Degrees of Dowd-type Generic Oracles

Toshio Suzuki

Osaka Prefecture University
Sakai, Osaka 599-8531, Japan
suzuki@mi.cias.osakafu-u.ac.jp

Information and Computation 176, pp. 66–87 (2002)
doi:10.1006/inco.2002.3149

Degrees of Dowd-type Generic Oracles

Toshio Suzuki *
Department of Mathematics and Information Sciences
Osaka Prefecture University, Sakai, Osaka 599-8531, Japan
suzuki@mi.cias.osakafu-u.ac.jp

April 9, 1999; final manuscript March 24, 2002

Abstract

For each positive integer r, Dowd (Information and Computation 96 (1992)) introduced r-generic oracles (we call them r-Dowd oracles; they are different from n-genericity of arithmetical forcing). An oracle D is r-Dowd if every r-query tautology with respect to D is forced by a polynomial-sized portion of D. We propose the study of degrees and complexity of 1-Dowd oracles. Dowd (1992) stated that no r-Dowd oracle is recursively enumerable. However, this is false. We show, among others, the following. There exists a primitive recursive 1-Dowd oracle; For every oracle A, there exists a 1-Dowd oracle D that is Turing-equivalent to A; For every 1-Dowd oracle D, there exists a 1-Dowd oracle E such that E is polynomial time many-one-equivalent to D and E is not 2-Dowd. Problems are formulated, and analogy between the jump-operator and the operation of taking the set of 1-query tautologies is discussed.

Keywords: Structural complexity; Generic oracle; The relativized propositional calculus.

Mathematics Subject Classification: Primary 68Q15; Secondary 03D15.

1 Introduction

In the theory of computational complexity, it is often asked whether there exists a set of bit strings satisfying a given property relating to complexity. The origin of problems of this type can be observed in classical recursion theory. The construction of incomparable degrees due to Kleene and Post is quite well-known. The method used in such a construction is sometimes called the finite extension method [Od 89, p.456]. We may think of it as an archetype of forcing. In fact, the introduction of arithmetical forcing by [Fe 65] and [Hi 69] enabled us to deal with the finite extension method and forcing in a unified manner.

*Research partially supported by Grant-in-Aid for Scientific Research (No. 09440072, 11740073), Ministry of Education, Science, Sports and Culture of Japan.

The forcing theorem of set theory states that every sentence true in a generic extension is forced by a forcing condition in the corresponding generic filter, and vice versa (see [Je 78, p.142]). Whereas, suppose that G is a generic set of arithmetical forcing and φ is an arithmetical sentence; then, φ is satisfied by G if and only if some finite portion of G forces φ [Jo 80, Lemma 2.2]. Thus, each requirement in the finite extension method corresponds to the problem whether some finite portion of a generic set forces the requirement, and we can restate this problem as the problem whether the corresponding set of forcing conditions is dense.

More recently, arithmetical forcing was applied to the study of the polynomial hierarchy and its variations [Po 86, BI 87, CIY 97]. Since 1980's, many applications of forcing to computational complexity have appeared. It is not the aim of the current paper to give an exhaustive survey of forcing in computational complexity. But, we remark that not only arithmetical forcing itself has been applied to this area, but also much effort have been made to formulate feasible versions of arithmetical forcing. One of famous streams in this line is what is called *resource-bounded genericity* ; an excellent survey is presented in [Am 96]. Very roughly speaking, the basic idea of resource-bounded genericity is to put a bound on time-complexity of a finite-extension strategy; then, they formulate feasible versions of comeager classes and open-dense classes; their concept of generic sets is formulated as sets that belong to sufficiently many "open-dense" classes.

However, there are different approaches to "feasible forcing." Dowd's concept of r-generic oracles is such an example. In this line of approach, we do not define generic sets to be sets that belong to sufficiently many comeager classes. We do not consider time-complexity of a finite-extension strategy, but consider the minimal size of a forcing condition that forces a given formula. In the most basic cases, formulas considered are those of *the relativized propositional calculus*, though, later, the idea was generalized to (second order) arithmetical predicates [Su (a), Su (b)]. The relativized propositional calculus is obtained by adding a set of query symbols $\{\xi^n : n \in \mathbb{N}\}$ to the propositional calculus. Each $\xi^n(q_1, \ldots, q_n)$ is an n-ary connective that is interpreted as the initial segment of a given oracle (we identify an oracle with its characteristic function) up to 2^nth bit string in length-lexicographic order. For example, suppose $n = 2$ and A is a given oracle. Then we define a function A^2 on $\{0,1\}^2$ as follows. $A^2(00) = A(\lambda)$, where λ is the empty string, $A^2(01) = A(0)$, $A^2(10) = A(1)$ and $A^2(11) = A(00)$. And, we interpret $\xi^2(q_1, q_2)$ as $A^2(q_1q_2)$. The set of all relativized tautologies with respect to a given oracle A is naturally defined, and we denote it by TAUT$[A]$. We often consider, for a fixed positive integer r, a relativized formula that has exactly r occurrences of query symbols, which we call an *r-query formula*. Each r-query formula belonging to TAUT$[A]$ is called an *r-query tautology with respect to A*. The set of all r-query tautologies with respect to A is denoted by rTAUT$[A]$. Clearly, each relativized tautology with respect to a given oracle A is forced to be a tautology by some finite portion of A, where "force" means that the relativized formula is a tautology with respect to every oracle extending the finite portion. However, the finite portion may

have very big size (of its domain).

Example 1. Let $\emptyset$ be the characteristic function of the empty set. For each n, let F_n be the following formula.

$$0 \Leftrightarrow \xi^n(q_1, \ldots, q_n).$$

Each F_n is a tautology with respect to $\emptyset$. But, the size of the minimal finite portion S_n of $\emptyset$ such that S_n forces F_n (to be a tautology) is exponential in n. □

Dowd defined t-generic oracles and r-generic oracles by investigating size of forcing conditions [Do 92]. An oracle D is *t-generic* if there exists a polynomial p such that for each $F \in \mathrm{TAUT}[D]$, there exists a finite portion S of A such that S forces F (to be a tautology) and the size of (the domain of) S is at most $p(|F|)$, where $|F|$ is the length of (the binary representation of) F. Let r be a positive integer. An oracle D is *r-generic in the sense of Dowd* if it satisfies the definition of a t-generic oracle for $r\mathrm{TAUT}[D]$ in place of $\mathrm{TAUT}[D]$. Thus, these definitions of genericity are based on analogy of forcing theorem rather than analogy of the argument about dense sets. Dowd showed that t-generic oracles do not exist, but for each r, the set of all r-generic oracles has Lebesgue measure one in the Cantor space. He studied basic properties of r-generic oracles. In particular, if NP = coNP, then for each r-generic oracle D in the sense of Dowd, we have $r\mathrm{TAUT}[D] \in \mathrm{NP}[D]$. Moreover, he showed that if G is a generic oracle of arithmetical forcing then G is not 1-generic in his sense.

Convention. In the following, we use the terminology an *r-Dowd oracle* to denote an r-generic oracle in the sense of Dowd. One reason is to respect the pioneer, and the other reason is to distinguish them explicitly from n-genericity of arithmetical forcing. And, we use the terminology a *tautology-generic oracle* to denote a t-generic oracle. In the following, in default of a specification, generic oracles and n-generic oracles mean those of arithmetical forcing.

In our previous papers, we extended Dowd's work. It was shown in [Su 98] that for each r, the following two assertions are equivalent:

- If A is a random oracle then $r\mathrm{TAUT}[A]$ is not polynomial time Turing-reducible to A with probability one.
- The unrelativized computational complexity classes R and NP are not identical.

See section 7 for the definition of R; R is a well-known complexity class such that $\mathrm{P} \subseteq \mathrm{R} \subseteq \mathrm{NP}$ and $\mathrm{R} \subseteq \mathrm{BPP}$. Note that if A is a random oracle then $\mathrm{TAUT}[A]$ is not polynomial time Turing-reducible to A with probability one [BG 81].

In [Su (a)], behavior of r-query tautologies relative to a generic oracle (of arithmetical forcing) was investigated. In particular, it was shown that if G is a generic oracle then $\mathrm{TAUT}[G]$ does not belong to $\mathrm{NP}[r\mathrm{TAUT}[G]]$. The proof uses

an extension of the concept of r-Dowd oracles to an arbitrary (second-order) arithmetical predicate (cf. Section 8).

Unfortunately, Dowd's proof of non-existence of tautology-generic oracles has a passage which certainly seems to be a fatal logical gap. More precisely, he showed non-existence of tautology-generic oracles by using [Do 92, Lemma 6], and he "proved" the lemma by showing that a certain set is sparse; however, there is no enough reason why the set is sparse. We analyzed this logical gap by showing a counterexample in [Su (a)].

However, by developing the study of the minimal size of the forcing condition that forces a given (second-order) arithmetical predicate, an alternative proof of non-existence of tautology-generic oracles was given in [Su (b)]. Moreover, an improved rigorous proof that r-Dowd oracles form a measure-one subset of the Cantor space was also given in [Su (b)]. Preliminary versions of [Su 98], [Su (a)] and [Su (b)] appeared as chapters of the author's doctoral dissertation [Su 99].

Aim of this paper. In this paper, we propose the study of degrees and complexity of 1-Dowd oracles.

It is well-known that there exists a 1-generic oracle (of arithmetical forcing) below $\emptyset'$ [Jo 80]; that is, there exists an oracle G such that G is Cohen-generic for 1-quantifier arithmetic and G is Turing-reducible to the halting problem. Whereas, Jockusch and Posner showed that 1-generic oracles are immune, and hence they are not recursively enumerable [So 87, p.99] (see also [JP 78, Do 82]).

We consider the case of 1-Dowd oracles. In the following passage cited from [Do 92, p.72], an r-generic oracle means an r-Dowd oracle.

Citation 1 (line 6–9 of p.72 of [Do 92])

> Direct arguments show that the r-generic oracles are closed under complementation and finite changes, and it follows as in (Dowd, 1982) that an r-generic oracle is not r.e.

The last sentence can be understood as nothing but the assertion that no r-Dowd oracle is recursively enumerable. The sentence does not mean the existence of at least one r-Dowd oracle which is not recursively enumerable, not only because of common usage of the indefinite article in English writing, but also because of the following contextual evidence. The cited passage appears after the proof [Do 92, p.71] that for each r, the class of all r-Dowd oracles has Lebesgue measure one in the Cantor space. Since every class of measure-one obviously has uncountable cardinality, it is trivial that such a class has at least one element which is not recursively enumerable.

However, it is false that no 1-Dowd oracle is recursively enumerable. In this paper, we show that there exists a 1-Dowd oracle which is not only recursively enumerable, but also primitive recursive. More generally, we show that for every oracle A, there exists a 1-Dowd oracle D that is Turing-equivalent to A (the Ubiquity Theorem of 1-Dowd Oracles). The Ubiquity Theorem is shown in Section 4. The idea of the proof is that, although lexicographic order of infinite binary sequence is not a well-ordering, some particular class of 1-Dowd

oracles has a least element with respect to this ordering. None of the following complexity classes contains any 1-Dowd oracle: NP, coNP, BPP, P/poly. We present these examples about complexity classes in Section 5. Polynomial time many-one-degrees of 1-Dowd oracles also have an interesting property. For every 1-Dowd oracle D, there exists an oracle E such that E is polynomial time many-one-equivalent to D, E is also 1-Dowd but E is not 2-Dowd (the Fragility Theorem on 2-Dowd Property). The Fragility Theorem is shown in Section 6. The proof is done by comparing the closure property of the class of all 1-Dowd oracles with that of 2-Dowd oracles. In the remaining Sections 7–9, for further study on degrees and complexity of 1-Dowd oracles, we formulate problems and explain their background. In particular, we discuss analogy between the jump-operator and the operation of taking the set of 1-query tautologies with respect to a given oracle. In order to avoid too long an introduction, we have not mentioned some results and related facts here; these topics are summarized in Section 10. Section 10 also includes diagrams that display the relationship of complexity classes mentioned in this paper. In Section 3, we present a basic investigation that is a prototype of arguments developed throughout the current paper. Section 3 and Section 9 are based on chapter 8 and chapter 6 of the author's doctoral dissertation [Su 99], respectively.

