Bounded Query Classes and the Difference Hierarchy

Richard Beigel[*]
Department of Computer Science
The Johns Hopkins University
Baltimore, MD 21218

William I. Gasarch[†]
Department of Computer Science
Institute for Advanced Studies
University of Maryland
College Park, MD 20742

Louise Hay[‡]
Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago
Chicago, IL 60680

January 5, 1995

Abstract

Let A be any nonrecursive set. We define a hierarchy of sets (and a corresponding hierarchy of degrees) that are reducible to A based on bounding the number of queries to A that an oracle machine can make. When A is the halting problem K our hierarchy of sets interleaves with the difference hierarchy

[*]Current address: Department of Computer Science, Yale University, 51 Prospect Street, P.O. Box 2158 Yale Station, New Haven, CT 06520. Supported in part by NSF grant CCR-8808949. Part of this work was completed while this author was a student at Stanford University supported by fellowships from the National Science Foundation and from the Fannie and John Hertz Foundation.

[†]Supported in part by NSF grant CCR-8803641.

[‡]Part of this work was completed while this author was on sabbatical leave at the University of California, Berkeley.

on the r.e. sets in a logarithmic way; this follows from a tradeoff between the number of parallel queries and the number of serial queries needed to compute a function with oracle K.

1 Introduction

The traditional reducibilities of recursion theory are one-one reducibility, many-one reducibility, truth-table reducibility, weak truth-table reducibility, and Turing reducibility [Rog67]. The differences among these reducibilities are essentially qualitative.

In this paper we consider variations on (weak) truth-table reducibility and Turing reducibility in which only a *fixed* number of queries are allowed. These reducibilities differ from each other quantitatively, depending on the number of queries allowed. It seems reasonable to consider the question of whether (and which) functions or decision problems are computable using, say, $k+1$ queries to the oracle but not using only k queries.

While this approach might seem unusual from the point of view of computability theory, it is quite natural from the point of view of complexity theory. For example the problem of minimum-comparison sorting is equivalent to determining the minimum number of sequential queries necessary in order to reduce the sorting problem to the comparison problem. Thus, given a problem, the number of queries required in order to solve that problem can be considered a measure of the problem's complexity.

In this paper we restrict our attention to computations relative to an oracle for the halting problem. A main result will be a numerical tradeoff between serial and parallel queries when the oracle is K. Bounded query classes relative to K have arisen in other contexts (e.g., [Hay78]), and turn out to be closely related to the difference hierarchy on the r.e. sets (henceforth, simply "the difference hierarchy"), which was defined in [Add65] and studied in detail in [Ers68]. Ershov showed that the (union of all levels of the) difference hierarchy is equal to the Boolean algebra generated by the r.e. sets, which, by [Rog67, Theorem 14-IX], is equal to the class of sets that are bounded truth-table reducible to K. We refine this result by showing that the nth level of the difference hierarchy is equal to the class of sets bounded truth-table reducible to K via a truth-table with norm n; the result also holds for weak truth-table reducibility. Known properties of the difference hierarchy can thus be used to show that the degree hierarchies defined by limiting the number of parallel (or serial) queries to K are proper. A normal form theorem for functions which can be computed by mixing serial and parallel queries to K is also obtained.

Related questions dealing with bounded query computations relative to arbitrary oracles have been considered in [BGGO93, Bei88b]. Bounded query classes for polynomial-time computations relative to an oracle for the Boolean satisfiability problem have also been studied (see, for example,
[AG88, ABG90, Bei90, Bei91, Bei88a, BH91, CGH$^+$88, KSW87, Wag88, WW85]).

2 Notation and terminology

We use lower-case italic letters to refer to integers and functions, upper-case italic letters to refer to sets of integers, upper-case roman letters to refer to sets of sets of integers, and boldface letters to refer to Turing degrees. The recursion-theoretic terminology and notation will be that of [Rog67]. In particular, N denotes the natural numbers. We define recursive sequence-coding functions as follows: denote a fixed recursive pairing function from $N \times N$ onto N by $\langle x_0, x_1 \rangle$. By iteration there is a recursive coding function $\langle x_0, \ldots, x_{k-1} \rangle$ mapping N^k onto N; and by "dovetailing" there is a recursive map of $\bigcup_{k \geq 1} N^k$ onto N denoted by $\langle\langle x_0, \ldots, x_{n-1} \rangle\rangle$. We will abbreviate $f(\langle x_1, \ldots, x_n \rangle)$ by $f(x_1, \ldots, x_n)$. We write $\chi_A(x)$ to denote the characteristic function of A, and $|A|$ to denote the cardinality of A. The symbol $\oplus$ denotes the exclusive-or operator on 0-1 functions. The recursive join of A and B is denoted by $A \wedge B = \{2x : x \in A\} \cup \{2x+1 : x \in B\}$.

We identify Turing machines with programs (and partial recursive functions). $\{0\}^A$, $\{1\}^A, \ldots$ is a list of all Turing machines with (arbitrary) oracle A. If f is a total function, $\{e\}^f$ denotes the Turing machine with program e which can make function calls to f. $\{e\}^A_s$ denotes s stages in the computation of $\{e\}^A$. W^A_e is the domain of $\{e\}^A$, W_e the domain of $\{e\} = \{e\}^{\emptyset}$. K is the oracle for the halting problem. Since all commonly considered variants of the halting problem are recursively isomorphic, we lose no generality by considering only the halting problem for "oblivious" Turing machines, *i.e.*, machines that ignore their input. Thus $K = \{e : \{e\} \text{ halts on blank input}\}$ and $K_s = \{e : \{e\}_s \text{ halts on blank input}\}$. μ denotes the minimization operator.

We follow the standard notation for reducibilities in recursion theory. $A \leq_1 B$ means A is one-one reducible to B; $A \leq_{\mathrm{T}} B$ means A is Turing reducible to B. We write $A \leq_{\mathrm{tt}} B$ if A is truth-table reducible to B and $A \leq_{\mathrm{wtt}} B$ if A is weak truth-table reducible to B. Since we are concerned with the distinction between truth-table and weak truth-table reducibility, we will briefly discuss their definitions, based on those in [Rog67].

Definition 2.1

i. (*truth-table reducibility*) $A \leq_{\mathrm{tt}} B$ iff there is a recursive function g such that for each x,

   (a) $g(x) = \langle y_1, \ldots, y_{k_x}, f_x \rangle$, where $y_1, \ldots, y_{k_x} \in N$ and f_x is a code for a k_x-ary Boolean function; and

   (b) $x \in A$ if and only if $f_x(\chi_B(y_1), \ldots, \chi_B(y_{k_x})) = 1$.

ii. (*bounded truth-table reducibility*) $A \leq_{\mathrm{btt}} B$ iff $A \leq_{\mathrm{tt}} B$ and there is a fixed n such that $k_x \leq n$ for all x.

