\mathcal{F}$ is comeager in the Cantor space, and is closed under finite changes; indeed, $\mathcal{F}$ contains all Cohen-Feferman generic oracles [13, Theorem 12]. Take an oracle $G_1 \in \mathcal{F}$, and let $\psi(X)(y)$ be the predicate $1TAUT(X)(y)$. Then, we get (4.1) by **Proposition 5.3**.

By the way, in the statement of **Theorem 5.1**, it is essential that the three hypothesisses (H. 1), (H. 2) and (H. 3) hold not only for $A = G_1$ but also for any A such that $A \equiv G_1$ (mod. finite). Compare **Proposition 5.3** with the following **Example**.

Example 5.1 There exists a pair $(\psi_0(X)(y), G_1)$ that satisfies all of the following three requirements.

1. $\psi_0(X)(y)$ is a finitely testable arithmetical predicate and G_1 is an oracle.

2. G_1 is not c-generic for $\psi_0(X)(y)$.

3. For each Cohen-Feferman generic oracle G_2, we have $\psi_0[G_2] \in P[G_2]$.

Proof: For each positive integer i and for each query free formula H, we denote the following 1-query formula by $\natural\langle 1, i, H\rangle$:

$$(a \Leftrightarrow \xi^i(q_1, \ldots, q_i)) \Rightarrow H.$$

Let G_1 be a 1-generic oracle in Dowd's sense with respect to a polynomial p. We may assume $n \leq p(n) \leq p(n{+}1)$, for all natural numbers n. We consider the arithmetical predicate $\psi_0(X)(y)$ defined by the following assertion: "for some $n \in \mathbb{N}$, $y = 0^n$, and for each $i \leq n$ and for each query free formula H, if $\natural\langle 1, i, H\rangle$ is a tautology with respect to X then there exists a condition $S \sqsubseteq X$ such that Card(dom(S)) is at most $p(|\natural\langle 1, i, H\rangle|)$ and S forces $\natural\langle 1, i, H\rangle \in TAUT[X]$."

We show that G_1 is not c-generic for $\psi_0(X)(y)$. Assume for a contradiction that G_1 is q-c-generic for $\psi_0(X)(y)$, where q is a polynomial. We may assume $n \leq q(n) \leq q(n+1)$, for all natural numbers n. Let c be a sufficiently large natural number and let m be a natural number satisfying the following inequality:

$$c \cdot p(c \cdot q(m)^c + c) < 2^m - 1. \tag{4.4}$$

Since G_1 is 1-generic in Dowd's sense with respect to the polynomial p, we have $\psi_0(G_1)(0^m)$. Therefore, by our assumption for a contradiction, there exists a condition $S \sqsubseteq G_1$ such that Card(dom(S)) is at most $q(m)$ and S forces $\psi_0(X)(0^m)$. Let $\{v^{(1)}, \ldots, v^{(d)}\}$ be an enumeration of all bit strings v such that $v \in \mathrm{dom}(S)$ and $S(v) = 1$. Of course, we have the following:

$$d \leq q(m).$$

Let H_0 be a query free formula such that for each oracle X, the 1-query formula $\natural\langle 1, m, H_0\rangle$ is a tautology with respect to X if and only if the following assertion holds:

$$(\forall u \in X \cap \{0,1\}^{\leq m-1})\,(u = v^{(1)} \text{ or } \ldots \text{ or } u = v^{(d)}). \tag{4.5}$$

We choose H_0 so that its length $|H_0|$ would be as short as possible. We define an oracle A as follows: $S \sqsubseteq A$, and $A(u) = 0$ for all $u \notin \mathrm{dom}(S)$. Then, we have $\natural\langle 1, m, H_0\rangle \in 1TAUT[A]$. On the other hand, $\psi_0(A)(0^m)$ holds, since this predicate is forced by S. Hence, by our definition of $\psi_0(X)(y)$, there exists a condition $T \sqsubseteq A$ such that Card(dom(T)) is at most $p(|\natural\langle 1, m, H_0\rangle|)$ and T forces $\natural\langle 1, m, H_0\rangle \in TAUT[X]$. Thus, T forces the assertion (4.5). However, by the inequality (4.4) and by our choice of the formula H_0, we may assume Card(dom(T)) $< (2^m - 1)/c$. Recall that the cardinality of $\{0,1\}^{\leq m-1}$ is $2^m - 1$. Hence, there exists an oracle X such that the assertion (4.5) fails but $T \sqsubseteq X$, a contradiction.

Finally, let G_2 be a Cohen-Feferman generic oracle; let us show $\psi_0[G_2] \in P[G_2]$. Then, G_2 is not a 1-generic oracle in Dowd's sense [13, Theorem 12]. Therefore, $\psi_0[G_2]$ is a finite set. □

Chapter 6

One-query versus One-question

In this chapter, we investigate, among others, whether for a random oracle A, $1TAUT[A]$ is one-question truth-table reducible to A. We present negative answer, i.e. it is not one-question truth-table reducible to A with probability one.

6.1 Reducibilities

In this chapter and in the next chapter, for several kinds of reducibilities, we consider the following assertion.

(†) "If A is a random oracle then $1TAUT[A]$ is not reducible to A with probability one."

Disjunctive reducibility ("d-reducibility" for short) and one-question truth-table reducibility ("1-tt-reducibility" for short) are well-known concepts locating between many-one reducibility and truth-table reducibility ("tt-reducibility" for short). An oracle A is called d-reducible (1-tt-reducible, respectively) to an oracle B if it is tt-reducible to B and every truth-table condition used in the reduction is a disjunctive formula (every truth-table condition used in the reduction has norm 1). For more formal treatment, see Odifreddi [25, p.268]. It is also well-known that for any oracle A, A is 1-tt-reducible to the complement of A. Some authors called disjunctive reducibility by different name (Rogers [27, p.123]). Polynomial time versions of tt-reducibility, d-reducibility and 1-tt-reducibility have been studied since 1970's (Ladner et al. [22]). In this chapter, we show the following.

Theorem 6.1 *1. If A is a random oracle, then $1TAUT[A]$ is not disjunctive reducible to A with probability one.*

2. If A is a random oracle, then $1TAUT[A]$ is not disjunctive reducible to the complement of A with probability one.

3. If A is a random oracle, then $1TAUT[A]$ is not 1-question truth-table reducible to A with probability one.

Convention for this chapter: Nbhd(S). For each forcing condition S, we define a class Nbhd(S) of oracles as follows. Nbhd(S) $=_{\text{def.}} \{X \in \mathcal{C} : S \sqsubseteq X\}$. "Nbhd" is an abbreviation of "neighborhood."

6.2 Initial segments of oracles

To show **Theorem 6.1**, we introduce an auxiliary language $IniSeg[X]$ in this section. For each oracle X, the language $IniSeg[X]$ consists of all initial segments of the oracle. For technical reasons, we adopt the following definition.