Background knowledge. See [BDG 95] for basic concepts of computational complexity, and see [Ro 67] or [Od 89] for classical recursion theory. Regarding generic oracles of arithmetical forcing, a concise explanation is given in [CIY 97, section 2], which is sufficient for reading the current paper. In fact, **Example 5** in Section 5 is the only one result in this paper whose *proof* requires knowledge of arithmetical forcing. **Example 5** would be a so-called folklore result, and, we do not use **Example 5** in the proofs of other results. Rather, knowledge of arithmetical forcing is helpful for understanding the *motive* for our arguments. The following references are only for those who are interested in deep background; other references of generic oracles can be found in the list of references of [CIY 97]. If the reader is interested in the basic role of generic oracles in computational complexity, consult [Po 86] and [BI 87]. Standard references of recursion-theoretical background of arithmetical forcing are [Jo 80], [Kum 96] and [Od 99, Chapter XII]. See also [Od 83]. As an introduction to forcing, [Kun 80] is a standard textbook.

2 Notation

Basic notation. If the reader is familiar with one of [Su 99] and [Su (b)], then he or she may skip the following four paragraphs and may go to the last paragraph in this section titled **Notation particular to this paper**.

For a set X, $\mathrm{Card}(X)$ denotes the cardinality of X. If f is a function and X is a subset of the domain of f, then $f \restriction X$ denotes the restriction of f to X. We denote $\log_2 x$ by $\log x$. The set of all bit strings of finite lengths is denoted by $\{0,1\}^*$. The set of all natural numbers is denoted by $\mathbb{N}$. An *oracle* means a sub-

set of $\{0,1\}^*$. We identify an oracle with its characteristic function. Usually, a set of oracles is called a *class*. Turing-reducibility (truth-table-reducibility, many-one-reducibility, one-one-reducibility, respectively) is abbreviated to T-reducibility (tt-reducibility, m-reducibility, 1-reducibility) and denoted by $\leq_T$ ($\leq_{tt}$, $\leq_m$, $\leq_1$, respectively). Recall the definition of tt-reducibility [Od 89, p.268]. The concept of *truth-table conditions* (tt-conditions, for short) is inductively defined as follows, where $X(\sim)$ is a unary symbol.

- For each bit string u, $X(u)$ is a tt-condition. We call it an *atomic tt-condition*.
- If s and t are tt-conditions, then $s \wedge t$, $s \vee t$ and $\neg s$ are tt-conditions.

We say an oracle A is tt-reducible to an oracle B if there exists a recursive function

$$f : \{0,1\}^* \to \{0,1\}^*$$

such that, for each bit string u, $f(u)$ is (the binary code of) a tt-condition, and the necessary and sufficient condition for $u \in A$ is that the tt-condition (coded by) $f(u)$ is satisfied by B, where we interpret $X(\sim)$ as membership in B. Polynomial time tt-reducibility was introduced and studied in [LLS 75]. The following is the basic characterization of polynomial time tt-reducibility [LLS 75, Proposition 3.4]. For every sets A and B, A is polynomial time tt-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 $T(u)$.

Polynomial time tt-reducibility is abbreviated to P-tt-reducibility and denoted by $\leq^P_{tt}$. Polynomial-time counterparts of other reducibilities are denoted in similar ways.

If A is an oracle then $\mathrm{P}[A]$ denotes the class of all oracles that are P-T-reducible to A. An oracle B belongs to $\mathrm{NP}[A]$ if there are a polynomial p and a set $C \in \mathrm{P}[A]$ such that for all bit string u, we have: $u \in B \Leftrightarrow \exists w \in \{0,1\}^* [|w| \leq p(|u|)$ and $\langle u, w\rangle \in C]$, where $|w|$ is the length of w. If $\mathcal{K}$ is a class of oracles then $\mathrm{P}[\mathcal{K}]$ denotes the class of all oracles that belong to $\mathrm{P}[A]$ for some $A \in \mathcal{K}$; if $\mathcal{C}$ is a complexity class such as NP then $\mathcal{C}[\mathcal{K}]$ is defined in a similar way. The polynomial hierarchy is defined as follows. Σ^P_0 $(= \Pi^P_0 = \Delta^P_0)$ denotes P and Σ^P_1 denotes NP. For each natural number n, Σ^P_{n+1} denotes $\mathrm{NP}[\Sigma^P_n]$, Δ^P_{n+1} denotes $\mathrm{P}[\Sigma^P_n]$ and Π^P_n denotes $co\Sigma^P_n$. It is easy to see $\mathrm{P} = \Delta^P_1$. The arithmetical hierarchy is defined as follows. Σ^0_0 $(= \Pi^0_0 = \Delta^0_0)$ denotes the class of all recursive sets and Σ^0_1 denotes the class of all recursively enumerable sets.

If A is an oracle then $\Sigma_0^0[A]$ denotes the class of all oracles that are T-reducible to A. An oracle B belongs to $\Sigma_1^0[A]$ if there is a set $C \in \Sigma_0^0[A]$ such that for all bit string u, we have: $u \in B \Leftrightarrow \exists w \in \{0,1\}^* \ \langle u, w\rangle \in C$. For each natural number n, Σ_{n+1}^0 denotes $\Sigma_1^0[\Sigma_n^0]$, Δ_{n+1}^0 denotes $\Sigma_0^0[\Sigma_n^0]$ and Π_n^0 denotes $co\Sigma_n^0$. It is easy to see $\Delta_0^0 = \Delta_1^0$. Our notation of the polynomial hierarchy is based on [BDG 95, Chapter 8]. Note that different usage of the symbol Δ_n^P appears in some papers, especially in [BI 87, Theorem 3.5]. Note also that the polynomial hierarchy is defined as the hierarchy of functions in some papers, especially in [Bu 86], but in this paper, the polynomial hierarchy denotes the hierarchy of oracles in default of a specification. Similar convention also applies to the arithmetical hierarchy.

Next, we review the relativized propositional calculus. The interpretation of the connective ξ^n is defined as follows. Let A be an oracle. To clarify our argument, we explain the case of $n = 3$. For each bit string, we assign the natural number denoting its position in length-lexicographic order. The empty bit string is denoted by either λ or $z(0)$. Likewise, 0, 1, 00, 01, ... are denoted by $z(1)$, $z(2)$, $z(3)$, $z(4)$, ..., respectively. Let Str(3) be the set $\{z(0), \ldots, z(2^3 - 1)\}$. That is, Str(3) is $\{\lambda, 0, 1, \ldots, 000\}$. By using the initial segment of (the characteristic function of) A on Str(3), we define the function A^3 on $\{0,1\}^3$ as follows. $A^3(000) = A(z(0))$, $A^3(001) = A(z(1))$, $A^3(010) = A(z(2))$, ..., $A^3(111) = A(z(7))$. Thus, by writing each non-negative integer $k \leq 7$ in the 3-bit binary notation, we may write

$$A^3(k) = A(z(k)).$$

For general cases, the n-ary Boolean function A^n is defined by using

$$\mathrm{Str}(n) =_{\mathrm{def.}} \{z(0), \ldots, z(2^n - 1)\} = \{\lambda, 0, 1, \ldots, 0^n\}$$

instead of Str(3). And, for each n, we have $A^{n+1}(0q_1 \cdots q_n) = A^n(q_1 \cdots q_n)$. The interpretation of $\xi^n(q_1, \ldots, q_n)$ for the oracle A is given by $A^n(q_1 \cdots q_n)$. For each relativized formula F, $\ell(F)$ denotes the total number of occurrences of propositional variables, constants, logical connectives, query symbols (ξ^1, ξ^2, ξ^3, ...) and punctuation marks (parentheses and commas). For all positive integers r, n and all query free formula H, $\natural\langle r, n, H\rangle$ denotes the formula defined as follows.

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

An *r-query formula* means a formula of the form $\natural\langle r, n, H\rangle$ for some n and some query free formula H. The reason why we restrict ourselves to formulas of the above form is technical, and the only important property of r-query formulas is that they have exactly r occurrences of query symbols. If A is an oracle and F is an r-query formula that is a tautology with respect to A, then we say F is an *r-query tautology with respect to A*. rTAUT[A] (TAUT[A], TAUT, respectively) denotes the set of all r-query tautologies with respect to

A (all relativized tautologies with respect to A, all tautologies of the usual propositional calculus, respectively).

A *forcing condition* means a function whose domain is a finite subset of $\{0,1\}^*$ and range is a subset of $\{0,1\}$. In [CIY 97], a forcing condition is called a *finite oracle*. If S is a forcing condition, A is an oracle and for every $u \in \mathrm{dom}(S)$ we have $S(u) = A(u)$, then we say "S is a finite portion of A" or we say "A is an extension of S." Suppose n is a positive integer. $\mathrm{Func}(n)$ denotes the collection of all forcing conditions whose domains are $\mathrm{Str}(n)$. We say a forcing condition S *forces* a relativized formula F (to be a tautology) if F is a tautology with respect to every oracle B such that B is an extension of S.

Notation particular to this paper. If F is of the form $\natural\langle r, n, H\rangle$ for some positive integers r, n and some query free formula H, then we say *the dimension of* F is n. Suppose r is a positive integer. An oracle is called *r-Dowd* if it is r-generic in the sense of Dowd. That is, an oracle D is r-Dowd if there exists a polynomial p such that for each $F \in r\mathrm{TAUT}[D]$, there exists a forcing condition S such that S is a finite portion of D, S forces F and the size of (the domain of) S is at most $p(|F|)$, where $|F|$ is the length of (the binary representation of) F. We conclude this section by stating the following convention of abuse of the terminology "force." Suppose F is a relativized formula of dimension n. When it is clear from the context that S^n is a forcing condition whose domain is a subset of $\{0,1\}^n$ and that S^n is not considered to be a finite portion of a given oracle A but to be a finite portion of the n-ary Boolean function A^n, we sometimes say "S^n forces F " in order to mean the following assertion: "For every oracle B such that the n-ary Boolean function B^n is an extension of S^n, F is a tautology with respect to B."

3 Basic investigation

Dowd showed that if there exists a 1-Dowd oracle D such that $1\mathrm{TAUT}[D] \notin \mathrm{NP}[D]$, then we have $\mathrm{NP} \neq \mathrm{coNP}$ [Do 92, Theorem 11]. However, even if we replace "$1\mathrm{TAUT}[D] \notin \mathrm{NP}[D]$" by a much weaker assertion, we get an interesting conclusion.

The following investigation is a prototype of arguments developed throughout the current paper. We expect it help the reader understand the main arguments of this paper. However, the content of this section is merely a slight technical improvement of [Su 98, Lemma 3]. Therefore, if the reader knows [Su 98] well, he or she may skip this section and may go to the next section.