   In this case, we write $A \leq_{n\text{-tt}} B$ and call n the *norm* of the reduction; clearly $A \leq_{\mathrm{btt}} B$ iff $A \leq_{n\text{-tt}} B$ for some n.

Definition 2.2

i. (*weak truth-table reducibility*) $A \leq_{\text{wtt}} B$ iff there is an e and a recursive function g such that $\chi_A = \{e\}^B$ and for each x, $g(x) = \langle y_1, \ldots, y_{k_x} \rangle$ is a list which includes all queries used in the computation of $\{e\}^B(x)$.

ii. (*bounded weak truth-table reducibility*) $A \leq_{\text{bwtt}} B$ iff $A \leq_{\text{wtt}} B$ and there is a fixed n such that $k_x \leq n$ for all x.

   In this case, we write $A \leq_{n\text{-wtt}} B$ and call n the *norm* of the reduction; clearly $A \leq_{\text{bwtt}} B$ iff $A \leq_{n\text{-wtt}} B$ for some n.

Note that in the definition of weak truth-table reducibility, what is missing is the Boolean function f_x. While the entire set of queries must be produced before any answers are obtained, we are not given a *total* recursive specification that indicates how the answers to the queries about membership in B determine whether x is a member of A. It is possible for the weak-tt reduction $\chi_A(x) = \{e\}^B(x)$ to be total while $\{e\}^C(x)$ relative to some other oracle C may diverge. (In fact, if $\chi_A(x) = \{e\}^B(x)$ and $\{e\}^C$ is a characteristic function for all oracles C, it follows that $A \leq_{\text{tt}} B$ [Rog67, Theorem 9-XIX]). That $\leq_{\text{wtt}}$ and $\leq_{\text{tt}}$ are distinct reducibilities even on the r.e. sets was shown in [Lac65] where r.e. sets A, B are constructed such that A is weak truth-table reducible to B with norm 1 while A is not truth-table reducible to B (even with unbounded truth tables). To illustrate the subtleties involved, we sketch Lachlan's construction. Let B be the union of two disjoint r.e. sets A and E. Then $A \leq_{1\text{-wtt}} B$ via the following reduction: To decide if $x \in A$, the only query made is "is x in B?" If the answer is no, then $x \notin A$, since $A \subseteq B$. If the answer is yes, then $x \in A \cup E$. Enumerate the r.e. sets A and E simultaneously; x will either appear in A or in E, and in the latter case, $x \notin A$. Thus A is weak truth-table reducible to B with norm 1, while there is no obvious truth-table reduction of A to B; indeed, [Lac65] constructs sets A, E, $B = A \cup E$ using a priority argument that "defeats" every possible (bounded or unbounded) tt-reduction of A to B. It is evident that in the reduction procedure given above, if B is replaced by an arbitrary oracle C, the procedure will terminate only for $x \in A \cup E \cup \bar{C}$.

3 The Bounded Query Classes

We say that n queries to an oracle are made *in parallel* (or that n parallel queries are made) if a list of all n queries is produced before any of them is made. Otherwise we say that n queries are made *in series* (or that n serial queries are made, or simply that n queries are made). The essential difference is that computation is allowed between serial queries to an oracle; the answer to an earlier query may determine what query is to be made next.

Definition 3.1

i. If $\{e\}^A$ is an oracle Turing machine, $\{e\}^{A,(\le n)}$ is identical to $\{e\}^A$ except that if it tries to make more than n queries to its oracle then it diverges.

ii. f is n-Turing reducible to g (denoted $f \le_{n\text{-T}} g$) if $f = \{e\}^{g,(\le n)}$ for some e.

iii. A is n-Turing reducible to B (denoted $A \le_{n\text{-T}} B$) if $\chi_A = \{e\}^{B,(\le n)}$ (equivalently, $\chi_A \le_{n\text{-T}} \chi_B$).

Definition 3.2

i. f is n-parallel-query reducible to g if $f \le_{n\text{-T}} g$ and the queries are made in parallel.

ii. A is n-parallel-query reducible to B if $A \le_{n\text{-T}} B$ and the queries are made in parallel.

Since this definition is evidently equivalent to Definition 2.2(ii), we can use $A \le_{n\text{-wtt}} B$ to denote that A is n-parallel-query reducible to B. By analogy, we will use $f \le_{n\text{-wtt}} g$ to denote that f is n-parallel-query reducible to g.

(Some closely related notions are the partial truth-table reducibilities of [PR77] and [PR79], and the branch-finite bounded enumeration reducibility of [Coo87].)

We can now define the bounded-query classes as follows:

Definition 3.3 (Serial query classes)

i. $\mathrm{FQ}(n, f)$ is the class of all partial functions g such that $g \le_{n\text{-T}} f$.

ii. $\mathrm{Q}(n, f)$ is the class of all (characteristic functions of) sets in $\mathrm{FQ}(n, f)$.

iii. $\mathbf{Q}(n, f)$ is the class of Turing degrees of sets in $\mathrm{Q}(n, f)$.

iv. $\mathrm{Q}(n, A)$ denotes $\mathrm{Q}(n, \chi_A)$, the class of all sets B such that $B \le_{n\text{-T}} A$.

v. $\mathbf{Q}(n, A)$ denotes $\mathbf{Q}(n, \chi_A)$, the class of Turing degrees of sets in $\mathrm{Q}(n, A)$.

Similarly, we have

Definition 3.4 (Parallel query classes)

i. $\mathrm{FQ}_{||}(n, f)$ is the class of all partial functions g such that $g \le_{n\text{-wtt}} f$.

ii. $\mathrm{Q}_{||}(n, f)$ is the class of all (characteristic functions of) sets in $\mathrm{FQ}_{||}(n, f)$.

iii. $\mathbf{Q}_{||}(n, f)$ is the class of Turing degrees of sets in $\mathrm{Q}_{||}(n, f)$.

iv. $\mathrm{Q}_{||}(n, A)$ denotes $\mathrm{Q}_{||}(n, \chi_A)$, the class of all sets B such that $B \le_{n\text{-wtt}} A$.

v. $\mathbf{Q}_{||}(n, A)$ denotes $\mathbf{Q}_{||}(n, \chi_A)$, the class of Turing degrees of sets in $\mathrm{Q}_{||}(n, A)$.

We note here some simple lemmas about the bounded query classes which will be needed later, and which are easily verified.

Lemma 3.5 (Transitivity Lemma)

i. If $f \in \mathrm{FQ}(m, g)$ *and* $g \in \mathrm{FQ}(n, h)$*, then* $f \in \mathrm{FQ}(mn, h)$.

ii. If $f \in \mathrm{FQ}_{||}(m, g)$ *and* $g \in \mathrm{FQ}_{||}(n, h)$*, then* $f \in \mathrm{FQ}_{||}(mn, h)$.

Definition 3.6 For any set A and each n, let

$$\mathrm{F}_n^A(x_1, \ldots, x_n) = \langle \chi_A(x_1), \ldots, \chi_A(x_n) \rangle.$$

Thus F_n^A computes the results of n parallel queries to A.