Definition 6.1 *1. For each bit string u of length at least 1, we define a forcing condition S_u as follows. Letting $u = u_1u_2\cdots u_{n+1}$,* dom(S_u)

is defined as $\{z(0), z(1), \dots, z(n)\}$, and $S_u(z(j))$ is defined as u_{j+1} (for $j = 0, 1, \dots, n$). For the empty bit string λ, we define a forcing condition S_λ as to be the empty function.

2. We define an oracle-dependent language $IniSeg[X]$ as follows.

$$IniSeg[X] =_{\text{def.}} \{u \in \{0,1\}^* : S_u \sqsubseteq X\}.$$

"IniSeg" is an abbreviation of "initial segment." □

Observe that $IniSeg[X]$ is uniformly (polynomial time) one-one reducible to $1TAUT[X]$ in the following sense. Recall that Str(k) is the following set of bit strings; Str(k) $= \{z(m) : 0 \leq m \leq 2^k - 1\}$. For each bit string u of length at least 1, we define a relativized formula F_u as follows.

$$F_u \equiv_{\text{def.}} \text{“}(a^{(1)} \Leftrightarrow \xi^k(q_1^{(1)}, \dots, q_k^{(1)})) \Rightarrow \triangle^{(k,1)}[S_u],\text{”}$$

where k is the least natural number such that dom(S_u) $\subseteq$ Str(k). For the symbol of the form $\triangle^{(n,i)}[S]$, see the last paragraph of Chapter 2 of this thesis. For the empty bit string λ, we define F_λ as to be "$(a^{(1)} \Leftrightarrow \xi^1(q_1^{(1)})) \Rightarrow (1 \Leftrightarrow 1)$" i.e. a trivial 1-query tautology. Thus, for any oracle X and for any bit string u, the necessary and sufficient condition for "$S_u \sqsubseteq X$" is "$F_u \in 1TAUT[X]$."

Thus, to prove **Theorem 6.1**, it is sufficient to show the following.

Lemma 6.2 *1. If A is a random oracle, then $IniSeg[A]$ is not disjunctive reducible to A with probability one.*

2. If A is a random oracle, then $IniSeg[A]$ is not disjunctive reducible to the complement of A with probability one.

3. If A is a random oracle, then $IniSeg[A]$ is not 1-question truth-table reducible to A with probability one.

6.3 Disjunctive reducibility

In this section, we show the assertions 1 and 2 of **Lemma 6.2**. Note that a language A is disjunctive reducible to a language B if and only if there exists a recursive mapping g satisfying the following three requirements.

(d:1) The domain of g is $\{0,1\}^*$.

(d:2) For each bit string u, $g(u)$ is (the binary representation of) a finite set of bit strings.

(d:3) For each bit string u, the necessary and sufficient condition for "$u \in A$" is "$g(u) \cap B$ is not empty."

Now, suppose B is a random oracle. Consider the situation where g is a recursive mapping satisfying the above requirements (d:1), (d:2) and (d:3) with $IniSeg[B]$ in place of A. Then, g may be not one-to-one. Thus, in the following **Definition**, we introduce a mapping $\mathrm{Core}[g]$ approximating g. For each bit string u, $\mathrm{Core}[g](u)$ is defined as a subset of $g(u)$ so that if bit strings u and v have the same binary length and we have $u \neq v$ then $\mathrm{Core}[g](u)$ and $\mathrm{Core}[g](v)$ are disjoint.

Definition 6.2 *Let g be a mapping satisfying the requirements (d:1) and (d:2). Assume $i \in \{0,1\}$. We use the following definitions 1 – 5 in this section only.*

1. *We define a mapping $\mathrm{Core}[g]$ as follows. $\mathrm{dom}(\mathrm{Core}[g]) =_{\mathrm{def.}} \{0,1\}^*$. For each bit string u, letting $n = |u|$,*

$$\mathrm{Core}[g](u) =_{\mathrm{def.}} \{w \in g(u) : \forall v \in \{0,1\}^n \setminus \{u\},\ w \notin g(v)\}.$$

2. *We define a mapping $\mathrm{AntiCore}[g]$ as follows. $\mathrm{dom}(\mathrm{AntiCore}[g]) =_{\mathrm{def.}} \{0,1\}^*$. For each bit string u, letting $n = |u|$,*

$$\mathrm{AntiCore}[g](u) =_{\mathrm{def.}} \bigcup\{\mathrm{Core}[g](v) : v \in \{0,1\}^n \setminus \{u\}\}.$$

3. *We define a mapping $\mathrm{Supp}[g]$ on natural numbers as follows. For each natural number n,*

$$\mathrm{Supp}[g](n) =_{\mathrm{def.}} \{u \in \{0,1\}^n : \mathrm{Core}[g](u) \text{ is non-empty}\}.$$

"Supp" is an abbreviation of "support." For each natural number n, we denote the cardinality of $\mathrm{Supp}[g](n)$ by $c(g,n)$.

$$c(g,n) =_{\mathrm{def.}} \mathrm{Card}(\mathrm{Supp}[g](n)).$$

4. *For each bit string u, we define a forcing condition $T_{(g,u,i)}$ as follows. $\mathrm{dom}(T_{(g,u,i)}) =_{\mathrm{def.}} \mathrm{AntiCore}[g](u)$, and for each $w \in \mathrm{AntiCore}[g](u)$, $T_{(g,u,i)}(w) =_{\mathrm{def.}} \neg i$, where $\neg$ is the operator of complement on the Boolean algebra $\{0,1\}$ i.e. $\neg 1 = 0$ and $\neg 0 = 1$. When the mapping g and the Boolean value i are clear from the context, we denote $T_{(g,u,i)}$ by T_u.*

5. *We define d-Red$(g,1)$ (d-Red$(g,0)$, respectively) as to be the collection of all oracles X such that the language $IniSeg[X]$ is disjunctive reducible to X (to the complement of X) via the mapping g. That is, d-Red$(g,i) =_{\mathrm{def.}} \{X \in \mathcal{C}$: For each bit string u, the necessary and sufficient condition for "$S_u \sqsubseteq X$" is "$\exists w \in g(u)$ such that $X(w) = i$" $\}$. "d-Red" is an abbreviation of "disjunctive reducible."* □

Lemma 6.3 *Let g be a mapping satisfying the requirements (d:1) and (d:2). Assume $i \in \{0,1\}$. Assume that n is a positive integer. Then, each of the following assertions 1 – 5 holds.*

1.
$$d\text{-Red}(g,i) \subseteq$$
$$\{X \in \mathcal{C} : X(z(0))X(z(1))\cdots X(z(n-1)) \in \mathrm{Supp}[g](n)\}.$$

2. $d\text{-Red}(g,i) \subseteq \bigcup\{d\text{-Red}(g,i) \cap \mathrm{Nbhd}(S_u) : u \in \mathrm{Supp}[g](n)\}$.