Lemma 1 *If there exists an oracle D such that D is 1-Dowd and $1\mathrm{TAUT}[D]$ does not P-tt-reduce to D, then we have $\mathrm{P} \neq \mathrm{NP}$.*

Proof: In the proof, the following three concepts play important roles: size of forcing conditions, lexicographic order and Δ_2^P $(= \mathrm{P}[\mathrm{TAUT}])$.

First, as is shown in [Do 92, Lemma 9], if a 1-query formula F is a tautology with respect to some oracle X, then there exists a *unique* minimal forcing condition S_F that forces F. More precisely, there exists a forcing condition S_F such that for every oracle A, the necessary and sufficient condition for $F \in 1\mathrm{TAUT}[A]$ is that A is an extension of S_F. Speaking in Dowd's terminology, F *specifies* S_F.

Review 1: The proof of the existence of S_F in [Do 92] can be summarized as follows. Suppose H is a query free formula, n is a positive integer and F is a relativized formula of the form $\natural\langle 1, n, H\rangle$. In other words, F is of the following form.

$$\left(a^{(1)} \Leftrightarrow \xi^n(q_1^{(1)}, \ldots, q_n^{(1)})\right) \Rightarrow H.$$

Let Critical(F) be the set of all bit strings $u = u_1 \cdots u_n$ of length n such that for at least one truth assignment $i = 0$ or 1 for the variable $a^{(1)}$, the formula

$$H[i/a^{(1)}][u_1/q_1^{(1)}] \cdots [u_n/q_n^{(1)}] \tag{3.1}$$

is not a tautology. Moreover, let Fatal(F) be the set of all bit strings $u = u_1 \cdots u_n$ of length n such that for each truth assignment $i \in \{0, 1\}$ for $a^{(1)}$, (3.1) is not a tautology. Thus, F is a tautology for some oracle X if and only if Fatal(F) is the empty set. In this case, Critical(F) determines the unique n-ary partial function S_F^n by the following property: the domain of S_F^n is Critical(F), and for every n-ary Boolean function X^n, F is a tautology with respect to X^n if and only if X^n is an extension of S_F^n. (End of Review 1)

Recall that, as is shown in [Su 98, Lemma 3], if we can use TAUT as an oracle then the mapping

$$F \mapsto \mathrm{Critical}(F) \tag{3.2}$$

is computable in a polynomial number of steps in $|F| + \mathrm{Card}(\mathrm{Critical}(F))$.

Review 2: The essence of the argument in [Su 98, Lemma 3] is as follows. Suppose a formula F of the form $\natural\langle 1, n, H\rangle$ is given. We consider the complexity of prefix searching for an element of Critical(F). Let Segment(F) be the set of all pairs $\langle u, \mathit{List}\rangle$ with the following property: u is a bit string of length at most n, List is (the binary expression of) a finite subset of $\{0, 1\}^n$ and there exists a bit string $v \in \mathrm{Critical}(F)$ such that v does not belong to List and u is an initial segment (or, a prefix) of v. Note that the following two relations (on u, F and List) belong to NP.

"$u \in \mathrm{Critical}(F)$",

"$\langle u, \mathit{List}\rangle \in \mathrm{Segment}(F)$".

Consider the following procedure searching the least element of Critical(F) with respect to lexicographic order. At the initial stage, we let u be the empty

string and we let *List* be the empty set. If the pair of the empty bit string and the empty set does not belong to Segment(F), then we abandon the search and we conclude that Critical(F) is empty. Otherwise, while the length of u is less than n, we extend u step-by-step as follows. If $u0 \in$ Segment(F), then we put $u := u0$. Otherwise, we put $u := u1$. Finally, the length of u becomes n. Then, u is the least element of Critical(F) with respect to lexicographic order. This procedure terminates within a polynomial number of steps in $|F|$, if we use TAUT as an oracle. By letting *List* be the singleton consisting of the least element of Critical(F), in a similar way, we can find the second-least element of Critical(F) with respect to lexicographic order. In such a way, we can generate a list of all the members of Critical(F), and if we can use TAUT as an oracle, this is done by a deterministic procedure within a polynomial number of steps in $|F| + \mathrm{Card}(\mathrm{Critical}(F))$. (End of Review 2)

Now, suppose that D is a 1-Dowd oracle with respect to a polynomial p. Then, for each $F \in 1\mathrm{TAUT}[D]$, the cardinality of Critical(F) is at most $p(|F|)$. We consider the following (oracle free) Turing machine M.

Turing machine M (**input** F: 1-query formula);
begin

1. If Fatal(F) is non-empty then output 0.
   /* In this case F is not a tautology with respect to any oracle.
   0 is the constant symbol denoting "false." */
2. If the cardinality of Critical(F) is more than $p(|F|)$ then output 0.
3. Produce S_F.
4. Output the following formula: $\bigwedge_{u \in \mathrm{Critical}(F)} S_F^n(u) \Leftrightarrow \xi^n(u)$.
   /* We may regard this formula as a tt-condition. */

end

Suppose that F is a 1-query formula and M on input F outputs a formula F'. Then D satisfies F' if and only if Fatal(F) is empty, the cardinality of Critical(F) is at most $p(|F|)$ and D extends S_F. Therefore, D satisfies F' if and only if F is a tautology with respect to D.

Moreover, the assertion "Fatal(F) is non-empty" belongs to NP. Therefore, if we have P = NP then 1TAUT[D] P-tt-reduces to D by M. □

The uniqueness of the minimal forcing condition S_F is a crucial property of 1-query formulas, and this uniqueness plays an important role in the proof of the existence of 1-Dowd oracles [Do 92, Theorem 10]. As we observed in [Su (b)], there is a 2-query formula which is a tautology for some oracle but the corresponding minimal forcing conditions are not unique. In fact, for example,

consider (the 2-query formula in the sense of our formal definition which is equivalent to) the following formula.

$$\xi^2(01) \Leftrightarrow \xi^2(10).$$

The above formula is a tautology for a given oracle A if and only if $A(z(1)) = A(z(2))$. Thus, there are two minimal forcing conditions that forces the above formula.

4 Ubiquity of 1-Dowd oracles

As we discussed in **Introduction**, Dowd states as follows, where r-generic oracles in the following citation means r-Dowd oracles.

Citation 1 (line 6–9 of p.72 of [Do 92])

> Direct arguments show that the r-generic oracles are closed under complementation and finite changes, and it follows as in (Dowd, 1982) that an r-generic oracle is not r.e.

However, the last sentence of the above citation is false. In this section, we prove the following theorem.

Theorem 2 (**The Ubiquity Theorem of 1-Dowd Oracles**) *For every oracle A, there exists a 1-Dowd oracle D that is Turing-equivalent to A.*

4.1 Where is a gap?

We first examine the meaning of "*it follows as in* (*Dowd, 1982*)" in **Citation 1**. We cite the following passages (without corrections) from [Do 82, section 3], where "fis" means a finite initial segment [Do 82, p.2].

Citation 2 (line 15–16 of p.5 of [Do 82])

> If $\{M_i^X(x)\}$ is a class of oracle machines, an oracle X is generic for the class if whenever M_i^X is identically 1 there is an fis $F \subset X$ such that for all $Y \supset F \quad M_i^Y$ is identically 1.

Citation 3 (line 12–14 of p.6 of [Do 82])

> Generic oracles cannot be Σ_1; for if X is recursively enumerable the statement $X \subseteq Y$ can be verified for the oracle Y, and is true of X. Hence it is true of some fis F, and we have $X \subseteq F$, a contradiction.

Thus, genericity in [Do 82] is a modification of 1-genericity of arithmetical forcing. If X is generic in the sense of [Do 82] then we have the following:

For every Π_1^0 sentence φ, if X satisfies φ then X has a finite portion that forces φ. $\cdots\cdots(*)$

The above property $(*)$ of generic oracles is the essence of the argument in **Citation 3**. However, how can we show $(*)$ under the assumption that X is 1-Dowd? Roughly speaking, if we substitute $\Pi_1^P (= \text{coNP})$ for Π_1^0 then 1-Dowd oracles has a property similar to $(*)$, and hence no 1-Dowd oracle belongs to $\Sigma_1^P (= \text{NP})$; we will discuss it later (**Example 3** in Section 5). The passage "*it follows as in (Dowd, 1982)*" in **Citation 1** would be confusion of the polynomial hierarchy and the arithmetical hierarchy.

4.2 Review: Dowd's counting argument

Dowd [Do 92, p.70] showed existence of 1-Dowd oracles by a counting argument. A modification of his counting argument is the core of our proof of **Theorem 2**. However, Dowd's original argument is difficult to follow. For a self-contained proof, we include Dowd's argument in a refined form in this subsection.

Recall the following notational conventions. For a natural number n, $\text{Str}(n)$ denotes the set $\{\lambda, 0, 1, 00, 01, \ldots, 0^n\}$. And, $\text{Func}(n)$ denotes the set of all forcing conditions whose domains are $\text{Str}(n)$. By the standard order-isomorphism, $\text{Str}(n)$ is mapped to $\{0,1\}^n$. Thus, for each $X \in \text{Func}(n)$, X^n is an n-ary Boolean function. For each relativized formula F, $\ell(F)$ denotes the total number of occurrences of propositional variables, constants, logical connectives, query symbols and punctuation marks. $\natural\langle 1, n, H\rangle$ denotes the following formula: $\left(a^{(1)} \Leftrightarrow \xi^n(q_1^{(1)}, \ldots, q_n^{(1)})\right) \Rightarrow H$. For each 1-query formula F, there exists a unique minimal forcing condition that forces F. We denote the unique minimal forcing condition by S_F. For every oracle X, F belongs to $\text{1TAUT}[X]$ if and only if X extends S_F (see the proof of **Lemma 1**).

We introduce requirements in order to construct a 1-Dowd oracle. For all positive integers k, C and n, and for all query free formula H, we denote by $\text{R}(k, C, n, H)$ the following requirement for a forcing condition $X \in \text{Func}(n)$.

$\text{R}(k, C, n, H)$: "Letting F be $\natural\langle 1, n, H\rangle$, if F belongs to $\text{1TAUT}[X]$ then we have $\text{Card}(\text{dom}(S_F)) \leq \ell(F)^k + C$."

And, for each positive integers k, C and n, we denote by $\text{R}(k, C, n)$ the following requirement for a forcing condition $X \in \text{Func}(n)$.

$\text{R}(k, C, n)$: "For each query free formula H, the requirement $\text{R}(k, C, n, H)$ holds."

If $\text{R}(k, C, n)$ holds for a forcing condition $Y \in \text{Func}(n)$ in place of X, we say "Y satisfies $\text{R}(k, C, n)$." We follow the same convention about $\text{R}(k, C, n, H)$, too. It is easy to see that an oracle D is 1-Dowd if there exist k and C such that for all $n \geq 1$, $D \upharpoonright \text{Str}(n)$ satisfies the requirement $\text{R}(k, C, n)$. Now we show the following.

Fact 1 (Dowd's Counting Argument) [Do 92, p.70] *Suppose k and C are sufficiently large natural numbers. Suppose that n is a positive integer and X is*

a member of $\mathrm{Func}(n)$ *that satisfies* $\mathrm{R}(k, C, n)$. *Then the cardinality of the set*

$$\{Y \in \mathrm{Func}(n+1) : Y \textit{ extends } X, \textit{ and } Y \textit{ does not satisfy } \mathrm{R}(k, C, n+1)\} \tag{4.1}$$

is at most $2^N \cdot (1/2^{n+2})$, *where* $N =_{\mathrm{def.}} 2^n$.