Lemma 3.7 $\mathrm{FQ}(1, \mathrm{F}_n^A) = \mathrm{FQ}_{||}(n, A)$.

4 A Tradeoff Between Serial Queries and Parallel Queries to K

In this section we exhibit a tradeoff between the number of parallel queries to K needed in order to compute a function and the number of serial queries to K needed in order to compute the same function. The tradeoff is roughly logarithmic, reflecting the use of binary search to find a number in a sequence.

Given an oracle A, we define some functions which are useful in analyzing bounded query classes relative to A.

Definition 4.1 $\#_n^A(x_1, \ldots, x_n) = |A \cap \{x_1, \ldots, x_n\}|$.

Thus $\#_n^A$ determines how many of n integers are elements of A.

Definition 4.2 $\mathrm{GEQ}^A = \{\langle\langle i, x_1, \ldots, x_n \rangle\rangle : \#_n^A(x_1, \ldots, x_n) \geq i\}$.

Thus GEQ^A determines if at least i out of n integers are elements of A.

Lemma 4.3 $\#_n^A \in \mathrm{FQ}(\lceil \log(n+1) \rceil, \mathrm{GEQ}^A)$.

Proof: $\#_n^A(x_1, \ldots, x_n)$ is an integer k such that $0 \leq k \leq n$. For any i, a single query to GEQ^A will tell us whether $k \geq i$. Let $m = \lceil \log(n+1) \rceil$, *i.e.*, $2^{m-1} \leq n \leq 2^m - 1$, $m \geq 1$. Thus k is an element of a sequence of at most 2^m numbers, and can be located by binary search as follows: One query ("is $k \geq 2^{m-1}$?") identifies which half of the sequence k is in, hence locates k as an element of a sequence of length at most 2^{m-1}. Iterating inductively, at most m queries will locate k as an element of a sequence of length at most 1 and hence will compute k. ▮

Lemma 4.4 *If A is r.e. then*

i. GEQ^A *is r.e.*

ii. $\mathrm{F}_n^A \in \mathrm{FQ}(1, \#_n^A)$.

Proof:

i. If A is r.e., $|A \cap \{x_1, \ldots, x_n\}| \geq i$ is evidently an r.e. condition.

ii. We can determine which of $x_1, \ldots, x_n$ are in A as follows: Compute $\#_n^A(x_1, \ldots, x_n) = k$, say; let s be the least stage in the enumeration of A such that exactly k elements from $x_1, \ldots, x_n \in A_s$. Then $x_i \in A$ if and only if $x_i \in A_s$.

▮

Lemma 4.5

i. $\mathrm{GEQ}^K \in \mathrm{Q}(1, K)$.

ii. $\#_n^K \in \mathrm{FQ}(\lceil \log(n+1) \rceil, K)$.

iii. $\mathrm{FQ}_{||}(n, K) \subseteq \mathrm{FQ}(1, \#_n^K)$.

Proof:

i. By Lemma 4.4, GEQ^K is r.e. and hence $\mathrm{GEQ}^K \leq_1 K$, which in turn implies $\mathrm{GEQ}^K \in \mathrm{Q}(1, K)$.

ii. By Lemma 4.3, $\#_n^K \in \mathrm{FQ}(\lceil \log(n+1) \rceil, \mathrm{GEQ}^K)$. The result now follows from (i) by the transitivity lemma (Lemma 3.5).

iii. Assume $f \in \mathrm{FQ}_{||}(n, K)$. By Lemma 3.7, $f \in \mathrm{FQ}(1, \mathrm{F}_n^K)$; hence by Lemma 4.4(ii) and the transitivity lemma, $f \in \mathrm{FQ}(1, \#_n^K)$.

▮

Lemma 4.6 $\mathrm{FQ}_{||}(n, K) \subseteq \mathrm{FQ}(\lceil \log(n+1) \rceil, K)$.

Proof: By Lemma 4.5(iii,ii) and the transitivity lemma. ▮

Corollary 4.7 $\mathrm{FQ}_{||}(2^n - 1, K) \subseteq \mathrm{FQ}(n, K)$.

Thus in reductions to K, $2^n - 1$ parallel queries can be replaced by n serial queries. The reverse inclusion was originally proved in [Bei87]. Since the proof uses substantially different techniques, it is deferred to Section 8. For our purposes, it is sufficient to prove the reverse only for total functions; this will be done in the next section.

5 Limiting Recursive Functions

The notion of bounded "mind changes" in recursive approximations to functions was introduced independently in [Put65] (which called them "k-trial predicates") and in [Gol65], which gave essentially the following definition:

Definition 5.1 A (partial) function g is n-limiting recursive (abbreviated n-l.r.) if there exists a two-place partial recursive function f such that for all x

$$\lim_{s\to\infty} f(x,s) = g(x) \qquad (i.e., (\exists s_0)(\forall s \geq s_0)[f(x,s) = g(x)])$$
$$|\{s : f(x,s) \neq f(x,s+1)\}| \leq n$$

We adopt the convention that if two functions both diverge, then their values are equal. If exactly one of them converges, then their values are unequal.

We say that f is an approximation to g if $f(x,\cdot)$ agrees with $g(x)$ in the limit; we say that f changes its mind at s if $f(x,s) \neq f(x,s+1)$. Observe that g is n-l.r. iff g is approximated by a function f that changes its mind at most n times.

Actually, our definition of n-l.r. is more general than Gold's original definition, because Gold's definition required that f and g both be total functions. The following is easily verified:

Proposition 5.2 *If g is n-l.r. then g has an approximation $f(x,s)$ which changes its mind at most n times and such that $\{s : f(x,s)$ converges$\}$ is an initial segment of N.*

In particular, if $g(x)$ converges, $f(x,s)$ will be a total function of s. It follows that our definition of n-l.r. is consistent with Gold's original definition in the case when g is a total n-l.r. function.

Definition 5.3

i. Let LR_n denote the class of (partial) n-l.r. functions.

ii. Let TLR_n denote the class of total n-l.r. functions.

The notion of n-l.r. ties in with bounded queries to K as follows:

Lemma 5.4 *If $f(x,s)$ is a total recursive function, let*

$$c(x,i) = \begin{cases} 1 & \text{if } f(x,s) \text{ changes its mind at least } i \text{ times} \\ 0 & \text{otherwise.} \end{cases}$$

Then $c \in \mathrm{FQ}(1,K)$.

