**Relativization of the Theory of Computational Complexity**

Nancy Ann Lynch, Albert R. Meyer, Michael J. Fischer

*Transactions of the American Mathematical Society*, Volume 220 (Jun., 1976), 243-287.

Stable URL:
http://links.jstor.org/sici?sici=0002-9947%28197606%29220%3C243%3AROTTOC%3E2.0.CO%3B2-R

TRANSACTIONS OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 220, 1976

# RELATIVIZATION OF THE THEORY OF COMPUTATIONAL COMPLEXITY([1])

BY

NANCY ANN LYNCH, ALBERT R. MEYER AND MICHAEL J. FISCHER

ABSTRACT. The axiomatic treatment of the computational complexity of partial recursive functions initiated by Blum is extended to relatively computable functions (as computed, for example, by Turing machines with oracles). Relativizations of several results of complexity theory are carried out. A recursive relatedness theorem is proved, showing that any two relative complexity measures are related by a fixed recursive function. This theorem allows proofs of results for all measures to be obtained from proofs for a particular measure.

Complexity-determined reducibilities are studied. Truth-table and primitive recursive reducibilities are proved to be reducibilities of this type.

The concept of a set "helping" the computation of a function (by causing a saving in resource when used as an oracle in the computation of the function) is formalized. Basic properties of the helping relation are given, including nontransitivity and bounds on the amount of help certain sets can provide.

Several independence results (results about sets that do not help each other's computation) are proved; they are subrecursive analogs to degrees-of-unsolvability theorems with proofs using diagonalization and priority arguments. In particular, the existence of a "universally-helped set" is discussed; partial results are obtained in both directions. The deepest result in the paper is a finite-injury priority argument (without an apparent recursive bound on the number of injuries) which produces sets preserving an arbitrary lower bound on the complexity of any given set.

## 1. Introduction.

An axiomatic framework for discussing the complexity of partial recursive functions of integer variables has been given by Blum [B1]. We use a similar approach for relatively computable functions (i.e., partial recursive functions of one integer and one set variable). Our relativization of

Received by the editors June 27, 1972 and, in revised form, April 14, 1975.

*AMS (MOS) subject classifications* (1970). Primary 02F15, 68A20, 94A30.

*Key words and phrases.* Complexity theory, Blum measure, relative algorithm, oracle Turing machine, relativization, reducibility, helping, diagonalization, priority argument, convergence argument.

([1]) This paper is based in part on the first author's Doctoral dissertation prepared at the Department of Mathematics, Massachusetts Institute of Technology, under the supervision of the second two authors. The research and preparation of this paper were supported by M.I.T. Project MAC under ONR contract N00014-70-A-0362-0001, by NSF grant numbers GJ-43634 and DCR-92373, and by an NSF graduate fellowship. Final preparation was completed while the first two authors were visiting the IBM Watson Research Center.

Blum's axioms allows us to treat several problems impossible to formulate from his original set of axioms.

For example, we can model the dependence of a computation on a subroutine. We can formalize the idea that knowledge of the values of one function "helps" us to compute a second function.

Our axioms include as special cases the "natural" measures on relative computations, such as the time and space measures on oracle Turing machines [D], [Ro1]. Thus, our theorems are also true when interpreted for the specific measures. We lose a certain amount of precision in determining time and space bounds by proving results axiomatically rather than for specific measures, but we gain independence from the definition of any computing machine model as well as some elegance. Intuitive remarks about time and space are given wherever possible.

In §2 we present our axioms for relative complexity and prove some basic results suggested by theorems of nonrelativized complexity theory.

The first important result is that any two measures satisfying the axioms are recursively related; the proof is by König's lemma. This theorem is important primarily because it provides a method of proof for certain types of theorems; they may be proved for a particular measure (usually space) and then the recursive relatedness between the particular measure and other measures can yield the general result. This method is used occasionally.

We note that the standard results of complexity theory, such as speed-up, compression and gap theorems [HH], all have full relativizations with proofs corresponding to the usual proofs. Several partial relativizations are also true; for instance, we prove a relativization of the combining lemma [HH], which states that the complexity of a computation is closely related to the complexity of its subcomputations. Its proof is an example of our use of a method of proof which we call the convergence method and which is used in axiomatic proofs throughout the paper.

The notion of complexity class [McC], [McCMe], i.e., the class of recursive functions computable within a given time bound, is generalized to a study of "complexity-determined" reducibilities in §3. To any class $C$ of functions corresponds $\{(A, B) \mid A$ is computable from $B$ within measure equal to some function in $C\}$. For certain classes $C$ this defines a natural reducibility. For example, truth-table reducibility [Ro1] and the relation "primitive recursive in" [K] are examples of reducibilities of this type; other commonly-studied reducibilities such as many-one and one-one reducibilities [Ro1] are not.

We show that neither truth-table reducibility nor primitive recursive reducibility can be completely specified by a single bound function (i.e., a singleton class $C$). However, each may be so specified on any countable class of

oracle sets by a relativization of the McCreight-Meyer union theorem [McC], [McCMe].

By selecting special classes of functions $C$, we define new complexity-determined reducibilities and discuss some of their properties.

In §4 we establish a formalism within which to discuss questions such as the following:

(1) Which sets make the computation of a function easier than it would be without the help of these sets?

(2) How much help (increase in speed) does an oracle for a set provide in a computation?

We propose several possible definitions of "helping", each of which provides a reasonable way of discussing the concept. Briefly, we define helping of the computation of a function on either an infinite set of arguments or on almost all (i.e., all but finitely many) arguments. We also discuss helping in the sense of lowering the complexity of a function below a given lower bound function.

We then present a series of basic results. First, we show that any set whose complexity is small cannot give much help to the computation of any function. We then show that any recursive set has arbitrarily complex recursive sets (with their complexity closely determined) that do help its computation.

Following Trahtenbrot ("autoreducible sets") [T1], we formalize the idea of a set which helps its own computation in the sense that values of its characteristic function at different arguments are strongly interdependent. We then prove the existence of such sets with complexity approximately equal to any given monotone running time.

By diagonalization with priorities [Ro1], we next construct a set with no interdependence between the values of its characteristic function at different arguments. Splitting this set into two pieces, we conclude that neither piece can help the other's computation.

This result illustrates proof techniques which will be used in a more complicated fashion in §5. It has several interesting corollaries, including the fact that "helping" is not a transitive relation.

Since the independent sets are constructed by a diagonalization, it is difficult to understand much about them. A more interesting result would arise if we could arbitrarily fix at least one of the sets. Thus, in §§5 and 6 we ask the following question:

Which is true?

(1) For all recursive sets $A$, there exist arbitrarily complex recursive sets $B$ that do not help the computation of $A$, or

(2) There is a recursive set $A$ whose computation is helped by all sufficiently complex recursive sets $B$ ( a "universally-helped set").

In §5 we produce results partially supporting conjecture (1). We first note that the complexity axioms are sufficiently general to be satisfied by various "pathological" measures; specifically, that *any* recursive set will be a "universally-helped set" in *some* relative complexity measure. We use a mechanism for eliminating such trivial cases.

Using diagonalizations with priorities, we prove two theorems whose intended interpretation is that given any recursive set $A$, there exist arbitrarily complex recursive sets $B$ which preserve the complexity of $A$.

Theorem 5.3 states that, given any recursive function $t_A$ such that the complexity of $A$ exceeds $t_A$ infinitely often, there exist arbitrarily complex sets $B$ such that the complexity of $A$ even given an oracle for $B$ still exceeds $t_A$ infinitely often.

Theorem 5.4 is identical to Theorem 5.3 except that "infinitely often" is replaced by "almost everywhere". Also, we require in this case that $t_A$ be a running time (cf. §2), rather than an arbitrary recursive function.

This theorem is our deepest result. The diagonalization required is considerably more complicated than that required for Theorem 5.3, and involves a finite-injury priority argument in which there is no apparent recursive bound on the number of times a requirement may be injured.