3. *We have* $\mu(d\text{-Red}(g,i)) \leq c(g,n)/2^n$, *where* μ *denotes Lebesgue measure.*

4. $d\text{-Red}(g,i) \subseteq \bigcup\{\mathrm{Nbhd}(T_u) : u \in \mathrm{Supp}[g](n)\}$.

5. *Assume* $c(g,n) \geq 2$. *Then, we have*
$$\mu(d\text{-Red}(g,i)) \leq c(g,n)/2^{c(g,n)-1}.$$

Proof:

Proof of 1 and 2: Suppose that X is a member of the set d-Red(g,i). Let u be the bit string $X(z(0))X(z(1))\cdots X(z(n-1))$. Then, $X \in \mathrm{Nbhd}(S_u)$. Hence, there exists a bit string $w \in g(u)$ such that $X(w) = i$; in the following, we verify $w \in \mathrm{Core}[g](u)$. Suppose $v \in \{0,1\}^n \setminus \{u\}$. Then, X is not an extension of S_v. Therefore, by our assumption of $X \in d$-Red(g,i), w does not belong to the set $g(v)$. Hence, we have $w \in \mathrm{Core}[g](u)$. Therefore, we have $u \in \mathrm{Supp}[g](n)$. Hence, the assertions 1 and 2 of **Lemma 6.3** hold.

Proof of 3: This is a direct corollary of the assertion 1 of **Lemma 6.3**.

Proof of 4: Suppose $u \in \{0,1\}^n$. We show the following relation.

$$(4') \qquad d\text{-Red}(g,i) \cap \mathrm{Nbhd}(S_u) \subseteq \mathrm{Nbhd}(T_u).$$

Then, the conjunction of the above $(4')$ and the assertion 2 of **Lemma 6.3** implies the assertion 4 of **Lemma 6.3**. To show the relation $(4')$, assume $X \in d$-Red$(g,i) \cap \mathrm{Nbhd}(S_u)$. Assume for a contradiction that the oracle X is not an extension of the forcing condition T_u. That is, there exists a bit string $w \in \mathrm{AntiCore}[g](u)$ (recall that $\mathrm{AntiCore}[g](u)$ is the domain of the forcing condition T_u) such that $X(w) = i$ (recall $T_u(w) = \neg i$). Then, by the definition of the mapping AntiCore$[g]$, there exists a bit string $v \in \{0,1\}^n \setminus \{u\}$ such that $w \in \mathrm{Core}[g](v) \subseteq g(v)$. Therefore, by our assumption of $X \in d$-Red(g,i) and by the fact that $X(w)$ equals i, we have $S_v \sqsubseteq X$. This contradicts our assumption of $S_u \sqsubseteq X$. Hence, the relation $(4')$ holds.

Proof of 5: Suppose $u \in \{0,1\}^n$. We define a family $\mathcal{F}_u$ of sets as follows.

$$\mathcal{F}_u =_{\text{def.}} \{\mathrm{Core}[g](v) : v \in \mathrm{Supp}[g](n) \setminus \{u\}\}.$$

Each element of $\mathcal{F}_u$ is a non-empty set. Moreover, it is easily verified that the members of $\mathcal{F}_u$ are mutually disjoint: that is, if $v \neq w$ then $\mathrm{Core}[g](v) \cap \mathrm{Core}[g](w) = \emptyset$. Recall the following relation.

$$\bigcup \mathcal{F}_u = \mathrm{AntiCore}[g](u) = \mathrm{dom}(T_u).$$

Therefore, the domain of T_u has cardinality at least $c(g,n) - 1$, where $c(g,n) = \mathrm{Card}(\mathrm{Supp}[g](n))$. Thus, we have the following.

$$\mu(\mathrm{Nbhd}(T_u)) \leq 1/2^{c(g,n)-1}.$$

Since u was arbitrary, by 4 of **Lemma** 6.3, we have the following.

$$\mu(d\text{-Red}(g,i)) \leq c(g,n)/2^{c(g,n)-1}.\square$$

Proof of 1 and 2 of **Lemma 6.2**: Let g be a mapping satisfying the requirements (d:1) and (d:2). Assume we have $i \in \{0,1\}$. We show $\mu(d\text{-Red}(g,i)) = 0$. **Case 1**: The case where there exist infinitely many natural numbers n such that $c(g,n) < n$. In this case, by 3 of **Lemma 6.3**, we have $\mu(d\text{-Red}(g,i)) = 0$. **Case 2**: Otherwise. Then, there exists a natural number n_0 such that for all natural numbers n greater than n_0, we have $c(g,n) \geq n$. In this case, by 5 of **Lemma 6.3**, we have $\mu(d\text{-Red}(g,i)) = 0$.

However, the following relation holds.

$$\{X \in \mathcal{C} : \mathit{IniSeg}[X] \text{ is disjunctive reducible to } X\}$$
$$= \bigcup\{d\text{-Red}(g,1) : g \text{ is a recursive mapping satisfying the requirements (d:1) and (d:2)}\}.$$

Hence, the above class of oracles has Lebesgue measure zero. In other words, if A is a random oracle, then $\mathit{IniSeg}[A]$ is not disjunctive reducible to A with probability one. Similarly, if A is a random oracle, then $\mathit{IniSeg}[A]$ is not disjunctive reducible to the complement of A with probability one. □

6.4 One-question truth table reducibility

In this section, we show the assertion 3 of **Lemma 6.2**. Note that a language A is 1-tt-reducible to a language B if and only if there exists an ordered pair of two recursive mappings g_1 and g_2 satisfying the following four requirements.

(1-tt:1) The domain of g_i is $\{0,1\}^*$ (for $i = 1, 2$).

(1-tt:2) For each bit string u, $g_1(u)$ is a bit string.

(1-tt:3) For each bit string u, $g_2(u)$ is either 0 or 1.

(1-tt:4) For each bit string u, the necessary and sufficient condition for "$u \in A$" is "$B(g_1(u)) = g_2(u)$."

Definition 6.3 *Let $g = (g_1, g_2)$ be a pair of mappings such that the above requirements (1-tt:1), (1-tt:2) and (1-tt:3) are satisfied. The following definitions 1 and 2 are used in this section only.*

1. *We define a mapping $\mathrm{Supp}[g]$ on natural numbers as follows. For each natural number n,*

$$\mathrm{Supp}[g](n) =_{\text{def.}} \{u \in \{0,1\}^n : \textit{For all } v \in \{0,1\}^n \setminus \{u\}, \textit{ the ordered pairs } (g_1(u), g_2(u)) \textit{ and } (g_1(v), g_2(v)) \textit{ are not identical}\}.$$

For each natural number n, we denote the cardinality of $\mathrm{Supp}[g](n)$ by $c(g,n)$.