Proof: Since f is total recursive, $|\{s : f(x,s) \neq f(x,s+1)\}| \geq i$ is evidently an r.e. condition. Therefore $c \in \mathrm{FQ}(1,K)$. ▮

Theorem 5.5 *Let $\mathrm{TFQ}_{||}(n,K)$ denote the class of total functions in $\mathrm{FQ}_{||}(n,K)$. Then*

i. $\mathrm{FQ}_{||}(n,K) \subsetneq \mathrm{LR}_n$*;*

ii. $\mathrm{TFQ}_{||}(n,K) = \mathrm{TLR}_n$*.*

Proof:

i. First we show that every function in $\mathrm{FQ}_{||}(n,K)$ is n-l.r. Let $g \in \mathrm{FQ}_{||}(n,K)$ be computed by $\{e\}^{K,(\leq n)}$, where all queries are produced before any of them is made. We define an approximation f as follows:

$$f(x,s) = \{e\}^{K_s,(\leq n)}(x).$$

Then $f(x,s)$ converges to $g(x)$. The approximation can change its mind only when the oracle provides different information. Since the parallel queries made by $\{e\}^{K_s,(\leq n)}$ do not depend on s, and

$$K_0 \subseteq K_1 \subseteq K_2 \subseteq \cdots,$$

the sequence of n-tuples of oracle answers is monotone in each component. Thus at most $n+1$ different n-tuples of oracle answers can be received and the approximation cannot change its mind more than n times.

That the inclusion is proper will follow from Proposition 5.6(i,iii).

ii. By (i), $\mathrm{TFQ}_{||}(n, K) \subseteq \mathrm{TLR}_n$, hence it suffices to show that $\mathrm{TLR}_n \subseteq \mathrm{TFQ}_{||}(n, K)$. Let g be a total n-l.r. function. By Proposition 5.2 we may assume f is a total recursive approximation to g. By Lemma 5.4, a single query to K tells us whether f changes its mind at least i times. By computing $c(x, 1), \ldots, c(x, n)$ simultaneously, n parallel queries to K allow us to determine the exact number t of times f changes its mind. We can now compute $g(x)$ by evaluating $f(x, s)$ for $s = 1, 2, \ldots$ until f has changed its mind t times. Hence n parallel queries to K allow us to compute g, so g is in $\mathrm{TFQ}_{||}(n, K)$.

▮

Proposition 5.6 *Let* $\mathrm{INF} = \{e : W_e \text{ is infinite}\}$*, and define*

$$h(x) = \begin{cases} 1 & \text{if } x \in \mathrm{INF} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Then

i. h is 1-l.r.,

ii. if $h = \{e\}^A$ *then* $\mathrm{INF} \leq_{\mathrm{T}} A \wedge K$*, the recursive join of A and K,*

iii. $(\forall e)(h \neq \{e\}^K)$.

Proof:

i. h is approximated by the partial recursive function

$$f(x, s) = \begin{cases} 1 & \text{if } |W_x| > s \\ \text{undefined} & \text{otherwise.} \end{cases}$$

The function $f(x, s)$ changes its mind at most once (when $s = |W_x|$ if W_x is finite, at which point $f(x, s)$ becomes and remains undefined); therefore h is 1-l.r.

ii. Suppose that $h = \{e\}^A$. Then $\mathrm{INF} = W_e^A$, so INF is r.e. in A. Since $\mathrm{INF} \in \Pi_2^0$, INF is co-r.e. in K. Thus, INF is r.e. in $A \wedge K$ and also co-r.e. in $A \wedge K$. Therefore INF is recursive in $A \wedge K$.

iii. follows from (ii) because INF is Π_2^0-complete and hence not Turing-reducible to K.

▮

Proposition 5.6(i) implies that h is n-l.r. for all n, but Proposition 5.6(iii) implies that h is not in $\mathrm{FQ}_{||}(m, K)$ or $\mathrm{FQ}(m, K)$ for any m. It may be of interest to note that every n-l.r. function $g(x)$ can be computed using a single query to INF followed by at most n parallel queries to K. (Using Proposition 5.2, let $f(x, s)$ be a partial recursive approximation to $g(x)$ in which $W_e = \{s : f(x, s) \text{ converges}\}$ is an initial segment. Then one query to INF determines if W_e is infinite and hence whether $g(x)$ converges. If it does, we can then compute $g(x)$ as in the proof of Theorem 5.5(ii).) Proposition 5.6 shows that this is in some sense the best possible result for partial n-l.r. functions.

Lemma 5.7 $\mathrm{FQ}(n, K) \underset{\neq}{\subset} \mathrm{LR}_{2^n-1}$

Proof: Let g be in $\mathrm{FQ}(n, K)$, and let g be computed by $\{e\}^{K,(\leq n)}$. As before, we define an approximation f as follows:

$$f(x, s) = \{e\}^{K_s,(\leq n)}(x).$$

The approximation can change its mind only when the oracle provides different information. While the queries made by $\{e\}^{K_s,(\leq n)}$ are no longer independent of s (since a query may depend on the answer to a prior query), there are only 2^n possible sequences of n oracle answers. Since

$$K_0 \subseteq K_1 \subseteq K_2 \ldots$$

the infinite sequence of sequences of n oracle answers has no repetitions except for blocks of identical answers. (This can be seen as follows: Consider sequences of "yes" or "no" answers as Boolean sequences of 0's and 1's of length n, arranged in increasing lexicographic order. As s increases, an oracle answer string can only move to the right, since the answer to the query "$a_i \in K_s$?" can change from 0 to 1, but the answer can change from 1 to 0 only if the query itself changes; this can only happen if the answer to a prior query changes, while the answer to the first query can only go from 0 to 1). Therefore $f(x, s)$ changes at most $2^n - 1$ different times, so g is $(2^n - 1)$-l.r.

Strict inclusion follows from Proposition 5.6(i,iii). ∎

Theorem 5.8 (Tradeoff Theorem for Sets) $\mathrm{Q}(n, K) = \mathrm{Q}_{||}(2^n - 1, K)$.

Proof: For $\subseteq$, assume $S \in \mathrm{Q}(n, K)$. Then $\chi_S \in \mathrm{FQ}(n, K)$, hence is $(2^n - 1)$-l.r. by Lemma 5.7. Since χ_S is total, Theorem 5.5(ii) implies that $\chi_S \in \mathrm{TFQ}_{||}(2^n - 1, K)$; therefore $S \in \mathrm{Q}_{||}(2^n - 1, K)$. The reverse inclusion follows from Corollary 4.7. ∎

6 The Difference Hierarchy

In this section we show that the bounded query classes interleave the difference hierarchy, and consequently we obtain separation results. The difference hierarchy was first studied in detail in [Ers68], where the connection with the "k-trial predicates" of [Put65] was noted. The n-r.e. sets and weakly n-r.e. sets were introduced in [Eps79] and [EHK81] respectively, where their Turing degrees were considered.

Definition 6.1

i. A set B is weakly n-r.e. if χ_B is n-l.r.

ii. A set B is n-r.e. if B is weakly n-r.e. via an approximation function f such that $f(x,0)=0$.

iii. A Turing degree is (weakly) **n**-r.e. if it contains an (weakly) n-r.e. set.

With no loss of generality, we will always assume that χ_A is approximated by a 0,1-valued total function.