These independence results might be interpreted as showing that there exist arbitrarily complex pairs of recursive sets which are complex for "different reasons".

In Theorems 5.3 and 5.4, the sets $B$ which are constructed depend on the particular lower bound $t_A$ which is to be preserved. They may not preserve all lower bounds on the complexity of $A$, however. This might occur if $A$ is a set for which no single lower bound on its complexity is very good (viz., if $A$ has speed-up in the sense of Blum **[B1]**, **[B2]**, so that for every lower bound on the complexity of $A$ there is another lower bound which is much larger infinitely often). Thus conjecture (2) remains consistent with these theorems.

In support of conjecture (2), we sketch in Theorem 6.1 the construction of sets which are helped by all recursive sets whose complexities are compressed around running times.

**2. Notation, axioms and basic results.** We assume familiarity with the basic methods and notation used by Rogers **[Ro1]** for recursive function theory. Some familiarity with the basic results of complexity theory (as surveyed for example in **[HH]**) will be helpful, but we include proof sketches of the particular theorems of the unrelativized theory which are required for our development.

We use $(\forall^{\infty}x)$ and *a.e.* $(x)$ to mean "for all but a finite number of $x$".

When no confusion is likely, we simply write *a.e.* (*almost everywhere*).

Similarly, $(\exists^{\infty}x)$ or *i.o.* $(x)$ means "for infinitely many $x$"; we also write *i.o.* (infinitely often).

We write $a \dot{-} b$ to mean

$$\begin{cases} a - b & \text{if } a \geqslant b, \\ 0 & \text{if } a < b. \end{cases}$$

The composition $g \circ t$ where $t$ is a function of one variable and $g$ is a function of two variables, will indicate $\lambda x[g(x, t(x))]$. That is, $g \circ t(x) = g(x, t(x))$.

$R_n$ represents the set of total recursive functions of $n$ integer variables.

$R_n^{(A)}$ represents the set of total $A$-recursive functions of $n$ integer variables.

$\varphi_i^{(A)}(x)$ denotes the value at argument $x$ of the $i$th function partial recursive in the set $A$ in a standard enumeration.

We write $\uparrow$ for divergence and $\downarrow$ for convergence of computations.

For any $i, A$, if $\varphi_i^{(A)}(x) \uparrow$, we use the convention that $\varphi_i^{(A)}(x) = \infty$.

By convention, $\infty \leqslant \infty$, and $n < \infty$ for any $n \in N$. ($N$ denotes the nonnegative integers.)

$\langle x, y\rangle$ will denote a pairing of $x$ and $y$ in some effective manner, such as $\langle x, y\rangle = \frac{1}{2}(x^2 + 2xy + y^2 + 3x + y)$.

We also write $\pi_1(\langle x, y\rangle) = x$ and $\pi_2(\langle x, y\rangle) = y$. The mapping is a bijection: $x = \langle \pi_1(x), \pi_2(x)\rangle$. Similarly, we let $\langle x_1, \ldots, x_k\rangle$ denote a $k$-ary combination of $x_1, \ldots, x_k$.

$K$ represents the halting set, $\{x \mid \varphi_x^{(\varphi)}(x) \downarrow\}$.

If $A, B$ are sets, then $A$ join $B = \{2x \mid x \in A\} \cup \{2x + 1 \mid x \in B\}$.

If $A$ is any set, $A'$ represents the jump of $A$.

The notions of "relative algorithm" and of an enumeration of partial relatively computable functions (i.e., partial recursive functions of integer and set variables) are amply described in **[Ro1**, §9.2]. Specifically, we use the following

DEFINITION 2.1. A sequence $\{\varphi_i^{(\ )}\}$ of partial relatively computable functions of one integer and one set variable is called *acceptable* if

(1) $\{\varphi_i^{(\ )}\}$ includes all partial relatively computable functions,

(2) Universal Property:

$$(\exists \Psi \in \{\varphi_i^{(\ )}\})(\forall i, x, A)[\Psi^{(A)}(\langle i, x\rangle) = \varphi_i^{(A)}(x)],$$

(3) *s-m-n* Property:

$$(\exists s \in R_2)(\forall i, x, y, A)[\varphi_{s(i,x)}^{(A)}(y) = \varphi_i^{(A)}(\langle x, y\rangle)].$$

We discover by methods analogous to those used in [Ro2] that

LEMMA 2.2. *Let* $\{\varphi_i^{(\ )}\}$ *and* $\{\hat{\varphi}_i^{(\ )}\}$ *be any two acceptable enumerations of partial relatively computable functions. Then there exists a recursive isomorphism* $\alpha$ *such that*

$$(\forall A, i)[\varphi_{\alpha(i)}^{(A)} = \hat{\varphi}_i^{(A)}].$$

Lemma 2.2 will make our theory independent of the particular formalism chosen. We will generally refer to the notion of an oracle Turing machine when precision is required.(2)

We now define a relative complexity measure.

DEFINITION 2.3. A *relative complexity measure* $\Phi^{(\ )}$ is a collection of partial functions from $N$ to $N$, $\{\Phi_i^{(A)}\}$, one for each $(i, A)$, satisfying the following two axioms:

(1) $(\forall i, A)$ [domain $\varphi_i^{(A)}$ = domain $\Phi_i^{(A)}$], and

(2) there exists $\psi^{(\ )}$, a partial relatively computable function, such that

$$(\forall i, x, y, A)\psi^{(A)}(\langle i, x, y\rangle) = \begin{cases} 1 & \text{if } \Phi_i^{(A)}(x) = y, \\ 0 & \text{otherwise.} \end{cases}$$

We abbreviate $\varphi_i^{(\phi)}$ as $\varphi_i$, and $\Phi_i^{(\phi)}$ as $\Phi_i$. The functions $\Phi_i$ are referred to informally as *running times*. There is no confusion here with usual Gödel numbering notation, as $\{\varphi_i^{(\phi)}\}$ is an acceptable Gödel numbering for the partial recursive functions [Ro2].

Thus, for the time measure, where $\Phi_i^{(A)}(x)$ is the number of steps required by machine $i$ on input $x$ with oracle $A$, axiom (1) says that a computation takes a finite amount of time if and only if it converges, and axiom (2) says that one can effectively tell if a computation halts in a given number of steps.

Axiom (1) is established for the space measure (i.e., when $\Phi_i^{(A)}(x)$ is the number of tape squares) by convention, viz., a Turing machine which diverges by looping forever on a bounded number of tape squares is still said to require infinite space. With this convention, axiom (2) then says that one can effectively tell if a computation halts using a given number of tape squares. These axioms will easily be shown to hold for the particular model of oracle Turing machine we introduce below.

It follows immediately from axiom (2) that $\Phi_i^{(A)}(x)$ is a partial relatively recursive function:

---

(2) It is possible to generalize Definition 2.1, allowing oracles for functions as well as sets. A result similar to Lemma 2.2 would still be true in that case. However, the generalization invalidates some of our later proof techniques (e.g. those needed to prove Theorem 2.5) and so we restrict ourselves to set oracles.

REMARK 2.4.

$$(\exists \alpha \in R_1)(\forall i)(\forall A)[\Phi_i^{(A)} = \varphi_{\alpha(i)}^{(A)}].$$

Although our results depend only on these axioms and so are independent of any particular formalization of relative algorithm, enumeration, and measure, it is helpful to keep in mind the natural measures of time and space on oracle Turing machines. The particular oracle Turing machine model we will use is described informally as follows:

Each Turing machine has four semi-infinite tapes: an input tape, an output tape, an oracle tape and a worktape. The first three are marked in binary notation, with the exception that the input tape has markers to indicate the beginning and the end of the input. Initially, the binary representation of input integer $x$ appears on the input tape with the input head scanning the left-most end marker; all other tapes are blank. The worktape uses $k$ symbols for some number $k$ which depends on the machine. We assume that the input head cannot move past the delimiting markers and the output head cannot move left. Also, the machine cannot write on its input tape or read from its output tape. There are otherwise no restrictions on the operation of the machine, other than the usual Turing machine constraints **[Ro1]**.