Proof of Fact 1 : If $2^{n+1} \leq C$, then the conclusion is obvious by the following claim.

Claim 1. If $2^{n+1} \leq C$ then every member of $\mathrm{Func}(n+1)$ satisfies $\mathrm{R}(k, C, n+1)$.

Proof of Claim 1: Suppose $2^{n+1} \leq C$. And, suppose that Y is a member of $\mathrm{Func}(n+1)$. If F is a 1-query formula of dimension $n+1$ and F is a tautology with respect to Y, then F is forced (to be a tautology) by Y. And, the size of (the domain of) Y is at most C. Thus, every $Y \in \mathrm{Func}(n+1)$ satisfies $\mathrm{R}(k, C, n+1)$. Q.E.D.(**Claim 1**)

Thus, in the following, we assume $2^{n+1} > C$. We want to count the cardinality of (4.1). We have the following.

$$\begin{aligned}
&(\text{The set (4.1)}) \\
=&\bigcup_H \{Y \in \mathrm{Func}(n+1) : Y \text{ extends } X, \text{ and } Y \text{ does not satisfy} \\
&\qquad \mathrm{R}(k, C, n+1, H)\},
\end{aligned}$$

where H varies over all query free formulas. However, for those H whose length is very long, all $Y \in \mathrm{Func}(n+1)$ satisfy $\mathrm{R}(k, C, n+1, H)$. More precisely, since the cardinality of $\mathrm{Str}(n+1)$ is $2N$, we may assume that H varies over all query free formulas such that

$$\ell(F)^k + C < 2N,$$

where F denotes $\natural\langle 1, n+1, H\rangle$. That is, F denotes the following.

$$\left(a^{(1)} \Leftrightarrow \xi^{n+1}(q_1^{(1)}, \ldots, q_n^{(1)}, q_{n+1}^{(1)})\right) \Rightarrow H.$$

Throughout the rest of the proof of **Fact 1**, whenever we talk about a query free formula H, F denotes $\natural\langle 1, n+1, H\rangle$.

Thus we have the following.

$$\begin{aligned}
&(\text{The set (4.1)}) \\
\subseteq &\bigcup_\ell \bigcup_H \{Y \in \mathrm{Func}(n+1) : Y \text{ extends } X, F \in \mathrm{1TAUT}[Y] \text{ and} \\
&\qquad \mathrm{Card}(\mathrm{dom}(S_F)) > \ell^k + C\},
\end{aligned} \tag{4.2}$$

where ℓ varies over numbers such that $n+2 \leq \ell \leq (2N)^{1/k}$; for each ℓ, H varies over all query free formulas such that $\ell(F) = \ell$.

Now, let ℓ be a natural number such that $n+2 \leq \ell \leq (2N)^{1/k}$. And, let H be a query free formula such that $\ell(F) = \ell$. We want to count the cardinality of the following set (with respect to the fixed ℓ and H).

$$\{Y \in \mathrm{Func}(n+1) : Y \text{ extends } X, F \in 1\mathrm{TAUT}[Y] \text{ and } \mathrm{Card}(\mathrm{dom}(S_F)) > \ell^k + C\}. \tag{4.3}$$

Note that by the standard order-isomorphism from $\mathrm{Str}(n+1)$ to $\{0,1\}^{n+1}$, $\mathrm{Str}(n)$ is mapped to $\{0u : u \in \{0,1\}^n\}$. And, the complement $\mathrm{Str}(n+1) \setminus \mathrm{Str}(n)$ is mapped to $\{1u : u \in \{0,1\}^n\}$.

Let S_0 be the restriction of S_F to $\mathrm{Str}(n)$. That is, the domain of S_0 is $\mathrm{dom}(S_F) \cap \mathrm{Str}(n)$. In a similar way, let S_1 be the restriction of S_F to $\mathrm{Str}(n+1) \setminus \mathrm{Str}(n)$.

Now, suppose that Y is a member of (4.3). Regarding the restriction of Y to $\mathrm{Str}(n)$, we have no choice; it is X. Since F belongs to $1\mathrm{TAUT}[Y]$, Y extends S_F. Thus Y is an extension of S_1. We investigate the size of the domain of S_1.

Let F' be the following.

$$\left(a^{(1)} \Leftrightarrow \xi^n(q_2^{(1)}, \ldots, q_n^{(1)}, q_{n+1}^{(1)})\right) \Rightarrow H[0/q_1^{(1)}].$$

F' is equivalent to $F[0/q_1^{(1)}]$. Since the restriction of Y to $\mathrm{Str}(n)$ is X, F' belongs to $1\mathrm{TAUT}[X]$. Moreover, it is not hard to see that S_0 is the minimal forcing condition that forces F'. By our assumption, X satisfies $\mathrm{R}(k, C, n)$. Therefore, the size of the domain of S_0 is at most $\ell(F')^k + C$. Thus it is at most $(\ell-1)^k + C$. Hence, we have the following.

$$\begin{aligned}\mathrm{Card}(\mathrm{dom}(S_1)) &= \mathrm{Card}(\mathrm{dom}(S_F)) - \mathrm{Card}(\mathrm{dom}(S_0)) \\ &> (\ell^k + C) - [(\ell-1)^k + C] = \ell^k - (\ell-1)^k.\end{aligned}$$

In summary, every member Y of the set (4.3) (with respect to the fixed ℓ and H) is an extension of X and S_1. The domain of X and that of S_1 are disjoint, and the size of the former is N and the size of the latter satisfies the above inequality. Therefore, at most

$$2N - N - [\ell^k - (\ell-1)^k] = N - [\ell^k - (\ell-1)^k]$$

bits are saved for each member Y of (4.3). Hence, we have the following.

$$\begin{aligned}&(\text{The cardinality of (4.3)}) \leq 2^{[\cdots]}, \\ &\text{where} \quad [\cdots] = N - [\ell^k - (\ell-1)^k].\end{aligned} \tag{4.4}$$

Next, for each ℓ, we investigate the number of query free formulas H such that $\ell(F) = \ell$.

Claim 2. There is a total function $f : \mathbb{N} \times \mathbb{N} \to \{0,1\}^*$ that satisfies the following three requirements for all $n \in \mathbb{N}$.

- For every positive integer ℓ, $f(n,\ell)$ is (the binary code of) a finite set of 1-query formulas. For every positive integer $\ell \leq n$, $f(n,\ell)$ is the empty set.
- For all but finitely many ℓ, the cardinality of $f(n,\ell)$ is at most $2^{(\ell^2)}$.
- For every positive integer $\ell > n$ and for every 1-query formula F such that the dimension of F is n and $\ell(F) = \ell$, there exists a formula $F' \in f(n,\ell)$ such that for every oracle X, the necessary and sufficient condition for $F \in \mathrm{1TAUT}[X]$ is $F' \in \mathrm{1TAUT}[X]$.

Proof of Claim 2: For given positive integers n and ℓ such that $n < \ell$, we fix a list of ℓ variables $\{a^{(1)}, q_1^{(1)}, \cdots, q_{\ell-1}^{(1)}\}$. Then, we let $f(n,\ell)$ be the set of all 1-query formulas F such that the dimension of F is n, $\ell(F) = \ell$ and F does not have any occurrences of variables which are not one of $a^{(1)}, q_1^{(1)}, \cdots, q_{\ell-1}^{(1)}$. Now, let Γ be the alphabet consisting of $a^{(1)}, q_1^{(1)}, \cdots, q_{\ell-1}^{(1)}$ and all symbols of the relativized propositional calculus which are not variables. The cardinality of $f(n,\ell)$ is at most the cardinality of Γ^ℓ, i.e. the set of all Γ-words of length ℓ. If ℓ is sufficiently large, then the cardinality of Γ is at most 2ℓ, and we have $(2\ell)^\ell \leq 2^{(\ell^2)}$. Hence, f satisfies the three requirements mentioned above. Q.E.D.(**Claim 2**)

Recall that we assumed $2^{n+1} > C$. Thus, n is sufficiently large. Hence, by **Claim 2**, for each $\ell \geq n+2$, we may assume that the number of query free formulas H such that $\ell(F) = \ell$ is at most $2^{(\ell^2)}$. Therefore, by (4.2) and (4.4), the cardinality of (4.1) is at most the following.

$$\sum_{\ell} 2^{N - k\, g(\ell)},$$

where ℓ varies from $n+2$ to $(2N)^{1/k}$, and g is a monic polynomial of degree $k-1$ such that $k\, g(\ell) = \ell^k - (\ell-1)^k - \ell^2$. Since n is sufficiently large, we may also assume that for each ℓ such that $n+2 \leq \ell \leq (2N)^{1/k}$, we have $g(\ell) \geq g(n+2)$. Hence, the cardinality of (4.1) is at most the following.

$$(2N)^{1/k}\, 2^{N - k\, g(n+2)} = 2^N\, 2^{[\cdots]},$$
$$\text{where} \quad [\cdots] = -k\, g(n+2) + (1/k)(n+1).$$

Since n is sufficiently large, we have $-k\, g(n+2) + (1/k)(n+1) \leq -(n+2)$. □

By the above result, Dowd observed that the set of all 1-Dowd oracles has Lebesgue measure one in the Cantor space [Do 92, p.71].

4.3 Proof of Theorem 2

We first consider the special case where the oracle A in the statement of **Theorem 2** is the empty set.

Theorem 3 *There exists a primitive recursive oracle D such that D is* 1*-Dowd.*

By identifying an oracle A with the infinite binary sequence

$$A(z(0))A(z(1))A(z(2))\cdots,$$

we can define lexicographic order on the class of all oracles. That is, for two oracles A and B, the definition of $A < B$ is that $A \neq B$ and, letting m be the least number such that $A(z(m)) \neq B(z(m))$, we have $A(z(m)) = 0$ and $B(z(m)) = 1$. As is well known, this order is not a well-ordering. For example, for each natural number n, let A_n be the infinite sequence $0^n111\cdots$; then, the class $\{A_n : n \in \mathbb{N}\}$ forms an infinite descending sequence and does not have a least element. However, it is possible for *some particular* class of 1-Dowd oracles, to have a least element: this is the key to the solution. In the rest of this subsection, lexicographic order of oracles denotes the above order. For each natural number n, we introduce lexicographic order of $\mathrm{Func}(n)$ in the same way.

Proof of Theorem 3 :

Construction.

We fix an ordered pair (k, C) of sufficiently large integers. By **Claim 1** in the proof of **Fact 1**, every member of $\mathrm{Func}(1)$ satisfies $\mathrm{R}(k, C, 1)$. Let D_1 be the least element (with respect to lexicographic order) of $\mathrm{Func}(1)$. Next, suppose that n is a natural number and we have defined $D_n \in \mathrm{Func}(n)$. Suppose that D_n satisfies $\mathrm{R}(k, C, n)$. By **Fact 1**, D_n has an extension in $\mathrm{Func}(n+1)$ that satisfies $\mathrm{R}(k, C, n+1)$. Let D_{n+1} be the least element (with respect to lexicographic order) of $\mathrm{Func}(n+1)$ such that D_{n+1} extends D_n and D_{n+1} satisfies $\mathrm{R}(k, C, n+1)$. Let D be the union of all D_n's (for $n = 1, 2, 3, \ldots$). Thus, for every n, $D \upharpoonright \mathrm{Str}(n)$ is D_n.

Verification.

For every $n \geq 1$, D_n satisfies $\mathrm{R}(k, C, n)$. Therefore, D is 1-Dowd. In the following, we verify that D is primitive recursive.