Definition 6.2

i. The levels of the difference hierarchy (on the r.e. sets) are defined by

$$\begin{aligned} \mathrm{D}_1 &= \{X : X \text{ is r.e.}\}, \\ \mathrm{D}_{n+1} &= \{X - Y : X \text{ is r.e., } Y \in \mathrm{D}_n\}. \end{aligned}$$

ii. co-$\mathrm{D}_n = \{X : \bar{X} \in \mathrm{D}_n\}$.

iii. $\mathbf{D}_n$ is the class of Turing degrees of sets in D_n.

iv. $\nabla_n = \mathrm{D}_n \cap \text{co-}\mathrm{D}_n$.

v. $\boldsymbol{\nabla}_n$ is the class of Turing degrees of sets in ∇_n.

The following facts follow from [Ers68] and [EHK81] (The case $\mathbf{D}_1 \underset{\neq}{\subset} \mathbf{D}_2$ is due to Cooper [Coo71]):

Fact 6.3

i. $\mathrm{D}_n =$ *the class of n-r.e. sets.*

ii. $\nabla_{n+1} =$ *the class of weakly n-r.e. sets.*

iii. $\mathrm{D}_n \underset{\neq}{\subset} \nabla_{n+1} \underset{\neq}{\subset} \mathrm{D}_{n+1}$.

iv. $\mathbf{D}_n \subsetneq \mathbf{D}_{n+1}$.

Although [EHK81] separated the $\mathbf{n}$-r.e. degrees, the question of how the $\mathbf{n}$-r.e. degrees interleave with the weakly $\mathbf{n}$-r.e. degrees was not considered (other than the observation that every weakly $\mathbf{n}$-r.e. degree is $(\mathbf{n+1})$-r.e.). We will show that there are $(\mathbf{n}+\mathbf{1})$-r.e. degrees that are not weakly $\mathbf{n}$-r.e.

Theorem 6.4

i. The n-r.e. sets and the weakly n-r.e. sets have the same 1-truth-table degrees.

ii. $\mathbf{D}_n = \boldsymbol{\nabla}_{n+1}$.

Proof:

i. This can be deduced from [Ers68, Corollary 1 to Proposition 3], where it is shown that $A \in \nabla_{n+1}$ if and only if there exists sets $B, C \in \mathrm{D}_n$ and a recursive set R such that $A \cap R = B$ and $A \cap \bar{R} = \bar{C}$. A self-contained proof follows: Every n-r.e. set is 1-truth-table equivalent to a weakly n-r.e. set, namely itself. Conversely, assume that A is weakly n-r.e., and let f be an approximation to χ_A. Let B be the set of all x such that $f(x,s)$ changes its mind an odd number of times; thus B is n-r.e. via the approximation function $f(x,s) \oplus f(x,0)$. Since

$$\chi_B(x) = f(x,0) \oplus \chi_A(x)$$

and

$$\chi_A(x) = f(x,0) \oplus \chi_B(x),$$

A and B are 1-truth-table equivalent.

ii. follows from (i) and Fact 6.3.

▮

Corollary 6.5 *For all n, there is an $(\mathbf{n+1})$-r.e. degree that is not weakly $\mathbf{n}$-r.e.*

Proof: Fact 6.3(iv) and Theorem 6.4(ii) imply that $\boldsymbol{\nabla}_{n+1} \subsetneq \mathbf{D}_{n+1}$. ▮

Corollary 6.6 *For all $n \geq 1$, $\boldsymbol{\nabla}_n \subsetneq \boldsymbol{\nabla}_{n+1}$.*

Proof: By Theorem 6.4(ii) and Fact 6.3(iv). ▮

Definition 6.7 $\mathrm{PARITY}_n^A(x_1, \ldots, x_n) = \#_n^A(x_1, \ldots, x_n) \bmod 2$.

Thus PARITY_n^A determines if an odd number of n integers are elements of A.

Lemma 6.8 *If A is weakly n-r.e. then*

i. $A \in \mathrm{Q}(1, \mathrm{PARITY}_n^K)$.

ii. $A \leq_{n\text{-tt}} K$.

Proof: Assume that A is weakly n-r.e.; thus χ_A is approximated by a total function $f(x,s)$ that changes its mind at most n times. Let $N(x)$ denote the number of times that f changes its mind when its first argument is x, and let

$$P(x) = N(x) \bmod 2.$$

Note that

$$\chi_A(x) = P(x) \oplus f(x,0)$$

Proceeding as in Lemma 5.4, we define for each k, $1 \leq k \leq n$, an oblivious program $\{e_k\}$ that simulates $f(x,s)$ for $s = 1, 2, \ldots$ and halts as soon as f makes its kth mind change. Thus $\{e_k\}$ halts if and only if $N(x) \geq k$. Then $e_1, \ldots, e_n$ can be effectively computed, and

$$P(x) = \mathrm{PARITY}_n^K(e_1, \ldots, e_n),$$

so that

$$\chi_A(x) = \mathrm{PARITY}_n^K(e_1, \ldots, e_n) \oplus f(x,0).$$

This implies both results. ∎

We can now show that n-tt and n-wtt reducibilities to K coincide.

Theorem 6.9 *The following are equivalent:*

i. $A \leq_{n\text{-wtt}} K$,

ii. $A \in \mathrm{Q}_{||}(n, K)$,

iii. *A is weakly n-r.e.,*

iv. $A \in \mathrm{Q}(1, \mathrm{PARITY}_n^K)$,

v. $A \leq_{n\text{-tt}} K$.

Proof: (i) and (ii) are equivalent by definition. (ii) implies (iii) by Theorem 5.5(i) and Definition 6.1. (iii) implies (iv) and (v) by Lemma 6.8. (v) implies (i) by definition. ∎

In contrast, recall that Lachlan showed that relative to some oracles n-wtt reducibility need not imply n-tt reducibility or even unbounded tt-reducibility, even when $n = 1$ [Lac65]. In [Rog67, Exercise 9-45], it was shown that if $K \leq_{\mathrm{tt}} B$ then

$$(A \leq_{\mathrm{wtt}} B) \Rightarrow (A \leq_{\mathrm{tt}} B);$$

hence, in particular

$$(A \leq_{\mathrm{wtt}} K) \Rightarrow (A \leq_{\mathrm{tt}} K).$$

In Rogers's proof sketch, however, the tt-reduction uses more queries than the wtt-reduction, and thus his methods do not directly yield our result that n-wtt reducibility to K is equivalent to n-tt reducibility to K.

That tt-reducibility to K via a *fixed* truth table of norm n implies membership in $\mathrm{Q}(1, \mathrm{PARITY}_n^K)$ also follows from [Hay78], where it is shown that $A \leq_{n\text{-tt}} K$ via a reduction whose truth table is independent of the input if and only if

$$(A \leq_1 \mathrm{PARITY}_n^K) \text{ or } (\bar{A} \leq_1 \mathrm{PARITY}_n^K).$$

Corollary 6.10

$$\mathrm{Q}(n, K) = \mathrm{Q}_{||}(2^n - 1, K) = \nabla_{2^n} = \textit{the class of weakly } (2^n - 1)\textit{-r.e. sets.}$$

Proof: This follows from Theorem 6.9 and Theorem 5.8. ∎

Theorem 6.11 (Separation)

i. $\mathbf{Q}_{||}(n, K) \subsetneq \mathbf{Q}_{||}(n+1, K)$.

ii. $\mathbf{Q}(n, K) \subsetneq \mathbf{Q}(n+1, K)$.