This Turing machine is designed to be used in conjunction with an "oracle" for any set. (An $X$-oracle is an unspecified agent having information about set $X$.) This is done as follows:

In addition to its other states, the Turing machine has a state called *INTERROGATE*. When the machine enters this state, it asks the oracle whether the number whose binary representation is currently written on the oracle tape is a member of the oracle set. The oracle gives its answer by causing the machine to enter either of two different states. The oracle tape is then automatically erased, the oracle tape head reset to the first tape square, and the computation allowed to continue.

Each oracle Turing machine may be described by a flowchart or some other finite description. The machine's description is independent of the particular oracle set used, so the same oracle machine may be used with any oracle set. The finite descriptions may be enumerated in a natural way so that the index of any machine is an upper bound on its number of states and worktape symbols. We identify $\varphi_n^{(\ )}$ with the $n$th machine description in this enumeration; an enumeration of this kind will be "acceptable", and so there is no notational inconsistency with usage in **[Ro1]**.

We now define two measures on this machine model:

$T^{(\ )}$, *time measure.* For any $i$, $x$, $A$, we define $T_i^{(A)}(x)$ to be the total number of steps executed in the computation $\varphi_i^{(A)}(x)$. Here, each oracle interrogation counts as a single step.

It is clear that the axioms for relative complexity are satisfied; for instance, to discover if $T_i^{(A)}(x) = y$, one need only construct the machine $\varphi_i^{(\ )}$, then simulate $\varphi_i^{(A)}(x)$ for $y$ steps to see if it converges. This procedure is obviously uniform in $A$, $i$, $x$, and $y$, yielding the function $\psi^{(\ )}$ of axiom (2).

$S^{(\ )}$, *space measure.* For any $i$, $x$, $A$, we define $S_i^{(A)}(x)$ to be the maximum of the number of worktape squares visited and the number of oracle tape squares visited during the computation $\varphi_i^{(A)}(x)$, provided that $\varphi_i^{(A)}(x) \downarrow$. Otherwise, we let $S_i^{(A)}(x) = \infty$.

Axiom (1) is satisfied by definition. To see that axiom (2) is also satisfied, we note that for any $i$, $x$, $y$ and $A$, if $\varphi_i^{(A)}(x)$ operates for $(i)$ $(i^y)$ $(y)$ $(2^{y+1})$ $(y)$ $(\lfloor \log_2 x \rfloor + 3)$ steps without exceeding space $y$, it must be in an infinite loop and hence will not converge.([3]) This bound arises since if the machine is ever twice in the same state with the same worktape contents, the same worktape head position, the same oracle tape contents, the same oracle tape head position and the same input tape head position, it must be in an infinite loop. The six factors in the above expression represent bounds on the number of different possibilities for each of the six items.

Thus, to see if $S_i^{(A)}(x) = y$, one need only simulate $\varphi_i^{(A)}(x)$ for $(i)$ $(i^y)$ $(y)$ $(2^{y+1})$ $(y)$ $(\lfloor \log_2 x \rfloor + 3)$ steps to see if it converges.

We note that the space measure has the following property, sometimes called the *parallel computation property* **[LR]**:

There exists a recursive function $\alpha$ such that for all $i$ and $j$,

$$\varphi_{\alpha(i,j)}(x) = \begin{cases} \varphi_i(x) & \text{if } S_i(x) \leqslant S_j(x), \\ \varphi_j(x) & \text{otherwise,} \end{cases}$$

and $S_{\alpha(i,j)}(x) = \min(S_i(x), S_j(x))$.

This property, which in part formally captures the possibility of reusing the same tape squares for different portions of a computation, often makes it easier to prove theorems for the space measure. It also causes some results for space measure to be sharper than those for other measures. We will point out such cases where they occur.

Many of the theorems concerning the complexity of partial recursive functions **[B]**, **[HH]**, **[McC]** have straightforward full relativizations. More interesting and useful are partial relativizations of the results on complexity of partial recursive functions. Following are several examples.

Our first theorem asserts that any two relative complexity measures are related by a fixed recursive function. Its usefulness lies in enabling us to draw conclusions about one relative measure from hypotheses about another relative

([3]) $\lfloor \delta \rfloor$ = the greatest integer $\leqslant \delta$ for any real number $\delta$.

measure, as we do in some of the results following the theorem.

THEOREM 2.5 (RECURSIVE RELATEDNESS). *If* $\Phi^{(\ )}$ *and* $\hat{\Phi}^{(\ )}$ *are two relative complexity measures on the same acceptable Godel numbering* $\{\varphi_i^{(\ )}\}$, *then there exists* $r \in R_2$ *such that*

$$(\forall A, i)[\Phi_i^{(A)} \leqslant r \circ \hat{\Phi}_i^{(A)} \text{ a.e.}], \text{ and}$$

$$(\forall A, i)[\hat{\Phi}_i^{(A)} \leqslant r \circ \Phi_i^{(A)} \text{ a.e.}].$$

PROOF. We use a lemma which is a direct consequence of König's lemma ("Endlichkeitslemma" [**Ro1**, Example 9-40]), and which will be used in several later theorems as well.

LEMMA 2.5.1 (implicit in [A]). *Suppose we have a total recursive function* $f'$ *of* $k$ *integer variables and one set variable. If*

$$(\forall x_1, \ldots, x_k)\left[f(x_1, \ldots, x_k) = \max_{A \subset N} f'(x_1, \ldots, x_k, A)\right],$$

*then* $f \in R_k$.

PROOF OF LEMMA 2.5.1. The computation of $f(x_1, \ldots, x_k)$ may be carried out as follows:

Generate a "computation tree" (cf. [**Ro1**, §9.2]) for a Turing machine computation of the function $f'(x_1, \ldots, x_k, A)$ as $A$ ranges over all subsets of $N$. Each branch of the tree must terminate, since $f'(x_1, \ldots, x_k, A)$ converges for all sets $A$. Therefore by König's lemma the entire tree is finite and we will eventually finish generating it. We can then take the maximum of the outputs on all branches as the value of $f(x_1, \ldots, x_k)$.

PROOF OF THEOREM 2.5, CONTINUED. By symmetry, it suffices to obtain $r \in R_2$ satisfying the first inequality.

We define $r(x, y) = \max_{A \subset N}(\max_{i \leqslant x} r'(x, y, i, A))$, where:

$$r'(x, y, i, A) = \begin{cases} \Phi_i^{(A)}(x) & \text{if } \hat{\Phi}_i^{(A)}(x) = y, \\ 0 & \text{otherwise.} \end{cases}$$

The axioms of Definition 2.3 immediately imply that $r'$ is a total recursive function of three integer variables and one set variable. Therefore, by Lemma 2.5.1, $r \in R_2$.

To see that $r$ has the required properties, consider any particular $A$ and $i$.

If $\hat{\Phi}_i^{(A)}(x)$ diverges, the inequality holds by convention.

If $\hat{\Phi}_i^{(A)}(x)$ converges and $x \geqslant i$, then:

$$r(x, \hat{\Phi}_i^{(A)}(x)) \geqslant r'(x, \hat{\Phi}_i^{(A)}(x), i, A) = \Phi_i^{(A)}(x),$$

as required. Q.E.D.