Suppose n is a natural number and X is a member of $\mathrm{Func}(n)$. First, note that X satisfies $\mathrm{R}(k, C, n)$ if and only if the following holds, where N denotes 2^n and $f(n, \ell)$ is the set of formulas introduced in **Claim 2** in the proof of **Fact 1**.

"For every ℓ such that $n + 2 \leq \ell \leq (2N)^{1/k}$ and for every $F \in f(n, \ell)$, if F belongs to $1\mathrm{TAUT}[X]$ then we have $\mathrm{Card}(\mathrm{dom}(S_F)) \leq \ell^k + C$."

Second, X equals D_n if and only if X is the least member of $\mathrm{Func}(n)$ (with respect to lexicographic order) that satisfies the following.

$$\text{"}1 \leq \forall m \leq n \ (X \upharpoonright \mathrm{Str}(m) \text{ satisfies } \mathrm{R}(k, C, m))\text{"} \tag{4.5}$$

Now, for a bit string u, we can compute $D(u)$ by the following algorithm.

Algorithm for D (**input** u: bit string);
begin

1. Let n be the least n such that $u \in \mathrm{Str}(n)$.
2. In accordance with lexicographic order, for every $X \in \mathrm{Func}(n)$, check whether (4.5) holds, until we find D_n.
3. Output $D_n(u)$.

end

Hence, it is easy to see that D is primitive recursive. □

Now, we are ready to show **Theorem 2**. The proof is given as a "relativization" of the proof of **Theorem 3**. For each oracle A, we shall construct a 1-Dowd oracle D^A. If A is the empty set, then the resulting oracle $D^\emptyset$ shall agree with D in the proof of **Theorem 3**.

Proof of Theorem 2 : We fix an ordered pair (k, C) of sufficiently large integers. Suppose A is a given oracle. We define an oracle D^A by inductively defining its initial segments on $\mathrm{Str}(n)$.

Stage 1. We define D_1^A as $A \restriction \mathrm{Str}(1)$. By **Claim 1** in the proof of **Fact 1**, D_1^A satisfies $\mathrm{R}(k, C, 1)$.

Stage $n+1$. Suppose that we have defined $D_n^A \in \mathrm{Func}(n)$ and D_n^A satisfies $\mathrm{R}(k, C, n)$. If $A(z(n+1)) = 0$, then we define D_{n+1}^A as the least X (with respect to lexicographic order) that satisfies the following.

$$X \in \mathrm{Func}(n+1), X \text{ extends } D_n^A \text{ and } X \text{ satisfies } \mathrm{R}(k, C, n+1). \tag{4.6}$$

Otherwise (that is, $A(z(n+1)) = 1$). Note that, by **Fact 1**, there are at least *two* X that satisfy (4.6). We define D_{n+1}^A as the second least X (with respect to lexicographic order) that satisfies (4.6).

Let D^A be the union of all D_n^A's ($n = 1, 2, 3 \cdots$). For every $n \geq 1$, D_n^A satisfies $\mathrm{R}(k, C, n)$. Thus, D^A is 1-Dowd. For each positive integer n, we can effectively compute D_n^A from $A \restriction \{z(0), z(1), \ldots, z(n)\}$, and vice versa. Hence, D^A and A are Turing-equivalent. Thus, we have shown **Theorem 2**. □

We conclude this section by a remark on the algorithm for D in the proof of **Theorem 3**. It is not hard to see that the deterministic space-complexity of D is at most exponential in a polynomial of $|u|$. Thus, there is a 1-Dowd oracle in the following complexity class.

$$\mathrm{EXPSPACE} =_{\mathrm{def.}} \mathrm{DSPACE}(2^{\mathrm{poly}}).$$

5 Complexity of 1-Dowd oracles

Although we have seen the existence of a primitive recursive 1-Dowd oracle, it is impossible to replace "primitive recursive" by "polynomial time computable."

In this section, we present basic examples of classes which do not contain any 1-Dowd oracles.

An oracle A is called *sparse* if there exists a polynomial p such that for each positive integer n, $\mathrm{Card}(A \cap \{0,1\}^{\leq n}) \leq p(n)$.

Example 2. No 1-Dowd oracle is sparse.

Proof: The argument is similar to **Example 1**. Assume for a contradiction that there exists a sparse oracle D with respect to a polynomial p such that D is 1-Dowd. Let n be a positive integer and let $\{u^{(1)}, \ldots, u^{(c(n))}\}$ be the list of all bit strings u of length n such that $D^n(u) = 1$, where $c(n) \leq p(n)$. Instead of F_n in **Example 1**, consider the following formula F'_n.

$$(1 \Leftrightarrow \xi^n(q_1, \ldots, q_n)) \Leftrightarrow [\text{ the bit string } q_1 \cdots q_n \text{ is one of } u^{(1)}, \ldots, u^{(c(n))}]$$

$[\cdots]$ can be obviously replaced by an appropriate formula of the propositional calculus. If n is sufficiently large, then it is impossible to force F'_n by a small forcing condition, and we get a contradiction. □

Example 3. Neither NP nor coNP contains any 1-Dowd oracle.

Proof: As is stated in the beginning of **Citation 1**, the class of r-Dowd oracles is closed under complementation; this fact is very easily verified. Thus, it is sufficient to show that no 1-Dowd oracle belongs to NP. Assume for a contradiction that D is a 1-Dowd oracle and D belongs to NP. Then, the following set is also in NP.

$$\{u \in \{0,1\}^* : D^{|u|}(u) = 1\}.$$

Therefore, there exists a non-deterministic Turing machine (without oracle) M and a polynomial p such that the following four requirements are satisfied.

1. D is a 1-Dowd oracle with respect to p.
2. M halts for every input u within $p(|u|)$ steps.
3. For every bit string u, $D^{|u|}(u) = 1$ if and only if there exists an accepting computation path of M with input u.
4. p is monotone-increasing.

Fix a sufficiently large constant C. We define a polynomial f as follows: $f(x) =_{\mathrm{def.}} Cx^C + C$. Let n be a natural number such that 2^n is larger than $p(f(nm))$, where $m =_{\mathrm{def.}} f(p(n))$. Then, the following assertion (Φ_n) is true for every truth assignment for free variables $q_i^{(j)}$'s, if we interpret ξ^n as D^n.

(Φ_n): " If $q_0^{(1)} = \xi^n(q_1^{(1)} \cdots q_n^{(1)})$ and $q_1^{(2)} \cdots q_m^{(2)}$ is the binary code of an accepting computation path of M with input $q_1^{(1)} \cdots q_n^{(1)}$, then $q_0^{(1)} = 1$. "

We can easily interpret (Φ_n) as a 1-query formula of length at most $f(nm)$. Since D is 1-Dowd, there exists a subset $E \subsetneqq \{0,1\}^n$ of size at most $p(f(nm))$

such that $D^n \upharpoonright E$ forces (Φ_n) (recall the convention of abuse of the terminology "force" in the last paragraph of the **Notation** section). But, for every bit string $u \in \{0,1\}^n \setminus E$, we can extend $D^n \upharpoonright E$ to an n-ary Boolean function A^n so that $A^n(u) = 0$. Hence, for every bit string $u \in \{0,1\}^n \setminus E$, there does not exist an accepting computation path of M with input u. Therefore, for such a u, $D^n(u)$ should be 0, because requirement 3 is satisfied. The remainder of the proof is similar to **Example 2**. □

Circuit complexity and 1-Dowd oracles. The class P/poly is defined as follows. An oracle A belongs to P/poly if there exists a set $B \in \mathrm{P}$ and a function $f : \mathbb{N} \to \{0,1\}^*$ satisfying the following two requirements:

- For every bit string u, $u \in A$ if and only if $\langle u, f(|u|)\rangle \in B$.
- There exists a polynomial p such that for each n, $|f(n)| \leq p(n)$.

It is well-known that the class P/poly coincides with the class of all oracles that have polynomial size circuits [BDG 95, Chapter 5]. In the following, we summarize the relationship between P/poly and 1-Dowd oracles. The following example is a variation of [Do 92, Theorem 17]. Note that this implies **Example 2**.

Example 4. No 1-Dowd oracle has polynomial size circuits.

Proof : Assume for a contradiction that D is a 1-Dowd oracle and D has polynomial size circuits. Thus, for each n, the partial function $D \upharpoonright \{0,1\}^n$ is computed by a circuit of size polynomial in n. A routine argument shows that, for each n, the n-ary Boolean function D^n is also computed by a circuit C_n of size polynomial in n. Then, consider the following assertion (Φ'_n).

(Φ'_n): " If $q_0^{(1)} = \xi^n(q_1^{(1)} \cdots q_n^{(1)})$ and C_n accepts the input $q_1^{(1)} \cdots q_n^{(1)}$, then $q_0^{(1)} = 1$. "

The remainder of the proof is similar to **Example 3**. □

However, the class of all 1-Dowd oracles is not the complement of P/poly. That is, there exists an oracle which is neither a 1-Dowd oracle nor a member of P/poly. This can be observed by the following example, which is a variation of [Do 92, Theorem 16].

Example 5. No generic oracle (of arithmetical forcing) has polynomial size circuits.

Proof : By the result of Schöning [BDG 95, Theorem 5.27], there exists an oracle A_0 such that for all but finitely many n, $A_0 \upharpoonright \{0,1\}^n$ cannot be computed by any circuit of size at most $n^{\log n}$. Now, let S be an arbitrary forcing condition. S can be extended to an oracle B such that $B(u) = A_0(u)$ for all but finitely many bit strings u. Therefore, S can be extended to a forcing condition T that forces the following arithmetical sentence (Φ).

(Φ): "For some $n \in \mathbb{N}$, there is no circuit C_n of size at most $n^{\log n}$ such that C_n computes the finite portion of the oracle restricted to $\{0,1\}^n$."

Hence, (Φ) holds for every generic oracle. □

It is known that no generic oracle is 1-Dowd [Do 92, Theorem 12]. Consequently, the following three classes are mutually disjoint: the class of all 1-Dowd oracles, the class of all generic oracles, and P/*poly*. The first is measure-one and meager, the second is measure-zero and comeager, and the last is measure-zero and meager.

And, since the computational complexity class BPP is a subclass of P/poly [BDG 95, Chapter 6], no 1-Dowd oracle belongs to BPP.

6 Fragility of 2-Dowd property

Polynomial time many-one-degrees of 1-Dowd oracles also have an interesting property. In this section, we show the following.

Theorem 4 (**The Fragility Theorem on 2-Dowd Property**) *For every 1-Dowd oracle D, there exists an oracle E such that E is polynomial time many-one-equivalent to D, E is also 1-Dowd but E is not 2-Dowd.*

In order to prove the above theorem, we investigate the closure property of the class of all 1-Dowd oracles.

We first discuss an easy example. Suppose that p is a polynomial, and both D_0 and D_1 are 1-Dowd oracles with respect to p. Suppose also that n is a positive integer and E is an oracle such that for all $u \in \{0,1\}^n$, we have $E^{n+1}(0u) = (D_0)^n(u)$ and $E^{n+1}(1u) = (D_1)^n(u)$. Assume that F is of the form $\left(a^{(1)} \Leftrightarrow \xi^{n+1}(q_1^{(1)}, \ldots, q_{n+1}^{(1)})\right) \Rightarrow H$, where H is a query free formula, and that F is a tautology with respect to E. For each $i \in \{0,1\}$, let F_i be $F[i/q_1^{(1)}]$. For each $i \in \{0,1\}$, it is easy to see the following: since D_i is 1-Dowd, $E^{n+1} \restriction \{iu : u \in \{0,1\}^n\}$ has a finite portion S_i such that the size of the domain of S_i is at most $p(|F|)$ and S_i forces F_i. Therefore, if we let S be the union of S_0 and S_1, then S is a finite portion of E, the size of the domain of S is at most $2p(|F|)$ and S forces F.