Proof:

i. By Theorem 5.5, $\mathrm{Q}_{||}(n, K) = \nabla_{n+1}$, hence $\mathbf{Q}_{||}(n, K) = \boldsymbol{\nabla}_{n+1}$. The result follows by Corollary 6.6.

ii. It follows from (i) that $\mathbf{Q}_{||}(2^n - 1, K) \subsetneq \mathbf{Q}_{||}(2^{n+1} - 1, K)$. The result follows by Theorem 5.8.

∎

We have thus obtained the following interleaving of the hierarchies:

Theorem 6.12 *The sets and degrees of the* Q *and* $\mathrm{Q}_{||}$ *hierarchies interleave with those of the difference hierarchy as follows:*

$$\begin{aligned}
\mathrm{D}_1 &\subsetneq \mathrm{Q}_{||}(1,K) = \mathrm{Q}(1,K) = \nabla_2 \subsetneq \mathrm{D}_2 \subsetneq \mathrm{Q}_{||}(2,K) = \nabla_3 \subsetneq \\
\mathrm{D}_3 &\subsetneq \mathrm{Q}_{||}(3,K) = \mathrm{Q}(2,K) = \nabla_4 \subsetneq \mathrm{D}_4 \subsetneq \mathrm{Q}_{||}(4,K) = \nabla_5 \subsetneq \cdots \subsetneq \\
\mathrm{D}_n &\subsetneq \mathrm{Q}_{||}(n,K) = \nabla_{n+1} \subsetneq \mathrm{D}_{n+1} \subsetneq \mathrm{Q}_{||}(n+1,K) = \nabla_{n+2} \subsetneq \cdots \subsetneq \\
\mathrm{D}_{2^n-1} &\subsetneq \mathrm{Q}_{||}(2^n-1,K) = \mathrm{Q}(n,K) = \nabla_{2^n} \subsetneq \mathrm{D}_{2^n} \subsetneq \mathrm{Q}_{||}(2^n,K) = \nabla_{2^n+1} \subsetneq \cdots
\end{aligned}$$

$$\begin{aligned}
\mathbf{D}_1 &= \mathbf{Q}_{||}(1,K) = \mathbf{Q}(1,K) = \boldsymbol{\nabla}_2 \subsetneq \mathbf{D}_2 = \mathbf{Q}_{||}(2,K) = \boldsymbol{\nabla}_3 \subsetneq \\
\mathbf{D}_3 &= \mathbf{Q}_{||}(3,K) = \mathbf{Q}(2,K) = \boldsymbol{\nabla}_4 \subsetneq \mathbf{D}_4 = \mathbf{Q}_{||}(4,K) = \boldsymbol{\nabla}_5 \subsetneq \cdots \subsetneq \\
\mathbf{D}_n &= \mathbf{Q}_{||}(n,K) = \boldsymbol{\nabla}_{n+1} \subsetneq \mathbf{D}_{n+1} = \mathbf{Q}_{||}(n+1,K) = \boldsymbol{\nabla}_{n+2} \subsetneq \cdots \subsetneq \\
\mathbf{D}_{2^n-1} &= \mathbf{Q}_{||}(2^n-1,K) = \mathbf{Q}(n,K) = \boldsymbol{\nabla}_{2^n} \subsetneq \mathbf{D}_{2^n} = \mathbf{Q}_{||}(2^n,K) = \boldsymbol{\nabla}_{2^n+1} \subsetneq \cdots
\end{aligned}$$

7 Mixing Serial and Parallel Queries

We ask next what total functions we can compute if we are allowed to make n_1 parallel queries to K, followed by n_2 parallel queries to K, ..., followed by n_r parallel queries to K. We present a normal form for such computations in terms of a single round of queries to K. This result can be extended to partial functions using techniques to be developed in the next section.

We define the composition of sets of functions in the natural way:

Definition 7.1 If C_1 and C_2 are two sets of functions then

$$C_1 \circ C_2 = \{f_1 \circ f_2 : f_1 \in C_1 \text{ and } f_2 \in C_2\}.$$

We now show that composition of two bounded-query classes corresponds to allowing a number of queries to one oracle followed by a number of queries to a second oracle.

Proposition 7.2 *A partial function g can be computed by making at most n_1 parallel queries to f_1, followed by at most n_2 parallel queries to f_2, ..., followed by at most n_r parallel queries to f_r if and only if*

$$g \in \mathrm{FQ}_{||}(n_r, f_r) \circ \mathrm{FQ}_{||}(n_{r-1}, f_{r-1}) \circ \cdots \circ \mathrm{FQ}_{||}(n_1, f_1).$$

Proof: Assume that g can be computed by an oracle Turing machine that makes at most n_1 parallel queries to f_1, followed by at most n_2 parallel queries to f_2, ..., followed by at most n_r parallel queries to f_r. We can define g_i as follows: g_i's input is an instantaneous description of g (*i.e.*, a tape configuration and internal state). Starting from that instantaneous description, g_i simulates g until g is about to make some parallel queries. If the parallel queries are to f_i, and g is not attempting to make more than n_i of them, then g_i makes the queries; otherwise g_i diverges. g_i then continues the simulation of g until g has terminated or is about to make more queries. If g is about to make more queries, then g_i prints the current instantaneous description of g and halts. Thus

$$g = g_r \circ \cdots \circ g_1 \in \mathrm{FQ}_{||}(n_r, f_r) \circ \cdots \circ \mathrm{FQ}_{||}(n_1, f_1).$$

The converse is immediate. ▮

The following generalization of Lemma 4.3 is the key to classifying compositions of bounded query classes:

Lemma 7.3

$$\#^A_{(n_1+1)\cdots(n_r+1)-1} \in \mathrm{FQ}_{||}(n_r, \mathrm{GEQ}^A) \circ \cdots \circ \mathrm{FQ}_{||}(n_1, \mathrm{GEQ}^A)$$

Proof: Let $N = (n_1+1)\cdots(n_r+1)$. Now $\#^A_{N-1}$ is a function whose result k has one of N possible values ($0 \leq k \leq N-1$). For any t, a single query to GEQ^A will tell us whether $k \geq t$. With n_1 parallel queries, we ask whether $k \geq N/(n_1+1)$, $k \geq 2N/(n_1+1)$, ..., $k \geq n_1 N/(n_1+1)$. These queries restrict k to a range of $N/(n_1+1)$ possible values. Similarly, our next n_2 parallel queries can restrict k to a range of $N/((n_1+1)(n_2+1))$ possible values. Continuing inductively, our final n_r parallel queries restrict k to a range of $N/((n_1+1)\cdots(n_r+1)) = 1$ possible value. The theorem now follows from Proposition 7.2. ▮