REMARK 2.5.2. The recursive isomorphism between any two acceptable enumerations of relatively computable functions (Lemma 2.2) allows us to conclude the recursive relatedness of relative complexity measures on two different enumerations. Thus, if $\{\varphi_i^{(\ )}\}$ and $\{\hat{\varphi}_i^{(\ )}\}$ are any two acceptable enumerations of partial relatively computable functions, with relative complexity measures $\Phi^{(\ )}$ and $\hat{\Phi}^{(\ )}$ respectively, and $\alpha$ is the recursive isomorphism in Lemma 2.2, there exists $r \in R_2$ such that

$$(\forall A, i)[\Phi_i^{(A)} \leqslant r \circ \hat{\Phi}_{\alpha(i)}^{(A)} \text{ a.e.}], \text{ and}$$

$$(\forall A, i)[\hat{\Phi}_{\alpha(i)}^{(A)} \leqslant r \circ \Phi_i^{(A)} \text{ a.e.}].$$

The proof is a trivial modification of the proof of Theorem 2.5.

Theorem 2.5 and Remark 2.5.2 provide an alternate method to general axiomatic proof for certain types of theorems about relative complexity measures. The method is to prove the theorem for one measure, and then apply Theorem 2.5 (or Remark 2.5.2) to obtain the result for all measures.

As an example of its use, we give Corollary 2.5.3. The result has two parts; in part (1) we see that (just as in the nonrelativized case) there exist arbitrarily complex functions. However, in contrast to the nonrelativized case, part (2) shows that inherently complex functions cannot be 0-1 valued. In fact, their complexity must result from the *size* of the function values. That is, functions must be as complex as their size, but (given the proper oracle) a function need be no more complex than its size.

First, a definition to simplify notation:

DEFINITION 2.6. Assume $B$ is a set, $f \in R_1^{(B)}$ and $g$ is a total function of one variable.

$\text{Comp}^{(B)} f > g$ *i.o.* (*a.e.*) means $(\forall i)[(\varphi_i^{(B)} = f) \Rightarrow (\Phi_i^{(B)} > g \text{ i.o. (a.e.)})]$.

$\text{Comp}^{(B)} f \leqslant g$ *i.o.* (*a.e.*) means $(\exists i)[\varphi_i^{(B)} = f \text{ and } \Phi_i^{(B)} \leqslant g \text{ i.o. (a.e.)}]$.

$\text{Comp}\, f > g$ *i.o.* means $\text{Comp}^{(\phi)} f > g$ i.o., etc.

If $f = C_A$ (where $C_A$ is the *characteristic function* of a set $A$, defined by $f(x) = 1$ if $x \in A$, 0 if $x \notin A$), we may write Comp $A$ in place of Comp $f$, etc.

COROLLARY 2.5.3. *Let $\Phi^{(\ )}$ be any relative complexity measure. Then*

(1) $(\forall f \in R_1)(\exists g \in R_1)(\forall A)[\text{Comp}^{(A)} g > f \text{ a.e.}]$,

(2) $(\forall h \in R_1)(\exists f \in R_1)(\forall g)[g \leqslant h \text{ a.e.}) \Rightarrow (\exists A)(\text{Comp}^{(A)} g \leqslant f \text{ a.e.})]$.

PROOF. (1) Assume our acceptable enumeration of partial relatively computable functions is the one for oracle Turing machines. Let $r$ be the function obtained by applying Theorem 2.5 to $\Phi^{(\ )}$ and $T^{(\ )}$. We may assume without loss of generality that $r$ is increasing in its second variable.

Given $f$, let $g(x) = 2^{r(x, f(x))+1}$.

If $\varphi_j^{(A)} = g$, then clearly $(\forall x)[T_j^{(A)}(x) > r(x, f(x))]$, since it requires

$r(x, f(x)) + 1$ steps merely to write the output in binary notation.

But $r(x, \Phi_j^{(A)}(x)) \geqslant T_j^{(A)}(x)$ a.e., by Theorem 2.5.

Thus, $r(x, \Phi_j^{(A)}(x)) > r(x, f(x))$ a.e., and since $r$ is increasing, $\Phi_j^{(A)}(x) > f(x)$ a.e., as required.

If the relative complexity measure $\Phi^{(\ )}$ is on an enumeration of relatively computable functions other than that obtained via oracle Turing machines, we apply Remark 2.5.2 in place of Theorem 2.5 and obtain the same result.

(2) Let $r$ be the function obtained by applying Theorem 2.5 to $\Phi^{(\ )}$ and $S^{(\ )}$, again choosing $r$ to be increasing in its second variable. Assume also that the pairing function $\langle x, y\rangle$ is increasing in $y$, and is computable in space $\log_2(\langle x, y\rangle)$. (The pairing function given at the beginning of this section has these properties.)

Let $h$ be given.

Define $f(x) = r(x, \lfloor \log_2\langle x, h(x)\rangle \rfloor + 1)$.

Now consider any $g$ with $g \leqslant h$ a.e.

Let $A = \{\langle x, g(x)\rangle \mid x \in N\}$.

It is straightforward to design a machine $\varphi_j^{(\ )}$ such that $\varphi_j^{(A)} = g$ and for which $S_j^{(A)} \leqslant \lfloor \log_2 \langle x, h(x)\rangle \rfloor + 1$ a.e. Namely, the machine, on argument $x$, operates by successively computing $\langle x, 0\rangle$, $\langle x, 1\rangle$, $\langle x, 2\rangle$, . . . , and asking if each is in $A$. If so, the machine terminates with the appropriate output.

But then

$$\begin{aligned}\Phi_j^{(A)}(x) &\leqslant r(x, S_j^{(A)}(x)) \text{ a.e., by Theorem 2.5}\\ &\leqslant r(x, \lfloor \log_2\langle x, h(x)\rangle \rfloor + 1) \text{ a.e.}\\ &= f(x).\end{aligned}$$

So $\Phi_j^{(A)}(x) \leqslant f(x)$ a.e., as required.

As in (1), if the relative complexity measure $\Phi^{(\ )}$ is on an enumeration of relatively computable functions other than oracle Turing machines, we apply Remark 2.5.2 in place of Theorem 2.5 and obtain the same result. Q.E.D.

Henceforth, whenever we use this method of proof, we will appeal to "recursive relatedness"; it should be understood that we intend this to mean we are applying Theorem 2.5 or Remark 2.5.2, whichever is appropriate, in a fashion similar to that used in the preceding proof.

In §§4, 5, and 6 we will discover ourselves repeatedly using another proof technique which we call the *convergence* method. It is so called because the basic idea of the constructions involves listing just enough recursive conditions to guarantee that the computation in question converges, while not ruling out any cases we wish to consider. The proof of Theorem 2.5 was a simple case of this method. Another example of a convergence argument is now

given in the proof of a relativization of the combining lemma [HH].

As in [Ro1, §5.6], we let $D_k$ represent the finite set with canonical index $k$. The intended interpretation of the lemma is that if there is a uniform algorithm (with index $c(i, j, k)$) for combining the results of machine $\varphi_i^{(\ )}$ with the results of machine $\varphi_j^{(D_k)}$, then the complexity of the combined procedure is not much larger than the complexity of computing the subresults.

LEMMA 2.7 (COMBINING LEMMA). *Let $\Phi^{(\ )}$ be a relative complexity measure. Let $c$ be a function in $R_3$ such that*

$$(\forall i, j, k, x, A)[((\varphi_i^{(A)}(x)\downarrow) \text{ and } (\varphi_j^{(D_k)}(x)\downarrow)) \Rightarrow (\varphi_{c(i,j,k)}^{(A)}(x)\downarrow)].$$

*Then there exists $g \in R_2$ such that for all $i, j, k, A$,*

$$g(x, \max(\Phi_i^{(A)}(x), \Phi_j^{(D_k)}(x))) \geqslant \Phi_{c(i,j,k)}^{(A)}(x) \text{ a.e. } (x).$$

PROOF. Define $g(x, y) = \max_{A \subset N}(\max_{i,j,k \leqslant x} g'(x, y, i, j, k, A))$, where:

$$g'(x, y, i, j, k, A) = \begin{cases} \Phi_{c(i,j,k)}^{(A)}(x) & \text{if } \Phi_i^{(A)}(x) \leqslant y \text{ and } \Phi_j^{(D_k)}(x) \leqslant y, \\ 0 & \text{otherwise.} \end{cases}$$

It is easily seen that $g'$ is total recursive in five integer variables and one set variable. Therefore, by Lemma 2.5.1, $g \in R_2$.