2. *We define $1\text{-}tt\text{-Red}(g)$ as to be the collection of all oracles X such that the language $\mathit{IniSeg}[X]$ is 1-tt-reducible to X via g. That is,*

$$1\text{-}tt\text{-Red}(g) =_{\text{def.}} \{X \in \mathcal{C} : \textit{For each bit string } u, \textit{ the necessary and sufficient condition for } \text{“}S_u \sqsubseteq X\text{”} \textit{ is } \text{“}X(g_1(u)) = g_2(u)\text{”}\}.$$

"$1\text{-}tt\text{-Red}$" is an abbreviation of "1-tt-reducible." □

Lemma 6.4 *Let $g = (g_1, g_2)$ be an ordered pair of mappings satisfying the requirements (1-tt:1), (1-tt:2) and (1-tt:3). Assume that n is a positive integer. Then, each of the following assertions 1 – 4 holds.*

1.

$$1\text{-}tt\text{-Red}(g,i) \subseteq \{X \in \mathcal{C} : X(z(0))X(z(1))\cdots X(z(n-1)) \in \mathrm{Supp}[g](n)\}.$$

2. $1\text{-}tt\text{-Red}(g,i) \subseteq \bigcup\{1\text{-}tt\text{-Red}(g,i) \cap \mathrm{Nbhd}(S_u) : u \in \mathrm{Supp}[g](n)\}.$

3. *We have* $\mu(1\text{-}tt\text{-Red}(g,i)) \leq c(g,n)/2^n$.

4. For each real number x, we let $\lceil x\rceil$ denote the least natural number m such that $m \geq x$. Assume $c(g,n) \geq 3$. Then, we have

$$\mu(1\text{-}tt\text{-}\mathrm{Red}(g,i)) \leq c(g,n)/2^{\lceil c(g,n)/2\rceil -1}.$$

Proof: The assertion 1, 2, 3 are shown similarly to corresponding assertions of **Lemma 6.3**. To show the assertion 4 of **Lemma 6.4**, we introduce the following equivalence relation "$\sim$" on $\mathrm{Supp}[g](n)$: $u \sim v$ if and only if $g_1(u) = g_1(v)$. Now, fix an arbitrary bit string $u \in \{0,1\}^n$. Let $\mathcal{F}_u$ be the family of all bit strings $v \in \mathrm{Supp}[g](n)$ satisfying the following two requirements.

- v is not equivalent to u with respect to the relation $\sim$.
- In the equivalence class of v with respect to the relation $\sim$, v is the least element with respect to lexicographic order.

We define a forcing condition T'_u as follows.

$$\mathrm{dom}(T'_u) =_{\mathrm{def.}} \{g_1(v) : v \in \mathcal{F}_u\},$$
$$T'_u(g_1(v)) =_{\mathrm{def.}} \neg g_2(v).$$

By the first requirement for $\mathcal{F}_u$, each bit string $v \in \mathcal{F}_u$ is not u. Hence, if X is an oracle such that $X \in 1\text{-}tt\text{-}\mathrm{Red}(g) \cap \mathrm{Nbhd}(S_u)$, then for every $v \in \mathcal{F}_u$, we have $X(g_1(v)) \neq g_2(v)$. Thus, the following holds.

(a) $$1\text{-}tt\text{-}\mathrm{Red}(g) \cap \mathrm{Nbhd}(S_u) \subseteq \mathrm{Nbhd}(T'_u).$$

Note that, for each bit string $v \in \mathrm{Supp}[g](n)$, the cardinality of the equivalence class $v/\sim$ is at most 2. Therefore, the number $c(g,n)$ (i.e. the cardinality of $\mathrm{Supp}[g](n)$) is at most $2 \times \mathrm{Card}(\mathcal{F}_u) + 2$. In other words, we have the following.

$$\mathrm{Card}(\mathcal{F}_u) \geq \lceil c(g,n)/2\rceil - 1.$$

Moreover, the mapping g_1 is one-to-one on $\mathcal{F}_u$. Hence, we have the following.

(b) $$\mu(\mathrm{Nbhd}(T'_u)) \leq 1/2^{\lceil c(g,n)/2\rceil -1}.$$

Therefore, by the above (a), (b) and by the assertion 2 of **Lemma 6.4**, we get the following inequality.

$$\mu(1\text{-}tt\text{-}\mathrm{Red}(g,i)) \leq c(g,n)/2^{\lceil c(g,n)/2\rceil -1}. \quad \square$$

Proof of 3 of **Lemma 6.2***:* We use **Lemma 6.4**. The reminder of the proof is similar to *Proof of 1 and 2 of* **Lemma 6.2**. □

Proof of **Theorem 6.1***:* Immediate from **Lemma 6.2** and from the fact that $IniSeg[X]$ is uniformly one-one reducible to $1TAUT[X]$. □

Chapter 7

Control of While-loops

In this chapter, we investigate the relationship between the r-query tautologies and fundamental questions in the theory of computational complexity. For this purpose, we construct a deterministic algorithm whose while-loop's execution time is controlled by a forcing method. The contents of this chapter is based on [29].

7.1 Problem of this chapter

In [6], Bennet and Gill showed that if A is a random oracle then $P[A] \neq NP[A]$ with probability 1. Since $TAUT[A]$ is a $coNP[A]$-complete set for an arbitrary oracle A, we obtain the following as its direct corollary.

Fact 7.1 *If A is a random oracle then we have $TAUT[A] \notin P[A]$ with probability one.*

The above statement means that the set $\{X : TAUT[X] \notin P[X]\}$ has Lebesgue measure 1 in the Cantor space. We consider the problem whether the statement of the above fact remains true when we substitute $rTAUT[A]$ for $TAUT[A]$. Extending Dowd's theory of r-generic oracles [13], we shall present a forcing argument to bound execution time of a while-loop of a deterministic oracle Turing machine, from which we shall show the following theorem, where $TAUT$ denotes the collection of all tautologies of the usual propositional calculus.

Theorem 7.1 *Suppose that r is a positive integer. Then, for each A that is an r-generic oracle in Dowd's sense, we have the following.*

$$rTAUT[A] \equiv_T^P TAUT \oplus A. \tag{1.1}$$

When A is a Feferman generic oracle, as is shown in Chapter 5, the formula (1.1) does not hold. Whereas, we do have the above formula (1.1) for A that is an r-generic oracle in the sense of Dowd.

Our conclusion in this chapter is the following theorem.

Theorem 7.2 *Suppose that r is a positive integer. Then, the following two statements are equivalent.*

(1) If A is a random oracle then $rTAUT[A] \notin P[A]$ with probability 1.

(2) The unrelativized class R does not equal NP.

We shall prove **Theorem 7.2** as a corollary of **Theorem 7.1**. However, to explain our motive for proofs, we first show **Theorem 7.1** in the case where $r = 1$ separately.

7.2 Algorithm for 1-query tautologies

In this section, we give a proof of **Theorem 7.1** in the case where $r = 1$. In this special case, each tautology F has the unique minimal forcing condition that forces F. Let us recall the following two results by Dowd.