Lemma 7.4

$$\mathrm{FQ}_{||}((n_1+1)\cdots(n_r+1)-1, K) \subseteq \mathrm{FQ}_{||}(n_r, K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1, K).$$

Proof: Let $N = (n_1+1)\cdots(n_r+1)-1$. By Lemma 4.5(i), $\mathrm{GEQ}^K \in \mathrm{Q}(1, K)$; hence by Lemma 7.3 and the transitivity lemma,

$$\#^K_N \in \mathrm{FQ}_{||}(n_r, K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1, K).$$

By Lemma 4.5(iii),

$$\mathrm{FQ}_{||}(N, K) \subseteq \mathrm{FQ}(1, \#^K_N);$$

the result now follows from the transitivity lemma. ▮

Next we prove a converse:

Lemma 7.5 *Every total function in* $\mathrm{FQ}_{||}(b, K) \circ \mathrm{FQ}_{||}(a, K)$ *belongs to* $\mathrm{FQ}_{||}((a+1)(b+1)-1, K)$.

Proof: By Theorem 5.5(ii), it suffices to show that the composition of an a-l.r. function with a b-l.r. function is $((a+1)(b+1)-1)$-l.r. Let g_1 be a-l.r. with approximation function f_1, and let g_2 be b-l.r. with approximation function f_2. Then we can define an approximation to g by

$$f(x,s) = f_1(f_2(x,s),s).$$

It is readily verified that g is $((a+1)(b+1)-1)$-l.r. ∎

Lemma 7.6 *Every total function in* $\mathrm{FQ}_{||}(n_r,K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1,K)$ *belongs to* $\mathrm{FQ}_{||}((n_1+1)\cdots(n_r+1)-1,K)$.

Proof: By induction on r. The base case $(r=1)$ is trivial. Assume the lemma holds for $r-1$. Let g be a total function in $\mathrm{FQ}_{||}(n_r,K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1,K)$. Then

$$g = g_r \circ g',$$

where g' is a total function in $\mathrm{FQ}_{||}(n_{r-1},K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1,K)$ and $g_r \in \mathrm{FQ}_{||}(n_r,K)$.

By the inductive hypothesis,

$$g' \in \mathrm{FQ}_{||}((n_1+1)\cdots(n_{r-1}+1)-1,K).$$

Therefore

$$g = g_r \circ g' \in \mathrm{FQ}_{||}(n_r,K) \circ \mathrm{FQ}_{||}((n_1+1)\cdots(n_{r-1}+1)-1,K).$$

Since g is total, Lemma 7.5 implies that

$$g \in \mathrm{FQ}_{||}((n_1+1)\cdots(n_r+1)-1,K).$$

∎

Theorem 7.7 (Normal Form)

i. $\mathrm{Q}_{||}((n_1+1)\cdots(n_r+1)-1,K) \subseteq \mathrm{FQ}_{||}(n_r,K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1,K)$.

ii. Every decision problem in $\mathrm{FQ}_{||}(n_r,K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1,K)$ *belongs to* $\mathrm{Q}_{||}((n_1+1)\cdots(n_r+1)-1,K)$.

Proof:

i. This follows from Lemma 7.4.

ii. Since decision problems are total functions, this follows from Lemma 7.6.

▮

The Normal Form Theorem allows a corollary about computations that are allowed to make n rounds of parallel queries, with at most p queries per round.

Corollary 7.8

i. $\mathrm{Q}_{||}((p+1)^n-1,K) \subseteq \underbrace{\mathrm{FQ}_{||}(p,K)\circ\cdots\circ\mathrm{FQ}_{||}(p,K)}_{n}$

ii. Every decision problem in $\underbrace{\mathrm{FQ}_{||}(p,k)\circ\cdots\circ\mathrm{FQ}_{||}(p,K)}_{n}$ *belongs to* $\mathrm{Q}_{||}((p+1)^n-1,K)$.

iii. $\mathrm{Q}(\lfloor n\log_2(p+1)\rfloor,K) \subseteq \underbrace{\mathrm{FQ}_{||}(p,K)\circ\cdots\circ\mathrm{FQ}_{||}(p,K)}_{n}$

iv. Every decision problem in $\underbrace{\mathrm{FQ}_{||}(p,K)\circ\cdots\circ\mathrm{FQ}_{||}(p,K)}_{n}$ *belongs to* $\mathrm{Q}(\lceil n\log_2(p+1)\rceil,K)$.

Proof: (i) and (ii) follow by letting $n_i = p$ in Theorem 7.7. (iii) and (iv) follow from (i) and (ii) by Theorem 5.8. ▮

8 Tradeoff Revisited

We now give the proof of the reverse inclusion to that in Corollary 4.7.

Lemma 8.1 $\mathrm{FQ}(n,K) \subseteq \mathrm{FQ}_{||}(2^n-1,K)$.

Proof: Let $f \in \mathrm{FQ}(n,K), f = \{e\}^{K,(\leq n)}$. For each i, $1 \leq i \leq n$, independently of the oracle A queried by $\{e\}^{A,(\leq n)}$, there are at most 2^{i-1} possibilities for the ith query — one for each sequence of answers to the previous $i-1$ queries. Summing over $1 \leq i \leq n$, this gives an *a priori* bound of 2^n-1 different queries that could be made in the computation of $\{e\}^{A,(\leq n)}$ on any input, regardless of the answers given by the oracle. It is not in general possible to pre-compute what all these queries might be, since some purported sequence of oracle answers might force $\{e\}^{A,(\leq n)}$ into a non-terminating computation for some oracles A.

However, we can construct a query that has the same answer as the ith query if an ith query is actually made. For each $i, 1 \leq i \leq n$, consider Boolean sequences α of $i-1$ potential answers (where 0 and 1 correspond to "no" and "yes" respectively). For each such α, define a Turing machine $\{e_\alpha\}$ which computes as follows: $\{e_\alpha\}(x)$ simulates $\{e\}^{A,(\leq n)}(x)$, except that instead of making calls to the oracle, $\{e_\alpha\}(x)$ uses the sequence α to answer the first $i-1$ queries, until an ith query q is produced. (If the computation halts before an ith query is produced, or if an ith query is never made, $\{e_\alpha\}(x)$ diverges). $\{e_\alpha\}$ then enumerates K and halts if and when q appears in K; otherwise, $\{e_\alpha\}(x)$ diverges. Note that $\{e_\alpha\}$ makes no queries to an oracle, and, if α is a correct sequence of $i-1$ answers for $\{e\}^{A,(\leq n)}(x)$ when the oracle is K, then $\{e_\alpha\}(x)$ halts if and only if $\{e\}^{K,(\leq n)}(x)$ makes at least i queries and the answer to the ith query is "yes".