To see that $g$ has the desired properties, we note that if $x \geqslant \max(i, j, k)$, then:

$$g(x, \max(\Phi_i^{(A)}(x), \Phi_j^{(D_k)}(x))) \geqslant g'(x, \max(\Phi_i^{(A)}(x), \Phi_j^{(D_k)}(x)), i, j, k, A) \geqslant \Phi_{c(i,j,k)}^{(A)}(x),$$

as required. Q.E.D.

We now establish a strong relativization of Blum's compression theorem [B1]. The theorem asserts that for any set $B$ and "honest" function $t \in R_1^{(B)}$, there is a set $A$ whose complexity relative to $B$ exceeds $t$ a.e., but is not much more than $t$. Thus the complexity of $A$ is "compressed" about $t$. The set $A$ is constructed by diagonalizing over all $B$-oracle machines which run faster than $t$ i.o.; this guarantees that the complexity of $A$ is as large as required. The time to carry out the diagonal construction of $A$ is not much more than the *time to compute t.*

The hypothesis "$\text{Comp}^{(B)}t \leqslant h \circ t$ (a.e.)" in Proposition 2.8 below will be called an "honesty" condition on $t$. It ensures that *the time to compute t is approximately equal to t*, so that the time to carry out the diagonal construction of $A$ may in fact be bounded by $t$. (The Gap theorem of Borodin [Bo], Constable [Con] and Trachtenbrot (cf.[HH]) shows that some such hypothesis is necessary.) Honesty conditions are extensively studied in [MoMe], [McC].

THEOREM 2.8 (RELATIVIZED COMPRESSION THEOREM). *For any relative complexity measure the following is true*:

$$(\forall h \in R_2)(\exists g \in R_2)(\forall B)(\forall t \in R_1^{(B)})(\exists A)$$
$$[(\mathrm{Comp}^{(B)} t \leqslant h \circ t \text{ a.e.}) \Rightarrow (\mathrm{Comp}^{(B)} A > t \text{ a.e. and } \mathrm{Comp}^{(B)} A < g \circ t \text{ a.e.})].$$

PROOF. The proof is virtually identical to that of Blum [B1] and so we merely sketch the construction informally. (This construction occurs as a special case in the proof of Theorem 4.7 which is carried out formally in §4.)

Let $h$, $t$ and $B$ be given. Because $t$ is honest, there is a procedure to enumerate the graph of $t$ so that $\langle x, t(x)\rangle$ turns up in the enumeration after roughly $h(x, t(x))$ steps. When $\langle x, t(x)\rangle$ turns up, we define $C_A(x) \neq \varphi_i^{(B)}(x)$ where $i$ is chosen to be the least index $\leqslant x$ (if any) such that $\Phi_i^{(B)}(x) \leqslant t(x)$ and $i$ has not already been "cancelled". If such an $i$ is found it is *cancelled* (guaranteeing that $C_A \neq \varphi_i^{(B)}$); if there is no such $i$, then $C_A(x)$ is defined arbitrarily.

It is easy to see that if $\Phi_i^{(B)} \leqslant t$ i.o., then $i$ becomes cancelled and $\varphi_i^{(B)} \neq C_A$. Hence, $\mathrm{Comp}^{(B)} A > t$ a.e. It is also clear (thinking in terms of oracle Turing machines), that the time to compute $C_A(x)$ using this procedure is dominated by the time to simulate $\varphi_0^{(B)}(x), \varphi_1^{(B)}(x), \ldots, \varphi_x^{(B)}(x)$ for $t(x)$ steps each in order to determine which index to cancel. This simulation can be performed in $g(x, t(x))$ steps for some $g \in R_2$ which depends on $h$ but not on $t$, hence $\mathrm{Comp}^{(B)} A < g \circ t$ a.e. Q.E.D.

In subsequent theorems, particularly in §§5 and 6, we shall for simplicity replace honesty conditions on a function $t$ by the hypothesis that $t$ is a running time. There is no loss of generality for our purposes since running times and honest functions are essentially equivalent (cf. [MoMe]). In particular, running times satisfy an honesty condition. This follows informally because if $t$ is the running time of some machine, one can compute $t(x)$ by simulating the machine on input $x$ and counting how many steps occur in the computation. The simulation requires little more than $t(x)$ steps. Formally we can state

LEMMA 2.8.1. *For any measure* $\Phi^{(\ )}$ *there exists* $h \in R_2$ *such that for all sets* $B$ *and* $\Phi_i^{(B)} \in R_1^{(B)}$

$$\mathrm{Comp}^{(B)} \Phi_i^{(B)} \leqslant h \circ \Phi_i^{(B)} \quad \text{a.e.}$$

PROOF. Let $\alpha \in R_1$ be the function such that $\varphi_{\alpha(i)}^{(B)} = \Phi_i^{(B)}$ for all $i$, $B$. Let $c(i, j, k) = \alpha(i)$, so $\varphi_i^{(B)}(x) \downarrow$ implies $\varphi_{c(i,j,k)}^{(B)}(x) \downarrow$. Choose $j_0$, $k_0$ such that

$\Phi_{j_0}^{(D_{k_0})} \in R_1$ and apply the combining lemma to obtain $g \in R_2$ such that for all $i, B$,

$$g \circ \max(\Phi_i^{(B)}, \Phi_{j_0}^{(D_{k_0})}) \geqslant \Phi_{\alpha(i)}^{(B)} \quad \text{a.e.}$$

Let $h(x, y) = g(x, \max(y, \Phi_{j_0}^{(D_{k_0})}(x)))$.

Hence

$$\Phi_{\alpha(i)}^{(B)} \leqslant h \circ \Phi_i^{(B)} \quad \text{a.e.},$$

as required. Q.E.D.

Since the intended interpretation of Theorem 2.8 is that there are arbitrarily complex sets $A$, we must include the observation in the following lemma that there are arbitrarily large running times.

LEMMA 2.8.2. *For any measure $\Phi^{(\ )}$, set $B$, and function $t \in R_1^{(B)}$, there is an increasing running time $\Phi_i^{(B)} \in R_1^{(B)}$ such that $\Phi_i^{(B)} > t$.*

PROOF. By the recursion theorem for acceptable enumerations **[Ro1]**, there is an $i$ such that

$$\varphi_i^{(B)}(x) = \begin{cases} 0 & \text{if } x = 0 \text{ and } \Phi_i^{(B)}(0) > t(0), \\ 0 & \text{if } x > 0, \varphi_i^{(B)}(x-1)\downarrow, \text{ and } \Phi_i^{(B)}(x) > \max(t(x), \Phi_i^{(B)}(x-1)), \\ \infty & \text{otherwise.} \end{cases}$$

It is easy to show by induction on $x$ that $\Phi_i^{(B)}(x)$ has the required properties. Q.E.D.

Finally, we note that any total function $t$ is honest with respect to an appropriate set $B$. That is, there is a $B$ (namely, the graph of $t$) such that $\mathrm{Comp}^{(B)} t \leqslant h \circ t$ a.e. for some $h \in R_2$ independent of $B$ and $t$. The proof is similar to that of Corollary 2.5.3(2) and is omitted. Combining this remark with Theorem 2.8 immediately yields

COROLLARY 2.8.3. *For any complexity measure there is a function $g \in R_2$ such that for any total function $t$,*

$$(\exists A, B \text{ recursive in } t)[\mathrm{Comp}^{(B)} A > t \text{ a.e. and } \mathrm{Comp}^{(B)} A \leqslant g \circ t \text{ a.e.}].$$

3. **Complexity-determined reducibilities.** Corresponding to complexity classes within nonrelativized complexity theory **[McC]**, **[McCMe]**, we may consider measure-bounded relative computation. A fixed measure bound defines a kind of "reducibility" as follows:

DEFINITION 3.1. For any relative complexity measure $\Phi_{\cdot}^{(\ )}$, any sets $A$ and $B$, and any total function $f$ of one variable,

$$A \leqslant_f B(\Phi^{(\ )}) \quad \text{means} \quad \mathrm{Comp}^{(B)}A \leqslant f \text{ a.e.},$$

where complexity is measured in $\Phi^{(\ )}$.