By a similar (but technically more complicated) argument, we get the following.

Lemma 5 *Suppose that D_0 and D_1 are 1-Dowd oracles. Then, their join*

$$D_0 \oplus D_1 = \{z(2n) : z(n) \in D_0\} \cup \{z(2n+1) : z(n) \in D_1\}$$

is also 1-Dowd. □

Complexity theorists and recursion theorists may have different usages of the join. For example, in [BDG 95], the join of two oracles A and B is defined

as $\{u0 : u \in A\} \cup \{v1 : v \in B\}$. However, **Lemma 5** holds for this join, too; the verification of this fact and **Lemma 5** is left to the reader.

Proof of Theorem 4 : Suppose D is a 1-Dowd oracle. Let $E =_{\mathrm{def.}} D \oplus D$. By **Lemma 5**, E is also 1-Dowd. It is clear that D and E are polynomial time many-one-equivalent.

It is enough to show that E is not 2-Dowd. For each n, let F_n be the following formula.

$$\xi^{n+1}(q_1^{(1)}, \ldots, q_n^{(1)}, 0) \Leftrightarrow \xi^{n+1}(q_1^{(1)}, \ldots, q_n^{(1)}, 1).$$

Of course, for each n, there exists a 2-query formula F_n' in the sense of our formal definition such that F_n' is equivalent to F_n and the length of F_n' is at most polynomial in the length of F_n. For each n, F_n is a tautology with respect to E. However, it is impossible to force F_n by a forcing condition of size at most polynomial in the length of F_n. Hence, E is not 2-Dowd. □

Thus, in contrast to 1-Dowd oracles, the class of all 2-Dowd oracles is not closed under the join-operator.

7 One-query tautologies and the jump-operator

In the remaining part of this paper, we present several problems that we left open, and we explain their background. We begin with problems relating to **Lemma 1** and complexity of 1-Dowd oracles.

Although an r-Dowd oracle is not a generic oracle, there is a resemblance between them. Again, let $\emptyset$ be the characteristic function of the empty set. Let n be a positive integer. If G is n-generic (i.e. Cohen-generic for n-quantifier arithmetic), then we have:

$$G^{(n)} \equiv_T G \oplus \emptyset^{(n)}. \tag{7.1}$$

That is, the nth jump of G is Turing-equivalent to the join of G and the nth jump of the empty set. A proof of this well-known result can be found in [Jo 80, Lemma 2.6]; note that the proof requires the fact that we have $\Sigma_1^0[X] \cap \Pi_1^0[X] = \Sigma_0^0[X]$ for every oracle X. On the other hand, regarding the relativized polynomial hierarchy, it is not obvious at all whether we have $\mathrm{NP}[X] \cap \mathrm{coNP}[X] = \mathrm{P}[X]$ for a given oracle X. Thus, a minor variation of the proof of [Jo 80, Lemma 2.6] does not produce an analogous result for r-Dowd oracles. However, in [Su 98], it was shown that for every $r \geq 1$, if D is r-Dowd, then we have:

$$r\mathrm{TAUT}[D] \equiv_T^P D \oplus \mathrm{TAUT}. \tag{7.2}$$

That is, $r\mathrm{TAUT}[D]$ is polynomial time Turing-equivalent to the join of D and TAUT. The reader may think it is strange that the right-hand side of (7.2) does not depend on r. However, it is easy to see that, for each r, $r\mathrm{TAUT}[\emptyset]$ and TAUT are polynomial time Turing-equivalent. Thus, (7.2) means the following.

$$r\mathrm{TAUT}[D] \equiv_T^P D \oplus r\mathrm{TAUT}[\emptyset].$$

For the simplest case of $r = 1$, the proof of (7.2) is given by the proof of **Lemma 1**. By using (7.2), the following was also shown in [Su 98].

Fact 2 [Su 98] *For each r, the following two assertions are equivalent.*

- *If A is a random oracle then rTAUT$[A]$ is not polynomial time Turing-reducible to A with probability one*: *in other words, the set of all oracles A of the above property forms a measure-one subset in the Cantor space.*
- *The unrelativized computational complexity classes* R *and* NP *are not identical.*

□

R is a well-known complexity class such that $\mathrm{P} \subseteq \mathrm{R} \subseteq \mathrm{NP}$ and $\mathrm{R} \subseteq \mathrm{BPP}$. The definition of R is given as follows [BDG 95]. A set A of bit strings belongs to R if and only if there exists a positive constant $\varepsilon < 1/2$ and a probabilistic Turing machine M clocked by a polynomial such that every bit string in A is accepted by M with probability more than $1/2 + \varepsilon$ and every bit string not in A is rejected by M with probability one.

The author left the following problems open.

Problem 1. Under the assumption of $\mathrm{P} \neq \mathrm{NP}$, can we prove the existence of a 1-Dowd oracle D such that 1TAUT$[D]$ does not P-tt-reduce to D?

Problem 2. Under the assumption that there exists a 1-Dowd oracle D such that 1TAUT$[D]$ does not P-T-reduce to D, can we prove $\mathrm{R} \neq \mathrm{NP}$?

The following problem would be more important. As we reviewed in **Introduction**, neither Σ_1^0 nor Π_1^0 contains any 1-generic oracle, but there exists a 1-generic oracle in Δ_2^0 $(= \Sigma_0^0[\emptyset'])$ [Jo 80]. Whereas, **Example 3** shows that neither Σ_1^P nor Π_1^P contains any 1-Dowd oracle. What about Δ_2^P $(= \mathrm{P[TAUT]})$?

Problem 3. Is there a 1-Dowd oracle in Δ_2^P ?

Natural variations of **Problem 3** are obtained by replacing Δ_2^P by PSPACE or EXPTIME $(= \mathrm{DTIME}(2^{\mathrm{poly}}))$.

The relationship of the complexity classes mentioned above shall be summarized as diagrams in Section 10. In Section 9, we shall again discuss whether there are more analogies between the jump-operator and the operation of taking the set of 1-query tautologies.

8 The Fragility Theorem and generic separation

In the case of $r \geq 2$, the r-Dowd property is more delicate than the 1-Dowd property (cf. [Su (b)]). We have to take caution against a rough argument merely being based on intuition. Regarding the Ubiquity Theorem, the author has not succeeded to solve the following yet.

Problem 4. For each $r \geq 2$, does the Ubiquity Theorem (**Theorem 2**) hold for r-Dowd instead of 1-Dowd?

The essence of the difficulty of **Problem 4** is as follows. Suppose $r \geq 2$. Suppose also that k and C are sufficiently large natural numbers. For all natural number n, as we did in the case of $r = 1$ (Subsection 4.2), we can naturally introduce a requirement $\mathrm{R}(r, k, C, n)$ for construction of an r-Dowd oracle. Then, for every forcing condition $X \in \mathrm{Func}(n)$ that satisfy $\mathrm{R}(r, k, C, n)$, does X have two different extensions $Y_1, Y_2 \in \mathrm{Func}(n+1)$ such that both of Y_1 and Y_2 satisfy $\mathrm{R}(r, k, C, n+1)$? Note that, for given X_1 and X_2 satisfying $\mathrm{R}(r, k, C, n)$, their concatenation $X_1{}^\frown X_2$ does not necessarily satisfy $\mathrm{R}(r, k, C, n+1)$. The reason is similar to the proof of the Fragility Theorem (**Theorem 4**).

In the following, we introduce one more problem relating to the Fragility Theorem. In order to explain the background of the problem, we review the relationship between *forcing complexity* (i.e. the minimal size of a forcing condition that forces a given arithmetical predicate) and separation of complexity classes relative to a generic oracle (of arithmetical forcing).

Poizat [Po 86] observed that if there exists an oracle which separates two given relativized computational complexity classes, then (under certain assumptions), every generic oracle separates the two complexity classes. This idea was extended to type-2 complexity in [CIY 97]. And, it was independently investigated in [Su (a)] that forcing complexity also has "a tight relationship" with generic oracles. Roughly speaking, for given two (type-2) predicates $\varphi(X)(y)$ and $\psi(X)(y)$, if we can find an oracle G_1 such that G_1 has small forcing complexity for φ and G_1 has large forcing complexity for ψ, then (under certain assumptions) the corresponding relativized languages $\varphi[G_2]$ and $\psi[G_2]$ relative to a generic oracle G_2 are separated in the sense of complexity. This method is quite helpful in the case where it is difficult to find an example of an oracle A such that $\psi[A] \notin \mathrm{P}[\varphi[A]]$.

In order to review the separation theorem in [Su (a)], let us review basic concepts closely related to Poizat's observation. Let y be a variable for a bit string of finite length, and $X(\sim)$ a unary predicate denoting membership in a given oracle. Suppose $\varphi(X)(y)$ is an arithmetical predicate. φ is called *finitely testable* (or, *test fini* [Po 86]) if there exists a function $f : \mathbb{N} \to \mathbb{N}$ of the following property: for every oracle A, for every positive integer n and for every bit string u of length n, the necessary and sufficient condition for $\varphi(A)(u)$ is $\varphi(B)(u)$, where B is the oracle such that $B(v) = A(v)$ for every bit string v of length at most $f(n)$, and $B(w) = 0$ for every bit string w of length longer than $f(n)$. Suppose $\varphi(X)(y)$ is a finitely testable arithmetical predicate and A is an oracle. $\varphi[A]$ denotes the set of all bit strings u such that we have $\varphi(A)(u)$.

In [Su (a)], by extending the method of [Po 86] and that of [Do 92], the following fact was shown.

Fact 3 [Su (a)] *Suppose that G is a generic oracle. Then for each positive integer r, we have the following.*

$$\mathrm{TAUT}[G] \notin \mathrm{NP}[r\mathrm{TAUT}[G]].$$

□

The above fact was obtained as a special case of a general theorem about forcing complexity. An oracle A is called a *ceiling-generic oracle for* φ [Su (a)] if there exists a polynomial p of the following property: for every positive integer n and for every bit string u of length n, if we have $\varphi(A)(u)$, then there exists a forcing condition S such that S is a finite portion of A, S forces $\varphi(X)(u)$ (i.e., for every oracle B extending S, we have $\varphi(B)(u)$) and the cardinality of (the domain of) S is at most $p(n)$.

Fact 4 [Su (a)] *Suppose that $\varphi(X)(y)$ and $\psi(X)(y)$ are finitely testable arithmetical predicates and G_1 is an oracle. Suppose that for every oracle A, if $A(u) = G_1(u)$ for all but finitely many bit strings u, then the following three hypotheses hold.*

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

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

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

Then, for every generic oracle G_2, we have the following.

$$\psi[G_2] \notin \mathrm{NP}[\varphi[G_2]].$$

□

Now, the class of all r-Dowd oracles is non-empty and closed under finite changes. Moreover, by the Fragility Theorem, there exists an oracle D such that D is 1-Dowd but not 2-Dowd. Hence, by the exactly same argument as in [Su (a)], we know that **Fact 3** holds for "$2\mathrm{TAUT}[G] \notin \mathrm{NP}[1\mathrm{TAUT}[G]]$" in place of "$\mathrm{TAUT}[G] \notin \mathrm{NP}[r\mathrm{TAUT}[G]]$."