Dowd's Lemma 9 ([13, Lemma 9]) *If F is a 1-query formula which is a tautology with respect to some X then there is a unique minimal set S of queries which forces F to be a tautology (we say F specifies S).*

Dowd's Theorem 11 ([13, Theorem 11]) *If $NP = coNP$ then there is a nondeterministic oracle machine M uniformly accepting $TAUT[X]$, such that with respect to any r-generic X (in the sense of Dowd), M accepts the members of $rTAUT[X]$ in polynomial time.*

We shall show an analogue of **Dowd's Theorem 11** for a deterministic oracle machine:

Lemma 7.3 *There exists a deterministic oracle Turing machine $M[X]$ for which (1) and (2) below hold.*

(1) For every oracle A, $\mathrm{Lang}(M[TAUT \oplus A]) = 1TAUT[A]$.

(2) If A is 1-generic in the sense of Dowd, then $M[TAUT \oplus A]$ accepts the members of $1TAUT[A]$ in polynomial time.

In the proof of **Dowd's Theorem 11**, Dowd used a forcing argument to bound the lengths of bit strings written on a query tape by a nondeterministic oracle machine. However, we want to bound the computing time of a deterministic oracle machine. To show the above **Lemma 7.3**, we shall carefully examine the computational complexity of a function which maps each 1-query formula F that is a tautology with respect to a given oracle X to the unique minimal forcing condition S such that F specifies S. We shall construct a transducer that has one while-loop so that, for each 1-query formula F, it computes a candidate for (the domain of) the specified forcing condition. The transducer computes (the domain of) the specified function S itself as long as input F is a tautology with respect to a given oracle. A forcing argument will put the bounds of the execution time of the while-loop. Finally, we shall check whether the candidate for the specified forcing condition really forces that F is a tautology.

Proof of Lemma 7.3: Let us define four predicates as follows.

$TAUTALL(F)$

$\equiv_{def.}$ F is a formula of the relativized propositional calculus and

F is a tautology with respect to any oracle.

$1QFORMULA(i, H, F)$

$\equiv_{def.}$ i is a positive natural number, H is a query free formula and

F is the 1-query formula "$(a \Leftrightarrow \xi^i(q_1, \ldots, q_i)) \Rightarrow H$."

$CRITICAL(u, i, H, F)$

$\equiv_{def.}$ $|u| = i$, $1QFORMULA(i, H, F)$ is true and

the following formula is not a tautology :

"$(\bigwedge_{j=1}^{i}(q_j \Leftrightarrow u_j)) \Rightarrow H$."

$SEGMENT(u, i, H, F, \langle v^{(1)}, \ldots, v^{(m)} \rangle)$

$\equiv_{def.}$ There exists w such that u is an initial segment of w,

$CRITICAL(w, i, H, F)$ is true and

no $v^{(j)}$ equals w $(j = 1, \ldots, m)$.

Hereafter, whenever we talk about a 1-query formula F, we assume that i and H satisfy $1QFORMULA(i, H, F)$.

$TAUTALL$ belongs to $coNP$ (see [13]) and $SEGMENT$ belongs to NP. Hence there are two deterministic polynomial time-bounded oracle Turing machines $N_{ALL}[X]$ and $N_{SEG}[X]$ such that Lang($N_{ALL}[TAUT]$) = $TAUTALL$ and Lang($N_{SEG}[TAUT]$) = $SEGMENT$. Using these machines, we construct two oracle machines $T[X]$ and $M[X]$ as follows.

Algorithm 7.1 ($T[\sim]$ and $M[\sim]$)

```
transducer $T[Y]$ (input $F$ : 1-query formula)
begin
  $List := \langle\rangle$; /* the empty list */
  while $N_{SEG}[Y]$ accepts $\langle \lambda, i, H, F, List \rangle$ do
    $u := \lambda$;
    for $i$ times do
      if $N_{SEG}[Y]$ accepts $\langle u0, i, H, F, List \rangle$
        then $u := u0$;  else $u := u1$;
      end-if;
    end-for;
    Add $u$ to $List$;
  end-while;
  output( $List$ )
end {$T[Y]$}

machine $M[Y \oplus Z]$ (input $F$ : 1-query formula)
begin
  $List := T[Y](F)$;
  if $List$ is empty then accept;
  $\langle v^{(1)}, \ldots, v^{(m)} \rangle := List$;
  $u_j := Z^i(v_1^{(j)}, \ldots, v_i^{(j)})$, for each $j = 1, \ldots, m$;
  $G :=$ the formula "$(\bigwedge_{j=1}^{m}(u_j \Leftrightarrow \xi^i(v_1^{(j)}, \ldots, v_i^{(j)})) ) \Rightarrow F$ ";
  if $N_{ALL}[Y]$ accepts $G$
    then accept;  else reject;
  end-if;
end {$M[Y \oplus Z]$}
```

Suppose that F is a 1-query formula and u is a string whose length is i. When $T[TAUT]$ runs on input F, u is added to *List* if and only if $CRITICAL(u, i, H, F)$ is true. Hence if F is a tautology with respect to an oracle A then it is accepted by $M[TAUT \oplus A]$. Conversely, the final test in $M[X]$ guarantees that any formula accepted by $M[TAUT \oplus A]$ is a tautology with respect to A. Hence (1) of **Lemma 7.3** holds.

Next, we show (2). Note that if a forcing condition forces a given 1-query formula, then the function should be an extension of the forcing condition specified by the 1-query formula. Thus, if F is a 1-query formula and a forcing condition S forces F, then for each $j \leq 2^n - 1$ such that $CRITICAL(z(2^i - 1 + j), i, H, F)$ is true, $z(j)$ belongs to $\mathrm{dom}(S)$. Therefore, for every A that is 1-generic in the sense of Dowd, there exists a polynomial $p(x)$ such that the while-loop of $T[TAUT]$ terminates within $p(|F|)$ steps whenever input F is a tautology with respect to A. □

Proof of Theorem 7.1 for $r = 1$*:* Now, for each A that is a 1-generic oracle in Dowd's sense, we construct a machine by adding an appropriate clock to $M[X]$ in **Lemma 7.3**. This machine witnesses $1TAUT[A] \leq_T^P TAUT \oplus A$. □

7.3 Algorithm for general case

In this section, we present a proof of **Theorem 7.1** without the assumption of $r = 1$.