More generally, if $C$ is any class of total functions of one variable,

$$A \leqslant_C B(\Phi^{(\ )}) \quad \text{means} \quad (\exists f \in C)[A \leqslant_f B(\Phi^{(\ )})].$$

We read this notation as "$A$ is $f$-reducible to $B$" and "$A$ is $C$-reducible to $B$", respectively. When no confusion is likely, we omit mention of the measure we are using, and write simply $A \leqslant_f B$ and $A \leqslant_C B$. We note that these reducibilities are not necessarily transitive.

Several commonly-studied reducibilities usually defined via structural restrictions on the method of computation may be expressed as $C$-reducibilities for appropriate choices of the class $C$, and thus may be regarded as complexity-determined. In particular, we shall show that truth-table reducibility [Ro1] and the relation "primitive recursive in" are complexity-determined reducibilities.([4])

We first consider primitive recursive reducibility. We write $A \leqslant_p B$ to indicate that $A$ is primitive recursive in $B$, and $f \leqslant_p B$ to indicate that $f$ is primitive recursive in $B$ [K].

PROPOSITION 3.2. *Let* $\Phi^{(\ )} = T^{(\ )}$ *or* $S^{(\ )}$. *Let* $C = \{$*primitive recursive functions of one variable*$\}$. *Then*

$$(\forall A, B)[(A \leqslant_p B) \Leftrightarrow (A \leqslant_C B(\Phi^{(\ )}))].$$

The proof is a simple relativization of a result of Cobham [C] that a total recursive function is primitive recursive if and only if it is computable on a Turing machine in primitive recursive time or space (see also [MeRD]). Similar observations appear in [A], [RD], [RRW], [Par], [Ma2]; we therefore omit the details.

We now consider truth-table reducibility [Ro1]. Proof methods similar to those used by Nerode [Ro1] combined with recursive relatedness, give the following complexity-determination result for truth-table reducibility; again we omit a proof.

PROPOSITION 3.3. *Fix any relative complexity measure. Let* $C = R_1$. *Then* $(\forall A, B)[A \leqslant_{tt} B \Leftrightarrow A \leqslant_C B]$.

---

([4])On the other hand, many-one and one-one reducibilities are not determined by a complexity restriction, in any relative complexity measure. The reason is that there are pairs of sets computable from each other in a very small measure but which are not many-one reducible to each other (for example, any nonrecursive recursively enumerable set and its complement).

Having noted that primitive recursive and truth-table reducibilities are complexity-determined, we ask if it is possible to express them even more succinctly; for instance, is it possible to characterize each by a single resource bound function rather than a class of functions?

This question immediately suggests that we relativize the union theorem [McC], [McCMe]; by a straightforward modification of McCreight's proof, we obtain the following

THEOREM 3.4 (RELATIVIZED UNION THEOREM). *Assume there is a sequence of total functions* $\{t_i\}$, *such that* $(\forall i, n)[t_{i+1}(n) \geqslant t_i(n)]$.

*Let* $T$ *be a set such that* $\lambda i, n[t_i(n)] \in R_2^{(T)}$.

*Also assume that there is a sequence* $\{B_i\}$ *of sets, and a set* $D$ *such that* $\lambda i, n[C_{B_i}(n)] \in R_2^{(D)}$.

*Then there exists a function* $f \in R_1^{(D \text{ join } T)}$ *such that*

$$(\forall i, j)[\Phi_i^{(B_j)} \leqslant f \text{ a.e.} \Leftrightarrow (\exists k)(\Phi_i^{(B_j)} \leqslant t_k \text{ a.e.})].$$

*(This means that for any* $B \in \{B_j\}$, *the class of functions computable with oracle* $B$ *within measure* $f$ *is exactly the union of the classes of functions computable with oracle* $B$ *within measure* $t_k$, *the union being taken over all* $t_k$.)

PROOF. The construction of $f$ is carried out in stages, with $f(n)$ being defined at stage $n$.

We define an auxiliary function $g(i, j)$, whose values may be changed at successive stages. The significance of $g(i, j)$ is that we "guess" that $\Phi_i^{(B_j)}(x) \leqslant t_{g(i,j)}(x)$ a.e.

*Stage* $n$. (Define $f(n)$.)

For all $(i, j)$ such that $i + j = n$, define $g(i, j) = n$. Let $E = \{(i, j) \mid i + j \leqslant n \text{ and } \Phi_i^{(B_j)}(n) > t_{g(i,j)}(n)\}$.

Define:

$$f(n) = \begin{cases} t_n(n) & \text{if } E = \varnothing, \\ \min\{t_{g(i,j)}(n) \mid (i, j) \in E\} & \text{otherwise.} \end{cases}$$

For all $(i, j) \in E$, redefine $g(i, j) = n$.

Go on to stage $n + 1$.

END OF CONSTRUCTION

Verification that $f$ has the required properties is as in [McCMe]. Q.E.D.

We now apply Theorem 3.4 to the cases of truth-table reducibility and primitive recursive reducibility.

COROLLARY 3.4.1. *Consider any countable collection of sets* $\{B_i\}$ *with* $D$ *as in Theorem* 3.4. *There exists* $f \in R_1^{(D \text{ join } K)}$ *such that*

$$(\forall i, A)[A \leqslant_{tt} B_i \Leftrightarrow A \leqslant_f B_i].$$

PROOF. We define a sequence $\{t_i\}$ as follows: Let

$$t_i(x) = \begin{cases} \max_{j \leqslant i} \{\varphi_j(x) \mid \varphi_j(y) \downarrow \text{ for all } y \leqslant x\} & \text{if this set is nonempty,} \\ 0 & \text{otherwise.} \end{cases}$$

These $t_i$ have the properties required for Theorem 3.4, with $T = K$. Also,

$$(\forall r \in R_1)(\exists j)[r \leqslant t_j \text{ a.e.}] \quad \text{and} \quad (\forall j)(\exists r \in R_1)[t_j \leqslant r \text{ a.e.}].$$

Thus, by Proposition 3.3, if $C = \{t_i\}$, then

$$(\forall A, B)[A \leqslant_{tt} B \Leftrightarrow A \leqslant_C B].$$

Application of Theorem 3.4 now gives the desired result. Q.E.D.

COROLLARY 3.4.2. *Assume the measure is* $S^{(\ )}$ *or* $T^{(\ )}$. *Consider any countable collection of sets* $\{B_i\}$, *with D as in Proposition* 3.4. *There exists* $f \in R_1^{(D)}$ *such that*

$$(\forall i, A)[A \leqslant_p B_i \Leftrightarrow A \leqslant_f B_i].$$

PROOF. Let $\{p_i\}$ be an enumeration of the primitive recursive functions such that $\lambda i, x[p_i(x)]$ is recursive. Then define:

$$t_i(x) = \max_{j \leqslant i} p_j(x).$$

$\{t_i\}$ satisfies the required properties for Theorem 3.4 with $T = \varnothing$. Clearly,

$$(\forall i)[p_i \leqslant t_i \text{ a.e.}], \quad \text{and} \quad (\forall i)(\exists j)[t_i \leqslant p_j \text{ a.e.}].$$

Applying Proposition 3.2 and Theorem 3.4 gives the desired result. Q.E.D.