Thus, separation of hierarchy with respect to forcing complexity closely relates to the structure of $\mathrm{coNP}[G]$ relative to a generic oracle G. Examples of collapse of hierarchy with respect to forcing complexity can be found in [Su (b)].

Our strategy of the proof of the Fragility Theorem seems useless to get "fragility of 3-Dowd property relative to 2-Dowd property." We left the following as an open problem.

Problem 5. Is the following assertion true for every positive integer $r \geq 2$? "For every r-Dowd oracle D, there exists an oracle E such that E is polynomial time many-one-equivalent to D, E is also r-Dowd but E is not $(r+1)$-Dowd."

9 One-query tautologies and random oracles

There are many classical results about the relationship between random oracles and the polynomial hierarchy. It is known that for every oracle A, the necessary and sufficient condition for $A \in \mathrm{BPP}$ is that the following class has Lebesgue measure one [BG 81]: $\{X : A \leq_T^P X\}$. Therefore, "NP $\not\subseteq$ BPP" is equivalent to "$\{X : \mathrm{TAUT} \leq_T^P X\}$ has Lebesgue measure zero." It is also known that "NP $\not\subseteq$ BPP" is equivalent to "R $\neq$ NP" [Ko 82]. In [Su 98], **Fact 2** (Section

7) was shown by using these facts and (7.2). It is also known that the necessary and sufficient condition for $A \in \mathrm{P}$ is that the following class has Lebesgue measure one [Am 86]: $\{X : A \leq_m^P X\}$. Thus, "P $\neq$ NP" is equivalent to "$\{X : \mathrm{TAUT} \leq_m^P X\}$ has Lebesgue measure zero."

In this section, we extend our investigation in Section 3 and Section 7, and discuss whether there are more analogies between the jump operator and the operation of taking the set of 1-query tautologies. Suppose $\leq_X$ is a concept of reducibility, such as $\leq_T^P$ or $\leq_{tt}^P$. We examine the nature of the following assertion, and call it 1-*query-jump hypothesis for the reducibility* $\leq_X$.

1-query-jump hypothesis for $\leq_X$. "The class of all oracles A of the following property has Lebesgue measure one: 1TAUT$[A]$ does not X-reduce to A."

We emphasize that we talk about Lebesgue measure, not about category. If we substitute "comeager" for "Lebesgue measure one," then the above assertion for polynomial time Turing-reducibility is a known fact. It is easy to see that for every generic oracle G (of arithmetical forcing), 1TAUT$[G]$ is not polynomial time Turing reducible to G: the proof is a paraphrase of the argument in [BGS 75] (see [Su (a)] for an alternative proof using **Fact 4**). And, the class of all generic oracles is comeager.

Recall the definitions of *disjunctive reducibility* (d-reducibility, $\leq_d$), *conjunctive reducibility* (c-reducibility, $\leq_c$) and 1-*question truth-table reducibility* (1-tt-reducibility, $\leq_{1-tt}$). An oracle A is d-reducible (c-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 such as $x_1 \vee \cdots \vee x_n$ (a conjunctive formula such as $x_1 \wedge \cdots \wedge x_n$, a formula of norm 1 such as x_1 or such as $\neg x_1$, respectively), where each x_i is an atomic tt-condition. For a more formal treatment, see [Od 89, p.268]. In [LLS 75], *Polynomial time disjunctive reducibility* (P-d-reducibility, $\leq_d^P$), *polynomial time conjunctive reducibility* (P-c-reducibility, $\leq_c^P$) and *polynomial time* 1-*question truth-table reducibility* (P-1-tt-reducibility, $\leq_{1-tt}^P$) are introduced.

The following diagram illustrates the obvious relationships between these reducibilities, where $\leq_X \rightarrow \leq_Y$ denotes that for all oracles A and B, $A \leq_X B$ implies $A \leq_Y B$.

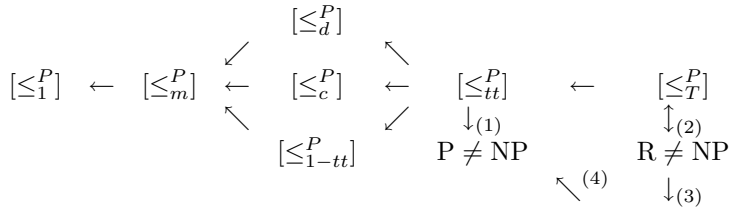
Diagram 9a. (Reducibilities)

$$\begin{array}{ccccccccccc} & & & & & \leq_d^P & & & & & \\ & & & \nearrow & & & \searrow & & & & \\ \leq_1^P & \rightarrow & \leq_m^P & \rightarrow & & \leq_c^P & & \rightarrow & \leq_{tt}^P & \rightarrow & \leq_T^P \\ & & & \searrow & & & & \nearrow & & & \\ & & & & & \leq_{1-tt}^P & & & & & \end{array}$$

The following diagram shows the relationship between 1-query jump hypotheses and famous open problems about unrelativized complexity classes. For

the time being, we introduce symbol $[\leq_X]$ as follows. For each reducibility $\leq_X$, $[\leq_X]$ denotes the corresponding 1-query-jump hypothesis. PRIMES denotes the set of all (binary codes of) prime numbers.

Diagram 9b. (1-query-jump hypotheses)



"PRIMES is not NP-complete"

The implication (1) holds because of **Lemma 1** and the fact that the class of all 1-Dowd oracles has Lebesgue measure one [Do 92, Theorem 10]. The equivalence (2) is the special case ($r = 1$) of **Fact 2**. The implications (3) and (4) are classical results. By Rabin's algorithm [BDG 95, section 6.1], PRIMES belongs to R. Moreover, R is closed under P-m-reducibility and we have P $\subseteq$ R $\subseteq$ NP. Therefore, the implications (3) and (4) hold.

However, some of the assertions in Diagram 9b certainly hold (more rigorously speaking, we can prove some of the assertions in Diagram 9b in Zermelo-Fraenkel set theory with the axiom of choice, the standard system of set theory). We observe the following.

Example 6.

1. If A is a random oracle then 1TAUT$[A]$ does not d-reduce to A with probability one.
2. If A is a random oracle then 1TAUT$[A]$ does not c-reduce to A with probability one.
3. If A is a random oracle then 1TAUT$[A]$ does not 1-tt-reduce to A with probability one.

In [Su 99], we showed the assertions 1 and 3 of **Example 6** by somewhat wordy arguments. We owe the following concise proof to the anonymous referee. In the proof of **Example 6**, for the simplicity, we regard oracles as functions from $\mathbb{N}$ to $\{0, 1\}$.

Proof of Example 6 : It is enough to show the following assertions, which would be folklore results.

(a) If A is a random oracle then the complement $\overline{A}$ does not d-reduce to A with probability one.

(b) If A is a random oracle then $\overline{A}$ does not c-reduce to A with probability one.

(c) If A is a random oracle then $A \times A$ does not 1-tt-reduce to A with probability one.

Proof of (a): If $\overline{A}$ disjunctively reduces to A then there is a recursive function $f : \mathbb{N} \to \mathbb{N}$ such that:

$$\forall x \in \mathbb{N}\,[x \notin A \Leftrightarrow D_{f(x)} \cap A \neq \emptyset], \tag{9.1}$$

where $\langle D_x : x \in \mathbb{N}\rangle$ is a canonical enumeration of finite sets.

Now fix a recursive function f. We consider Lebesgue measure of the class of all oracles A such that (9.1) (with respect to the fixed f) holds. There are two cases.

Case a-1: The case where there are infinitely many x such that $x \in D_{f(x)}$. Note that if (9.1) holds for an oracle A then for every $x \in D_{f(x)}$, we have $x \notin A$. Therefore, the class $\{A : A \text{ satisfy (9.1) with respect to } f\}$ has Lebesgue measure zero.

Case a-2: Otherwise. For all but finitely many x, we have $x \notin D_{f(x)}$. We may assume that for every natural number y, there is $x > y$ such that $D_{f(x)}$ contains a number larger than y; because, otherwise, all A satisfying (9.1) (with respect to the fixed f) are recursive. Then there are sequences of natural numbers $\langle x^{(i)} : i \in \mathbb{N}\rangle$ and $\langle y^{(i)} : i \in \mathbb{N}\rangle$ such that the following holds for every $i \in \mathbb{N}$:

$$y^{(i)} < x^{(i)},\ x^{(i)} \notin D_{f(x^{(i)})},\ D_{f(x^{(i)})} \text{ contains a number larger than } y^{(i)},$$
$$x^{(i)} < y^{(i+1)} \text{ and } \forall z \in D_{f(x^{(i)})}\ z < y^{(i+1)}.$$

Now suppose that i is a natural number and we randomly choose a function g from $\{y^{(i)}, y^{(i)}+1, \ldots, y^{(i+1)}-1\}$ to $\{0,1\}$. Then with probability at least $1/2$, we have:

$$\exists z \in D_{f(x^{(i)})}\,[y^{(i)} < z < y^{(i+1)} \text{ and } g(z) = 1]. \tag{9.2}$$

Thus, with probability at least $1/4$, we have (9.2) and $g(x^{(i)}) = 1$, and in this case there is no oracle A such that A extends g and A satisfies (9.1) with respect to f. Therefore, for a randomly chosen g, we have:

$$\text{Prob}\{\exists A\,[A \text{ extends } g \text{ and } A \text{ satisfies (9.1) with respect to } f]\} \leq \frac{3}{4}.$$

Since this holds for all i, $\{A : A \text{ satisfy (9.1) with respect to } f\}$ has Lebesgue measure zero in **Case a-2**, too.

Since there are only countable many recursive functions, the class

$$\{A : A \text{ satisfies (9.1) for some recursive function } f\}$$

has Lebesgue measure zero. Q.E.D.(a)

Proof of (b): The argument is almost same as the proof of (a). Q.E.D.(b)

Proof of (c) (sketch): Fix a recursive function $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ such that for every $(x, y) \in \mathbb{N} \times \mathbb{N}$, $f(x, y)$ is of the form $(\varphi_{(x,y)}, m_{(x,y)})$, where $\varphi_{(x,y)}$ is

(the code of) a function from $\{0,1\}$ to $\{0,1\}$ and $m_{(x,y)}$ is a natural number. We investigate Lebesgue measure of the class of all oracles A of the following property (for this fixed f).

$$\forall (x,y) \in \mathbb{N} \times \mathbb{N}\,[(x,y) \in A \times A \Leftrightarrow \varphi_{(x,y)}(A(m_{(x,y)})) = 1]. \tag{9.3}$$

Case c-1: The case where there exists a natural number z such that $\exists^{\infty} x\, \exists y > x\ (m_{(x,y)} = z)$, where $\exists^{\infty} x$ means "there are infinitely many x such that ...." Fix such z. Then there is a function $\varphi : \{0,1\} \to \{0,1\}$ such that $\exists^{\infty} x\, \exists y > x\ (m_{(x,y)} = z$ and $\varphi_{(x,y)} = \varphi)$. It is easy to see that $\{A : A$ satisfies (9.3) with respect to $f\}$ has Lebesgue measure zero.

Case c-2: Otherwise. The remainder of the proof is not hard. Q.E.D.(c) □

A positive solution for the following implies a positive solution for **Problem 1** (Section 7).

Problem 6. Under the assumption of P $\neq$ NP, can we prove the 1-query-jump hypothesis for polynomial time truth-table-reducibility?

10 Conclusion

In this section, we summarize our results by presenting diagrams.