We begin with rewriting a given r-query formula so that the resulting formula is longer than the original one but it has fewer occurrences of query symbols. Then we shall investigate a fast algorithm for the rewriting. Let r and i be positive integers and H a query free formula. $\natural\langle r, i, H\rangle$ denotes the following r-query formula.

$$\Big(\bigwedge_{j=1}^{r} (a^{(j)} \Leftrightarrow \xi^i(q_1^{(j)}, \ldots, q_i^{(j)}))\Big) \Rightarrow H.$$

$\sharp\langle r, i, H\rangle$ denotes the r-query formula below :

$$\Big[\textstyle\bigwedge_{j=1}^{r} \quad (a^{(j)} \Leftrightarrow \xi^i(q_1^{(j)}, \ldots, q_i^{(j)}))\Big] \Rightarrow$$
$$\Big[[\textstyle\bigwedge_{j=1}^{r} \quad \Big((\displaystyle\bigwedge_{k=1}^{i} (q_k^{(j)} \Leftrightarrow q_k^{(r+1)})) \Rightarrow (a^{(j)} \Leftrightarrow a^{(r+1)})\Big)] \Rightarrow H\Big].$$

Note that the length of $\natural\langle r, i, H\rangle$ is given by a polynomial of $r + i +$ the length of H and the length of $\sharp\langle r, i, H\rangle$ is similarly bounded.

From now on, we fix a positive integer r throughout this section. For an oracle A, $(r+1)CRITICAL[A]$ denotes the collection of all triples $\langle w, i, H\rangle$ for which the following two conditions hold.

(1) i is a positive integer, H is a query free formula and w is a bit string whose length is i.

(2) $\sharp\langle r, i, \bigwedge_{k=1}^{i}(w_k \Leftrightarrow q_k^{(r+1)}) \Rightarrow H\rangle$ does not belong to $rTAUT[A]$.

Proposition 7.4 *Let i and m be positive integers and H a query free formula. And, let A be an oracle and $u, v^{(1)}, \ldots, v^{(m)}$ bit strings. Assume the following four conditions hold.*

(i) $|u| = m$,

(ii) $|v^{(l)}| = i$ *and* $u_l = A^i(v_1^{(l)}, \ldots, v_i^{(l)}) \quad (l = 1, \ldots, m)$,

(iii) G is the query free formula below:

$$\bigwedge_{l=1}^{m}\Big((\bigwedge_{k=1}^{i}(v_k^{(l)} \Leftrightarrow q_k^{(r+1)})) \Rightarrow (u_l \Leftrightarrow a^{(r+1)})\Big),$$

(iv) $\{v^{(1)}, \ldots, v^{(m)}\} = \{w : \langle w, i, H\rangle \in (r+1)CRITICAL[A]\}$.

Then the following two assertions are equivalent.

(a) $\sharp\langle r, i, G \Rightarrow H\rangle \in rTAUT[A]$.

(b) $\natural\langle r+1, i, H\rangle \in (r+1)TAUT[A]$.

Proof: First, we show that (a) implies (b). Assume (a). Then, the formula $\natural\langle r+1, i, G \Rightarrow H\rangle$ belongs to $(r+1)TAUT[A]$. However, by (ii) and (iii), $\natural\langle r+1, i, G\rangle$ also belongs to $(r+1)TAUT[A]$. Hence (b) holds.

Next, we show that the negation of (a) implies the negation of (b). Assume $\sharp\langle r, i, G \Rightarrow H\rangle$ does not belong to $rTAUT[A]$. Then, for some truth assignment $\mathcal{V}$, the following four formulas are true :

(1) $\bigwedge_{j=1}^{r}(a^{(j)} \Leftrightarrow A^i(q_1^{(j)}, \ldots, q_i^{(j)}))$,

(2) $\bigwedge_{j=1}^{r}\Big((\bigwedge_{k=1}^{i}(q_k^{(j)} \Leftrightarrow q_k^{(r+1)})) \Rightarrow (a^{(j)} \Leftrightarrow a^{(r+1)})\Big)$,

(3) G,

(4) $\neg H$.

We define a bit string w whose length is i as follows. For each k $(k = 1, \ldots, i)$, let $w_k = \mathcal{V}(q_k^{(r+1)})$ i.e. the truth value of the atom $q_k^{(r+1)}$. Then, the following formula is true with respect to $\mathcal{V}$:

(5) $\bigwedge_{k=1}^{i}(w_k \Leftrightarrow q_k^{(r+1)})$.

Since (1), (2), (4) and (5) are true with respect to $\mathcal{V}$, $\langle w, i, H\rangle$ belongs to $(r+1)CRITICAL[A]$. Therefore, by (iv), there is an integer n $(1 \leq n \leq m)$ such that $w = v^{(n)}$. Then, the following formulas are true with respect to $\mathcal{V}$:

(6) $(\bigwedge_{k=1}^{i}(v_k^{(n)} \Leftrightarrow q_k^{(r+1)})) \Rightarrow (u_n \Leftrightarrow a^{(r+1)})$ (by (3)),

(7) $\bigwedge_{k=1}^{i}(v_k^{(n)} \Leftrightarrow q_k^{(r+1)})$ (by (5)),

(8) $a^{(r+1)} \Leftrightarrow A^i(q_1^{(r+1)}, \ldots, q_i^{(r+1)})$ (by (ii), (6) and (7)),

(9) $\bigwedge_{j=1}^{r+1}(a^{(j)} \Leftrightarrow A^i(q_1^{(j)}, \ldots, q_i^{(j)}))$ (by (1) and (8)).

Since (4) and (9) are true with respect to $\mathcal{V}$, $\natural\langle r+1, i, H\rangle \notin (r+1)TAUT[A]$. Thus we have shown that the negation of (a) implies the negation of (b). □

Remark. We did not use (iv) to show that (a) implies (b). The statements (1), (8) and (9) of the above proof are not formulas of the relativized propositional calculus in the strict sense but the interpretations of formulas with respect to the particular oracle. However, we abuse terminology. □

Our next problems for a given formula $F = \natural\langle r+1, i, H\rangle$, are the following two.

How long is the rewritten formula $\sharp\langle r, i, G \Rightarrow H\rangle$?

How fast can we rewrite F by a deterministic algorithm ?

The length of the rewritten formula is determined by the cardinality of the following set:

(6.1) $$B = \{w : \langle w, i, H\rangle \in (r+1)CRITICAL[A]\}.$$

For the second problem, it is enough to construct a deterministic transducer that outputs a list of all the members of the above set B in polynomial time when it runs on input F.

Proposition 7.5 *Suppose i is a positive integer and H a query free formula. Let A be an oracle, S a forcing condition, and B the set given by (6.1). Assume $S \sqsubseteq A$, and assume that S forces $\natural\langle r+1, i, H\rangle$. Then,* $\mathrm{Card}(B) \leq \mathrm{Card}(\mathrm{dom}(S))$.

Proof: Let C be the following set :

$$\{z(j) : \langle z(2^i - 1 + j), i, H\rangle \in (r+1)CRITICAL[A]\}.$$

Clearly, $\mathrm{Card}(B) = \mathrm{Card}(C)$. Therefore it is sufficient to show that $C \subseteq \mathrm{dom}(S)$.