Thus, we see that for any countable collection of oracle sets (e.g. recursive sets, arithmetical sets), truth-table reducibility is determined by a single resource bound function on any measure, and primitive recursive reducibility is determined by a single resource bound function on measures $T^{(\ )}$ and $S^{(\ )}$.

We note that no single resource bound function can determine either of these two reducibilities on *all* pairs of sets; thus, the countability hypothesis in Corollaries 3.4.1 and 3.4.2 cannot be eliminated:

THEOREM 3.5. *For no relative complexity measure is there a function f of one variable such that* $(\forall A, B)[A \leqslant_{tt} B \Leftrightarrow A \leqslant_f B]$.

PROOF. Assume such a function $f$ exists. We claim that $(\forall r \in R_1)$

$[f > r$ a.e.$]$. For if not, then $(\exists r \in R_1)[r \geqslant f$ i.o.$]$. But then, by Theorem 2.8 (with $B = \varnothing$), there exists a recursive set $A$ such that Comp $A > r$ a.e. We have $A \leqslant_{tt} \varnothing$ since $A$ is recursive, but clearly $\neg(A \leqslant_f \varnothing)$, a contradiction.

Now consider the recursive function $g$ whose existence is asserted by Corollary 2.8.3. We may assume without loss of generality that $g$ is increasing in both variables.

Define a function $t$ as follows:

$$t(x) = \begin{cases} \max\{y \mid g(x, y) \leqslant f(x)\} & \text{if the set is nonempty,} \\ 0 & \text{otherwise.} \end{cases}$$

We claim that $(\forall r \in R_1)[t > r$ a.e.$]$. This is easily concluded from the facts that $(\forall r \in R_1)[f > r$ a.e.$]$ and that $g(x, 1 + t(x)) > f(x)$ by definition.

We now apply Corollary 2.8.3 to obtain $A$ and $B$ recursive in $t$ such that

$$(A \leqslant_{g \circ t} B) \quad \text{and} \quad \neg(A \leqslant_t B).$$

But $A \leqslant_{g \circ t} B$ implies $A \leqslant_f B$, since $g \circ t \leqslant f$ a.e. Also, $\neg(A \leqslant_t B)$ implies $\neg(A \leqslant_{tt} B)$ since $t$ is almost everywhere greater than each recursive function.

Thus, $f$ does not determine truth-table reducibility on all pairs of sets recursive in $t$. Q.E.D.

In an entirely analogous way, we obtain

REMARK 3.6. For the space or time measures on oracle Turing machines there is no function $f$ of one variable such that:

$$(\forall A, B)[A \leqslant_p B \Leftrightarrow A \leqslant_f B].$$

We have seen that some reducibilities with structural definitions may be alternatively described by a complexity restriction. Conversely, it is possible to define new reducibilities by a complexity restriction. We give an example of such a definition, and note some properties of the resulting reducibilities.

DEFINITION 3.7. For any sets $A$, $B$, $C$, we say *$A$ is $C$-reducible to $B$* $(A \leqslant_C B)$ provided: $A \leqslant_{\mathcal{C}} B$ for $\mathcal{C} = R_1^{(C)}$. We write *$C$-reducibility* to indicate $\{(A, B) \mid A \leqslant_C B\}$.

Thus, any set $C$ determines a new reducibility, namely, the collection of pairs of sets computable from each other in $C$-recursive measure. The reducibilities are clearly measure-invariant; they are also transitive:

LEMMA 3.8. *For any sets $A$, $B$, $C$ and $D$,*

$$[A \leqslant_C B \text{ and } B \leqslant_C D] \Rightarrow [A \leqslant_C D].$$

PROOF. By measure-invariance of $C$-reducibility and closure of $R_1^{(C)}$

under finite modification, the hypotheses $A \leqslant_C B$ and $B \leqslant_C D$ imply:

$$(\exists i)(\exists c_1 \in R_1^{(C)})[C_A = \varphi_i^{(B)} \text{ and } S_i^{(B)} \leqslant c_1] \quad \text{and}$$

$$(\exists j)(\exists c_2 \in R_1^{(C)})[C_B = \varphi_j^{(D)} \text{ and } S_j^{(D)} \leqslant c_2].$$

We describe an oracle Turing machine which computes $C_A$ using a $D$-oracle.

The Turing machine computes $C_A$ according to procedure $\varphi_i^{(B)}$, but a value about which the $B$-oracle is queried is written on a second track of the worktape instead of the oracle tape. Then, to decide its membership in $B$, $\varphi_j^{(D)}$ is applied on this second track using the available $D$-oracle.

For input $x$, the machine uses $S_i^{(B)}(x)$ space to carry out the computation $\varphi_i^{(B)}(x)$. In addition, the largest argument for which it might need to compute $C_B$ is ${}_2S_i^{(B)}(x)$.

Thus, the space needed is bounded above by the maximum of

$$S_i^{(B)}(x), S_j^{(D)}(0), \ldots, S_j^{(D)}({}_2S_i^{(B)}(x)),$$

which in turn is bounded above by the maximum of $c_1(x), c_2(0), \ldots, c_2({}_2c_1(x))$. But this maximum is a function in $R_1^{(C)}$. Thus, $A \leqslant_C D$. (An axiomatic version of this argument is given in Theorem 4.2.) Q.E.D.

The $C$-reducibilities form a partial ordering of reducibilities, ordered by containment, between truth-table and Turing reducibilities. We may show that containment of reducibilities is exactly determined by size of functions. Namely,

THEOREM 3.9. *For any sets C, D,*

$$(C\text{-reducibility} \subset D\text{-reducibility}) \Leftrightarrow (\forall f \in R_1^{(C)})(\exists g \in R_1^{(D)})[g \geqslant f].$$

PROOF. ($\Leftarrow$) Obvious.

($\Rightarrow$) If the right-hand side fails to hold, then $(\exists f \in R_1^{(C)})(\forall g \in R_1^{(D)})$ $[g < f \text{ i.o.}]$.

Then by Lemma 2.8.2 there is a running time $\Phi_i^{(C)} > f$, so by Lemma 2.8.1 and Theorem 2.8,

$$(\exists A, \text{ recursive in } C)[\text{Comp}^{(C)}A > f \text{ a.e.}].$$

Hence $A \leqslant_C C$ but $\neg(A \leqslant_D C)$, a contradiction to the left-hand side. Q.E.D.

Clearly, we have

COROLLARY 3.9.1. *For any sets C, D,*

$$(C \equiv_T D) \Rightarrow (C\text{-reducibility} = D\text{-reducibility}).(5)$$

The converse of Corollary 3.9.1 is true for a large collection of sets $C$ and $D$, but not for all sets $C$ and $D$:

DEFINITION 3.10. A set $A$ is *weakly majoreducible* if there exists $f \in R_1^{(A)}$ such that

$$(\forall g, \text{ total functions of one variable})[g \geqslant f \Rightarrow A \text{ is recursive in } g].$$

This definition is weaker than, although similar to, the definition of "majoreducible" used in [J].

THEOREM 3.11. *If sets $C$ and $D$ are weakly majoreducible, then*

$$(C\text{-reducibility} = D\text{-reducibility}) \Rightarrow (C \equiv_T D).$$

PROOF. If $C$-reducibility = $D$-reducibility, then by Theorem 3.9,

$$(\forall f \in R_1^{(C)})(\exists g \in R_1^{(D)})[g \geqslant f].$$

By weak majoreducibility of $C$, $C$ is recursive in $g$ for the appropriate choice of $f$.

Therefore, $C$ is Turing-reducible to $D$.

Symmetrically, we have $D$ is Turing-reducible to $C$. Q.E.D.

COROLLARY 3.11.1. *If sets $C$ and $D$ are recursively enumerable, then* $(C\text{-reducibility} = D\text{-reducibility}) \Leftrightarrow (C \equiv_T D)$.