Unrelativized hierarchy (Sections 4, 5 and 7): As is shown in the Ubiquity Theorem (**Theorem 2**), every Turing degree contains a 1-Dowd oracle. Now, $\Delta^0_2 \cup$ (P/poly) is a measure-zero and meager subclass of the Cantor space. This is a class of "selected minority." The following three diagrams illustrate the distribution of 1-Dowd oracles within this class. In diagrams 10a, 10b and 10c, $C_1 \to C_2$ denotes $C_1 \subseteq C_2$. And, $C_1 \to_{\neq} C_2$ denotes $C_1 \subsetneqq C_2$. The symbol $C+$ ($C-$, respectively) denotes that the class C contains 1-Dowd oracles (does not contain any 1-Dowd oracle, respectively). Whereas, $C\boxplus$ ($C\boxminus$, respectively) denotes that the class C contains a 1-generic oracle of arithmetical forcing (does not contain any 1-generic oracle). PrRec denotes the class of all primitive recursive sets and EXPTIME denotes DTIME(2^{poly}).

Diagram 10a. (Higher levels)

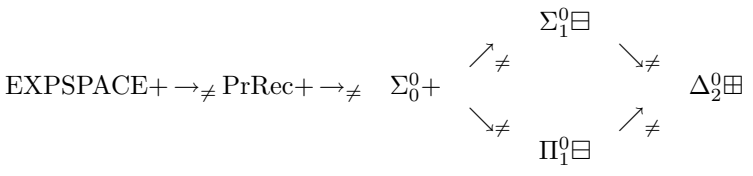


Diagram 10b. (Middle levels)

$$\begin{array}{ccccccc} \Delta_2^P & \rightarrow & \text{PSPACE} & \rightarrow & \text{EXPTIME} & \rightarrow & \text{EXPSPACE+} \\ & \nearrow & & & & & \\ \text{BPP}- & \rightarrow_{\neq} & \text{P/poly}- & & & & \end{array}$$

Diagram 10c. (Lower levels)

$$\begin{array}{ccccccc} & & \text{R}- & \rightarrow & \text{NP}- & & \\ & \nearrow & & \searrow & & \searrow & \\ \text{P}- & & & & \text{BPP}- & & \Delta_2^P \\ & \searrow & & \nearrow & & \nearrow & \\ & & \text{coR}- & \rightarrow & \text{coNP}- & & \end{array}$$

References closely related to Sections 4 and 5 are [Do 92, sections 2,4,6] and [Su (b)].

Generic oracles (Sections 6 and 8): The class of all generic oracles is measure-zero and comeager. For each positive integer r, let rDO be the class of all r-Dowd oracles. Let tGO be the class of all tautology-generic oracles. In addition, let $\mathcal{C}$ be the class of all oracles. By the Fragility Theorem (**Theorem 4**), we have Diagram 10d. In Diagram 10d, 10e and 10f, r is an arbitrary integer such that $r \geq 3$.

Diagram 10d. (Hierarchy with respect to forcing complexity)

$$\mathcal{C} \supsetneqq \text{1DO} \supsetneqq \text{2DO} \supseteq r\text{DO} \supsetneqq t\text{GO} = \emptyset$$

Consequently, by the results in [Su (a)] (**Fact 3** and **Fact 4** in Section 8), for every generic oracle G of arithmetical forcing, we have the following diagram.

Diagram 10e. (Structure of coNP[G] relative to a generic oracle G)

$$\text{TAUT} \oplus G <_T^P \text{1TAUT}[G] <_T^P \text{2TAUT}[G] \leq_T^P r\text{TAUT}[G] <_T^P \text{TAUT}[G]$$

References closely related to Sections 6 and 8 are [Do 92, sections 3,4], [Su (a)] and [Su (b)].

Random oracles (Sections 3, 7 and 9): By the result in [Su 98] (equation (7.2) in section 7) and by the fact that the class of all 1-Dowd oracles has Lebesgue measure one [Do 92, Theorem 10], the class of all oracles A satisfying Diagram 10f is measure-one (and meager). The most right inequality holds by the result in [Kur 83] that if A is a random oracle then we have $\text{TAUT} \oplus A <_T^P \text{TAUT}[A]$ with probability one.

Diagram 10f. (Structure of coNP[A] relative to a random oracle A)

$$\mathrm{TAUT} \oplus A \equiv_T^P \mathrm{1TAUT}[A] \equiv_T^P \mathrm{2TAUT}[A] \equiv_T^P r\mathrm{TAUT}[A] <_T^P \mathrm{TAUT}[A]$$

For each reducibility concept $\leq_X$, we introduced the 1-query-jump hypothesis for $\leq_X$ as the following assertion: if A is a random oracle then 1TAUT[A] does not X-reduce to A with probability one.

In diagram 10g, $[\leq_X]$ denotes the 1-query-jump hypothesis for $\leq_X$. $\vdash [\leq_X]$ means that the 1-query-jump hypothesis for $\leq_X$ is a theorem (of ZFC). PRIMES denotes the set of all (binary codes of) prime numbers.

Diagram 10g. (1-query-jump hypotheses)

$$\begin{array}{ccccccc}
 & & \vdash [\leq_d^P] & & & & \\
\vdash [\leq_1^P] & \vdash [\leq_m^P] & \vdash [\leq_c^P] & [\leq_{tt}^P] & \leftarrow & [\leq_T^P] & \\
 & & & \downarrow & & \updownarrow & \\
 & & \vdash [\leq_{1-tt}^P] & \mathrm{P} \neq \mathrm{NP} & & \mathrm{R} \neq \mathrm{NP} & \\
 & & & & \nwarrow & \downarrow & \\
 & & & & & \text{“PRIMES is not NP-complete”} &
\end{array}$$

References closely related to Sections 3, 7 and 9 are [Do 92, section 4] and [Su 98].

Concluding Words: Although it has been about two decades since *the random oracle hypothesis* [BG 81] was negatively solved [Kur 83], it is still a right idea that there is a close relationship between the polynomial hierarchy and complexity issues relative to random oracles. We hope to bridge over the unrelativized hierarchy, generic oracles and random oracles, by developing further research on degrees and complexity of 1-Dowd oracles.

Acknowledgment

The author would like to thank Martin Dowd for his comment on [Su 98]. It was helpful to improve presentation of the proof of **Lemma 1** of the current paper. He would also like to thank the anonymous referee for many valuable comments. The proof of **Examples 6** is simplified by the referee's suggestion. This research was partially supported by Grant-in-Aid for Scientific Research (No. 09440072, 11740073), Ministry of Education, Science, Sports and Culture of Japan.

References

[Am 86] Ambos-Spies, K. (1986), Randomness, relativizations, and polynomial reducibilities, *in* "Structure in Complexity Theory," Lect. Notes Comput. Sci. vol. 223 (A. L. Selman, Eds.), pp.23-34, Springer, Berlin.

[Am 96] Ambos-Spies, K. (1996), Resource-bounded genericity, *in* "Computability, enumerability, unsolvability," London Math. Soc. Lect. Note Series vol. 224 (S. B. Cooper, T. A. Slaman and S. S. Wainer, Eds.), pp.1-59, Cambridge University Press, Cambridge.

[BGS 75] Baker, T., Gill, J., and Solovay, R. (1975), "Relativizations of the $P = ?NP$ question," *SIAM J. Comput.*, vol. 4, pp.431-442.

[BDG 95] Balcázar, J. L., Díaz J., and Gabarró J. (1995), "Structural complexity I, second edition," Springer, Berlin.

[BG 81] Bennett, C. H. and Gill J. (1981), "Relative to a random oracle A, $P^A \neq NP^A \neq co\text{-}NP^A$ with probability 1," *SIAM J. Comput.*, vol. 10, pp.96-113.

[BI 87] Blum, M. and Impagliazzo R. (1987), Generic oracles and oracle classes, *in* "Proceedings, 28th IEEE Symposium on Foundations of Computer Science," pp.118-126.

[Bu 86] Buss, S. R. (1986), "Bounded arithmetic," Bibliopolis, Naples.

[CIY 97] Cook, S., Impagliazzo R. and Yamakami T. (1997), "A tight relationship between generic oracles and type-2 complexity theory," *Information and Computation*, vol. 137, pp.159-170 (doi:10.1006/inco.1997.2646).

[Do 82] Dowd, M. (1982), "Forcing and the P-hierarchy," Laboratory for Computer Science Research, Rutgers University, Technical Report No. **LCSR-TR-35**.

[Do 92] Dowd, M. (1992), "Generic oracles, uniform machines, and codes," *Information and Computation*, vol. 96, pp.65-76.

[Fe 65] Feferman, S. (1965), "Some applications of the notions of forcing and generic sets," *Fund. Math.*, vol. 56, pp.325-345.

[Hi 69] Hinman, P. G. (1969), "Some applications of forcing to hierarchy problems in arithmetic," *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, vol. 15, pp.341-352.

[Je 78] Jech, T. (1978), "Set theory," Academic Press, New York-London.

[Jo 80] Jockusch, C. G. Jr. (1980), Degrees of generic sets, *in* "Recursion theory: its generalizations and applications," London Math. Soc. Lect. Note Series vol. 45 (F.R. Drake and S.S. Wainer, Eds.), pp.110-139, Cambridge University Press, Cambridge.

[JP 78] Jockusch, C. G. Jr. and Posner, D. (1978), "Double jumps of minimal degrees," *J. Symbolic Logic*, vol. 43, pp.715-724.

[Ko 82] Ko, Ker-I. (1982), Some observations on the probabilistic algorithms and NP-hard problems, *Inform. Process. Lett.* **14** 39-43.

[Kum 96] Kumabe, M. (1996), Degrees of generic sets, *in* "Computability, enumerability, unsolvability," London Math. Soc. Lect. Note Series vol. 224 (S. B. Cooper, T. A. Slaman and S. S. Wainer, Eds.), pp.167-183, Cambridge University Press, Cambridge.

[Kun 80] Kunen, K. (1980), "Set theory," North-Holland, Amsterdam.

[Kur 83] Kurtz, S. A. (1983), "On the random oracle hypothesis," *Inform. Control*, vol. 57, pp.40-47.

[LLS 75] Ladner, R. E., Lynch N. A., and Selman A. L. (1975), "A comparison of polynomial time reducibilities," *Theoret. Comput. Sci.*, vol. 1, pp.103-123.

[Od 83] Odifreddi, P. (1983), "Forcing and reducibilities," *J. Symbolic Logic*, vol. 48, pp.288-310.

[Od 89] Odifreddi, P. (1989), "Classical recursion theory," North-Holland, Amsterdam.

[Od 99] Odifreddi, P. (1999), "Classical recursion theory, vol. II," North-Holland, Amsterdam.

[Po 86] Poizat, B. (1986), "$\mathcal{Q} = \mathcal{NQ}$?," *J. Symbolic Logic*, vol. 51, pp.22-32.

[Ro 67] Rogers, H. Jr. (1967), "Theory of recursive functions and effective computability," MacGraw-Hill, New York.

[So 87] Soare, R.I. (1987), "Recursively enumerable sets and degrees," Springer, Berlin.

[Su 98] Suzuki, T. (1998), "Recognizing tautology by a deterministic algorithm whose while-loop's execution time is bounded by forcing," *Kobe Journal of Mathematics*, vol.15, pp.91-102.

[Su 99] Suzuki, T. (1999), "Computational complexity of Boolean formulas with query symbols," Doctoral dissertation, Institute of Mathematics, University of Tsukuba, Tsukuba-City, Japan.

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

[Su (b)] Suzuki, T., 'Forcing complexity: supplement to "Complexity of the r-query tautologies," ' *submitted*.