Assume for a contradiction that $z(n) \in C \setminus \mathrm{dom}(S)$ for some n. Fix such an integer n and put $w = z(2^i - 1 + n)$. Recall that $X^i(w_1, \dots, w_i) = X(z(n))$ for each oracle X. Since $w \in B$, there is a truth assignment $\mathcal{V}$ for which the following four formulas are true :

(1) $\bigwedge_{j=1}^{r}(a^{(j)} \Leftrightarrow A^i(q_1^{(j)}, \dots, q_i^{(j)}))$,

(2) $\bigwedge_{j=1}^{r}\Big((\bigwedge_{k=1}^{i}(q_k^{(j)} \Leftrightarrow q_k^{(r+1)})) \Rightarrow (a^{(j)} \Leftrightarrow a^{(r+1)})\Big)$,

(3) $\bigwedge_{k=1}^{i}(w_k \Leftrightarrow q_k^{(r+1)})$,

(4) $\neg H$.

Define an oracle D as follows. Let $D(z(n)) = \mathcal{V}(a^{(r+1)})$. For strings $u \neq z(n)$, let $D(u) = A(u)$. Then, the following two are true with respect to $\mathcal{V}$:

(5) $(a^{(r+1)} \Leftrightarrow D^i(q_1^{(r+1)}, \dots, q_i^{(r+1)}))$ (by (3)),

(6) $\bigwedge_{j=1}^{r}(a^{(j)} \Leftrightarrow D^i(q_1^{(j)}, \dots, q_i^{(j)}))$ (by (1), (2) and (3)).

Since (4), (5) and (6) are true with respect to $\mathcal{V}$, $\natural\langle r+1, i, H\rangle \notin TAUT^D$. Since $z(n)$ does not belong to $\mathrm{dom}(S)$, we have $S \sqsubseteq D$. Therefore, S does not force $TAUT[X](\natural\langle r+1, i, H\rangle)$; thus we get a contradiction. □

Given an oracle A, let $(r+1)SEGMENT[A]$ denote the collection of all sequences of the form $\langle u, i, H, \langle v^{(1)}, \dots, v^{(m)}\rangle\rangle$ such that for some bit string w the following three conditions hold.

(1) u is an initial segment of w.

(2) $\langle w, i, H\rangle \in (r+1)CRITICAL[A]$.

(3) $w \neq v^{(l)}$ $(l = 1, \dots, m)$.

Proposition 7.6 *For each oracle A, we have*

$$(r+1)SEGMENT[A] \leq_T^P rTAUT[A].$$

Proof: Suppose that i is a positive integer, H is a query free formula and we have $|u| \leq |v^{(1)}| = \cdots = |v^{(m)}| = i$. Then, the sequence $\langle u, i, H, \langle v^{(1)}, \dots, v^{(m)}\rangle\rangle$ belongs to $(r+1)SEGMENT[A]$ if and only if the following formula does not belong to $rTAUT[A]$.

$$\sharp\langle r, i, [(\bigwedge_{k=1}^{|u|}(u_k \Leftrightarrow q_k^{(r+1)})) \wedge (\bigwedge_{l=1}^{m} \neg \bigwedge_{k=1}^{i}(v_k^{(l)} \Leftrightarrow q_k^{(r+1)}))] \Rightarrow H\rangle.$$

□

Lemma 7.7 *There is a deterministic oracle Turing machine $M_{r+1}[X]$ such that (1) and (2) below hold.*

(1) For every oracle A, $\mathrm{Lang}(M_{r+1}[rTAUT[A]]) = (r+1)TAUT[A]$.

(2) If A is an $(r+1)$-generic oracle in the sense of Dowd, then the machine $M_{r+1}[rTAUT[A]]$ accepts the members of $(r+1)TAUT[A]$ in polynomial time.

Proof: As in our proof of **Lemma 7.3**, we construct a deterministic oracle transducer $T_{r+1}[X]$ that works as follows. Running on input $F \equiv_{\mathrm{def.}} \natural\langle r+1, i, H\rangle$, $T_{r+1}[rTAUT[A]]$ outputs the list of all members of the set B as in (6.1). Further, if A is $(r+1)$-generic in the sense of Dowd, then $T_{r+1}[rTAUT[A]]$ terminates within polynomial steps of the length of input F, where such a polynomial depends upon A. Such a construction is possible by **Proposition 7.5** and **7.6**. By using the transducer $T_{r+1}[X]$, again as in our proof of **Lemma 7.3**, we construct a deterministic oracle machine $M_{r+1}[X]$ as follows. When $M_{r+1}[rTAUT[A]]$ runs on input $F = \natural\langle r+1, i, H\rangle$, it checks whether $\sharp\langle r, i, G \Rightarrow H\rangle$ belongs to $rTAUT[A]$, where G is the formula given in (iii) of **Proposition 7.4**. By the fact that A is polynomial time Turing reducible to $rTAUT[A]$, we can construct such a machine $M_{r+1}[X]$. Thus by **Proposition 7.4** and by our construction of $T_{r+1}[X]$, $M_{r+1}[X]$ satisfies the requirements of the lemma. □

Proof of Theorem 7.1: Suppose that r is a positive integer and A is an r-generic oracle in Dowd's sense. By **Lemma 7.7**, for any positive integer $s < r$, $(s+1)TAUT[A]$ is polynomial time Turing equivalent to $sTAUT[A]$, and hence we have $rTAUT[A] \equiv_T^P 1TAUT[A]$. Thus, by **Theorem 7.1** for $r = 1$, we have **Theorem 7.1** for all $r \geq 1$. □

7.4 Conclusion of this chapter

For each positive integer r, the following two statements are equivalent (**Theorem 7.2**).

(1) If A is a random oracle then $rTAUT[A] \notin P[A]$ with probability 1.

(2) The unrelativized class R does not equal NP.

Proof of Theorem 7.2: Consider the following five subsets of the Cantor space;

$$D_1 = \{X : TAUT \leq_T^P X\},$$

$$D_2 = \{X : X \text{ is } r\text{-generic in the sense of Dowd}\},$$

$$D_3 = \{X : rTAUT[X] \equiv_T^P TAUT \oplus X\},$$

$$D_4 = \{X : rTAUT[X] \leq_T^P X\},$$

$$D_5 = \{X : rTAUT[X] \notin P[X]\}.$$

First, $R = NP$ if and only if $NP \subseteq BPP$ (by Ko [19]). The latter is equivalent to the assertion $coNP \subseteq BPP$. Next, $TAUT \in BPP$ if and only if $\mu(D_1) = 1$ (by Bennet and Gill [6], see also [1]), where μ means Lebesgue measure. Now, $\mu(D_2) = 1$ (see section 4 of Dowd [13]). Therefore, by **Theorem 7.1**, $\mu(D_3) = 1$. Hence $\mu(D_1) = 1$ if and only if $\mu(D_4) = 1$. Moreover, the set D_4 is closed under finite changes i.e. if $A \in D_4$ and $\{x : A(x) \neq B(x)\}$ is a finite set then $B \in D_4$. Consequently, if $\mu(D_4) > 0$ then $\mu(D_4) = 1$. Hence D_5 has Lebesgue measure 1 if and only if $R \neq NP$. □

Chapter 8

Conclusive Remark

A problem that we left open in Chapter 4 is whether the horizontal hierarchy of Diagram 4.2 collapses or not. That is, for each positive integer r, whether $(r+1)\mathrm{GEN}_3$ equals $r\mathrm{GEN}_3$ or not.