Thus, for each sequence of potential answers to the first $i-1$ queries, we have shown how to produce a query to K ("does $\{e_\alpha\}(x)$ halt?") that has the same answer as the ith query of $\{e\}^{K,(\leq n)}(x)$ if the first $i-1$ answers given by α are correct and if i queries are actually made. (If the first $i-1$ answers are not all correct for K or if $\{e\}^{K,(\leq n)}(x)$ makes fewer than i queries, we do not care about the answer to the query that we produce).

By determining for each Boolean sequence α of length $i, 1 \leq i \leq n$, whether $\{e_\alpha\}(x)$ halts, we determine the answer to all the possible queries made by $\{e\}^{K,(\leq n)}(x)$. Thus the following algorithm simulates $\{e\}^{K,(\leq n)}(x)$ by making $\sum_{i=1}^{n} 2^{i-1} = 2^n - 1$ parallel queries to K: Simultaneously ask, for each $i, 1 \leq i \leq n$, and each α of length $i-1$, whether $\{e_\alpha\}(x)$ halts. Then simulate $\{e\}^{K,(\leq n)}(x)$ with no further queries, inductively using the answer to the question "does $\{e_\alpha\}(x)$ halt?" as the answer to the ith query if α is the sequence of answers to the first $i-1$ queries. Thus $f \in \mathrm{FQ}_{||}(2^n-1, K)$. ▮

Hence we have

Theorem 8.2 (Tradeoff theorem for functions) *For each n,*

$$\mathrm{FQ}(n, K) = \mathrm{FQ}_{||}(2^n - 1, K).$$

Using similar techniques, it is not hard to extend the results of the preceding section to partial functions:

Theorem 8.3 (Normal Form)

$$\mathrm{FQ}_{||}(n_r, K) \circ \cdots \circ \mathrm{FQ}_{||}(n_1, K) = \mathrm{FQ}_{||}((n_1+1)\cdots(n_r+1)-1, K).$$

9 Acknowledgments

The authors would like to thank Robert Floyd, John Gill, and Jim Owings for helpful comments.

References

[ABG90] Amihood Amir, Richard Beigel, and William I. Gasarch. Some connections between bounded query classes and nonuniform complexity. In *Proceedings of the 5th Annual Conference on Structure in Complexity Theory*, pages 232–243, 1990.

[Add65] J. W. Addison. The method of alternating chains. In *Theory of Models*, pages 1–16, Amsterdam, 1965. North-Holland Publishing Co.

[AG88] Amihood Amir and William I. Gasarch. Polynomial terse sets. *Inf. & Comp.*, 77:37–56, April 1988.

[Bei87] Richard Beigel. *Query-Limited Reducibilities.* PhD thesis, Stanford University, 1987. Also available as Report No. STAN-CS-88-1221.

[Bei88a] Richard Beigel. NP-hard sets are p-superterse unless R = NP. Technical Report 88-04, The Johns Hopkins University, Dept. of Computer Science, 1988.

[Bei88b] Richard Beigel. When are $k + 1$ queries better than k? Technical Report 88-06, The Johns Hopkins University, Dept. of Computer Science, 1988.

[Bei90] Richard Beigel. Bi-immunity results for cheatable sets. *Theoretical Computer Science*, 73(3):249–263, 1990.

[Bei91] Richard Beigel. Bounded queries to SAT and the Boolean hierarchy. *Theoretical Computer Science*, 84(2):199–223, July 1991.

[BGGO93] Richard Beigel, William I. Gasarch, John T. Gill, and James C. Owings. Terse, superterse, and verbose sets. *Inf. & Comp.*, 103:68–85, 1993.

[BH91] Sam R. Buss and Louise E. Hay. On truth table reducibility to SAT. *Inf. & Comp.*, 91(1):86–102, March 1991.

[CGH$^+$88] J. Cai, T. Gundermann, J. Hartmanis, L. A. Hemachandra, V. Sewelson, K. W. Wagner, and G. Wechsung. The Boolean hierarchy I: structural properties. *SICOMP*, 17(6):1232–1252, December 1988.

[Coo71] S. B. Cooper. *Degrees of Unsolvability.* PhD thesis, Leicester University, 1971.

[Coo87] S. Barry Cooper. Enumeration reducibility using bounded information: counting minimal covers. *Zeitsch. f. math. Logik und Grundlagen d. Math.*, 33:537–560, 1987.

[EHK81] Richard L. Epstein, Richard Haas, and Richard L. Kramer. Hierarchies of sets and degrees below $\mathbf{0}'$. In *Logic Year 1979–80*, volume 859 of *Lecture Notes in Mathematics*, pages 32–48, Berlin, 1981. Springer-Verlag. Volume 859 of *Lecture Notes in Mathematics.*

[Eps79] Richard L. Epstein. *Degrees of Unsolvability: Structure and Theory*, volume 759 of *Lecture Notes in Mathematics.* Springer-Verlag, Berlin, 1979.

[Ers68] Yu. L. Ershov. A hierarchy of sets, I. *Algebra i Logika*, 7(1):47–74, January–February 1968. English Translation, Consultants Bureau, NY, pp. 25–43.

[Gol65] E Mark Gold. Limiting recursion. *JSL*, 30(1):28–48, March 1965.

[Hay78] Louise Hay. Convex subsets of 2^n and bounded truth-table reducibility. *Discrete Mathematics*, 21(1):31–46, January 1978.

[KSW87] Johannes Köbler, Uwe Schöning, and Klaus W. Wagner. The difference and truth-table hierarchies for NP. *RAIRO Theoretical Informatics and Applications*, 21:419–435, 1987.

[Lac65] Alistair H. Lachlan. Some notions of reducibility and productiveness. *Zeitsch. f. math. Logik und Grundlagen d. Math.*, 11:17–44, 1965.

[PR77] E. A. Polyakov and M. G. Rozinas. Enumeration reducibilities. *Siberian Mathematical Journal*, 18(4):594–599, 1977.

[PR79] E. A. Poljakov and M. G. Rozinas. Relationships between different forms of relative computability. *Mathematics of the USSR–Sbornik*, 35(3):425–436, 1979.

[Put65] Hilary Putnam. Trial and error predicates and the solution to a problem of Mostowski. *JSL*, 30(1):49–57, March 1965.

[Rog67] Hartley Rogers, Jr. *Theory of Recursive Functions and Effective Computability.* McGraw Hill, New York, 1967.

[Wag88] Klaus W. Wagner. Bounded query computations. In *Proceedings of the 3rd Annual Conference on Structure in Complexity Theory*, pages 260–277. IEEE Computer Society Press, June 1988.

[WW85] G. Wechsung and K. Wagner. On the Boolean closure of NP. In *Proceedings of the 1985 International Conference on Fundamentals of Computation Theory*, pages 485–493. Springer-Verlag, 1985. Volume 199 of *Lecture Notes in Computer Science.*