We conclude this thesis by giving a remark on the relationship between the 1-query tautologies and $P =?NP$ problem. Recall the following basic property of polynomial time truth-table reducibility [22, Proposition 3.4]. For all sets A and B, A is polynomial time truth-table reducible to B if and only if there is a deterministic oracle Turing machine M and a polynomial time-bounded deterministic transducer T such that the following three requirements are satisfied.

- On each input u, T outputs (the binary representation of) a list $T(u)$ of bit strings.
- M reduces A to B in polynomial time.
- On each input u, M only asks questions of B from the list $T(u)$.

Thus, by analyzing the construction of the machine $M[\sim]$ of **Algorithm 7.1**, it is not hard to see that if $P = NP$ and A is a 1-generic oracle in

Dowd's sense then $1TAUT[A]$ is polynomial time truth-table reducible to A. Hence, by [13, Theorem 10] (**Dowd's Theorem 10**), we get the following.

Lemma 8.1 (Corollary of **Lemma 7.3**) *The following assertion 1 implies the assertion 2.*

1. *If A is a random oracle then $1TAUT[A]$ is not polynomial time truth-table reducible to A with probability one.*

2. *The unrelativizied classes P and NP are not identical.* □

Bibliography

[1] Ambos-Spies, K., "Randomness, relativizations, and polynomial reducibilities," pp. 23-34 in *Structure in Complexity Theory*, Lect. Notes Comput. Sci. vol. 223, edited by A. L. Selman, Springer, Berlin, 1986.

[2] Ambos-Spies, K., H. Fleischhack, and H. Huwig, "*P*-generic sets," pp. 58-68 in *ICALP 84*, Lect. Notes Comput. Sci. vol. 172, edited by J. Paredaens, Springer, Berlin, 1984.

[3] Baker, T., J. Gill, and R. Solovay, "Relativizations of the $P = ?NP$ question," *SIAM J. Comput.*, vol. 4 (1975), pp. 431-442.

[4] Balcázar, J. L., R. V. Book, and U. Schöning, "On bounded query machines," *Theoret. Comput. Sci.*, vol. 40 (1985), pp. 237-243.

[5] Balcázar, J. L., J. Díaz, and J. Gabarró, *Structural complexity I, II*, Springer, Berlin, 1988, 1990.

[6] Bennett, C. H. and J. Gill, "Relative to a random oracle A, $P^A \neq NP^A \neq co\text{-}NP^A$ with probability 1," *SIAM J. Comput.*, vol. 10 (1981), pp. 96-113.

[7] Blum, M. and R. Impagliazzo, "Generic oracles and oracle classes," pp. 118-126 in *28th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society Press, Los Alamitos, 1987.

[8] Book, R. V., "Bounded query machines: on NP and $PSPACE$," *Theoret. Comput. Sci.*, vol. 15 (1981), pp. 27-39.

[9] Book, R. V. and C. Wrathall, "Bounded query machines: on $NP()$ and $NPQUERY()$," *Theoret. Comput. Sci.*, vol. 15 (1981), pp. 41-50.

[10] Cohen, P. J., "The independence of the continuum hypothesis I," *Proc. Nat. Acad. Sci. USA*, vol. 50 (1963), pp. 1143-1148.

[11] Cohen, P. J., "The independence of the continuum hypothesis II," *Proc. Nat. Acad. Sci. USA*, vol. 51 (1964), pp. 105-110.

[12] Dowd, M., "Forcing and the P-hierarchy," Laboratory for Computer Science Research, Rutgers University, Technical Report No. **LCSR-TR-35**, 1982.

[13] Dowd, M., "Generic oracles, uniform machines, and codes," *Information and Computation*, vol. 96 (1992), pp. 65-76.

[14] Feferman, S., "Some applications of the notions of forcing and generic sets," *Fund. Math.*, vol. 56 (1965), pp. 325-345.

[15] Feller, W., *An introduction to probability theory and its applications II (second edition)*, John-Wiley, New York, 1971.

[16] Hinman, P. G., "Some applications of forcing to hierarchy problems in arithmetic," *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, vol. 15 (1969), pp. 341-352.

[17] Jech, T., *Set theory*, Academic Press, San Diego, 1978.

[18] Jockusch, C. G. Jr., "Degrees of generic sets," pp. 110-139 in *Recursion theory: its generalizations and applications*, London Math. Soc. Lect. Note Series vol. 45, edited by F.R. Drake and S.S. Wainer, Cambridge University Press, Cambridge, 1980.

[19] Ko, Ker-I., Some observations on the probabilistic algorithms and NP-hard problems, *Inform. Process. Lett.* **14** (1982) 39-43.

[20] Kunen, K., *Set theory*, North-Holland, Amsterdam, 1980.

[21] Kurtz, S. A., "On the random oracle hypothesis," *Inform. Control*, vol. 57 (1983), pp. 40-47.

[22] Ladner, R. E., N. A. Lynch, and A. L. Selman, "A comparison of polynomial time reducibilities," *Theoret. Comput. Sci.*, vol. 1 (1975), pp. 103-123.

[23] Mehlhorn, K., "On the size of sets of computable functions," pp. 190-196 in *14th Annual Symposium on Switching & Automata Theory*, IEEE Computer Society Press, Los Alamitos, 1973.

[24] Odifreddi, P., "Forcing and reducibilities," *J. Symbolic Logic*, vol. 48 (1983), pp. 288-310.

[25] Odifreddi, P., *Classical recursion theory*, North-Holland, Amsterdam, 1989.

[26] Poizat, B., "$Q = NQ$?," *J. Symbolic Logic*, vol. 51 (1986), pp. 22-32.

[27] Rogers, H. Jr., *Theory of recursive functions and effective computability*, MacGraw-Hill, New York, 1967.

[28] Suzuki, T., "Complexity of the r-query tautologies: in presence of a generic oracle," *Notre Dame J. Formal Logic*, *to appear*.

[29] Suzuki, T., "Recognizing tautology by a deterministic algorithm whose while-loop's execution time is bounded by forcing," *Kobe Journal of Mathematics, to appear.*

[30] Suzuki, T., ' Forcing complexity: supplement to "complexity of the r-query tautologies," ' *preprint.*

[31] Tanaka, H. and M. Kudoh, "On relativized probabilistic polynomial time algorithms," *J. Math. Soc. Japan*, vol. 49 (1997), pp. 15-30.

Index