PROOF. It follows immediately from work in [J] that all recursively enumerable sets are majoreducible (according to his definition) and hence weakly majoreducible. The reason is as follows:

If $\{c_i\}$ is an effective enumeration, without repetitions, of an infinite recursively enumerable set $C$, and if $f$ is defined by

$$f(n) = \mu z[(\forall y)(y > z \Rightarrow c_y > n)],$$

then clearly $f \in R_1^{(C)}$, and $C$ is recursive in any $g \geqslant f$. Q.E.D.

However, we may obtain the following

THEOREM 3.12. *Given any set $C$, there exist two sets $A$ and $B$ such that* $A|_T B$ *and* $A\text{-reducibility} = B\text{-reducibility} = C\text{-reducibility}$.(6)

PROOF. A detailed proof is omitted because it is long and not very differ-

(5) $\equiv_T$ designates Turing equivalence.

(6) $|_T$ designates incomparability under Turing-reducibility.

ent from other proofs in the literature. A modified version [Ma1] of Spector's splitting-tree construction of minimal sets [Ro1] produces a nonrecursive set $A$ which is "small" rather than minimal. That is,

$$(\forall f \in R_1^{(A)})(\exists g \in R_1)[g \geqslant f].$$

To prove Theorem 3.12, we simultaneously construct two "small" sets, $A$ and $B$, by modified splitting-tree constructions, with two added changes:

(1) We encode $C$ into both sets at the beginning of the construction.

(2) We alternate the splitting-tree construction with a diagonalization making $A$ and $B$ Turing incomparable.

Theorem 3.9 gives the required result. Q.E.D.

Although pairs of sets can have the same reducibility and still be Turing-incomparable, there do exist limits on what Turing-reducibility relationships sets can have and still determine the same reducibility. For example

THEOREM 3.13. *If $C'$ is Turing-reducible to $D$, then $C$-reducibility $\neq$ $D$-reducibility.*

PROOF. Define

$$g(n) = \max\{\varphi_i^{(C)}(n) \mid 0 \leqslant i \leqslant n \text{ and } \varphi_i^{(C)}(n) \downarrow\}.$$

Then $g \in R_1^{(C')}$, so that $g \in R_1^{(D)}$. But clearly $(\forall f \in R_1^{(C)})[g > f \text{ a.e.}]$. So by Theorem 3.9, $C$-reducibility $\neq$ $D$-reducibility. Q.E.D.

4. **Helping.** Intuitively, it is clear that some sets $B$ help to compute some functions $f$. That is, when $B$ is used as an oracle, the complexity of $f$ is smaller than it was without the oracle for $B$.

In this section we formalize this observation. We use the word "helping" in informal discussions only, and give precise meanings to several interpretations. We also give basic results about the existence of sets which help or do not help the computation of certain functions.

DEFINITION 4.1. Assume $B$ is a set, $f \in R_1$ and $s$ is a total function of two variables.

*$B$ s-improves $f$ i.o.* (a.e.) means:

$$(\exists t \in R_1^{(B)})[\text{Comp}^{(B)} f \leqslant t \text{ a.e. and Comp } f > s \circ t \text{ i.o. (a.e.)}].$$

We remark that these definitions do not provide us with notions of helping that are transitive or symmetric, which is perhaps surprising for the case of improvement a.e. Appropriate counterexamples will be given later in this section.

To place these definitions in some perspective, it is helpful to note a relationship between "$A \leqslant_p B$" and "$B$ $s$-improves $A$ (i.o.)": if $A$ is not primitive

recursive and $A \leqslant_p B$, then $B$ $s$-improves $A$ i.o. in the space measure for any primitive recursive function $s$.

To verify this, note that if $A \leqslant_p B$, then by Proposition 3.2 $\mathrm{Comp}^{(B)}A \leqslant f$ a.e. for some primitive recursive $f$. If $A$ is not primitive recursive, then $\mathrm{Comp}(A) > g$ i.o. for any primitive recursive $g$ as we noted following Proposition 3.2. In particular, if $s$ is primitive recursive, so is $s \circ f$ and therefore $\mathrm{Comp}\, A > s \circ f$ i.o. Hence $B$ $s$-improves $A$ i.o. (This observation is also implicit in [Ma2].)

The amount of help a recursive oracle $B$ is able to give the computation of a function is restricted by the complexity of $B$. This is because any program using $B$ as an oracle may be converted to one not using the $B$-oracle, by directly computing the answers to the oracle queries. The complexity of the new program is bounded as follows:

THEOREM 4.2. *There exist $g_1 \in R_3$ and $g_2 \in R_2$ such that for all recursive sets $B$ and $f, t, t' \in R_1$, if*

$$\mathrm{Comp}^{(B)}f \leqslant t \text{ a.e.} \quad \text{and} \quad \mathrm{Comp}\, B \leqslant t' \text{ a.e.},$$

*then*

$$\mathrm{Comp}\, f \leqslant \lambda x[g_1(x, t(x), \max\{t'(y) \mid y \leqslant g_2 \circ t(x)\})] \text{ a.e.}$$

PROOF. The proof is a convergence argument.

We use the following general lemma which provides a bound on the portion of the oracle on which a computation and its measure may depend:

LEMMA 4.2.1 (implicit in [A]). *Fix any acceptable enumeration of relative algorithms $\{\varphi_i^{(\ )}\}$ and any relative complexity measure $\Phi^{(\ )}$. Then*

$$(\exists h \in R_3)(\forall i, x, y, A, B)$$
$$[(A \cap \{0, \ldots, h(i, x, y)\} = B \cap \{0, \ldots, h(i, x, y)\} \text{ and } \Phi_i^{(A)}(x) \leqslant y) \Rightarrow (\varphi_i^{(A)}(x) = \varphi_i^{(B)}(x) \text{ and } \Phi_i^{(A)}(x) = \Phi_i^{(B)}(x))].$$

PROOF OF LEMMA 4.2.1. Let $\{\hat{\varphi}^{(\ )}\}$ be the enumeration of relatively computable functions arising from oracle Turing machines.

Choose $\alpha \in R_1$ (by Lemma 2.2) so that $\hat{\varphi}^{(\ )}_{\alpha(i)} = \varphi_i^{(\ )}$.

Choose $\beta \in R_1$ (by Lemma 2.2 and remarks following Definition 2.3) so that $\hat{\varphi}^{(\ )}_{\beta(i)} = \Phi_i^{(\ )}$.

Define $h(i, x, y) = \max_{X \subset N} h'(i, x, y, X)$, where

$$h'(i, x, y, X) = \begin{cases} 0 & \text{if } \Phi_i^{(X)}(x) > y, \\ \text{the largest number whose membership in } X \text{ is questioned} & \\ \quad \text{in either computation } \hat{\varphi}^{(X)}_{\alpha(i)}(x) \text{ or } \hat{\varphi}^{(X)}_{\beta(i)}(x), & \text{otherwise.} \end{cases}$$

$h'$ is total recursive since any convergent oracle Turing machine computation is determined by answers to a finite set of questions, so that $h \in R_3$ by Lemma 2.5.1.

We leave to the reader the verification that $h$ has the required properties.

PROOF OF THEOREM 4.2, CONTINUED. If $B$ is a recursive set, then $B$-oracle machines can be transformed effectively into equivalent machines without oracles. Formally, there is a $\gamma \in R_2$ such that

$$(\forall a, b, B)[\varphi_b = C_B \Rightarrow \varphi_a^{(B)} = \varphi_{\gamma(a,b)}].$$

Namely, for any $z$ let $z_1, z_2, z_3, z_4$ be such that $z = \langle z_1, z_2, z_3, z_4 \rangle$, let $h \in R_3$ be the function given in Lemma 4.2.1, and obtain $\gamma$ by the *s-m-n* theorem such that

$$\varphi_{\gamma(a,b)}(x) = \pi_1(\mu z[\Phi_a^{(D_{z_2})}(x) = z_3 \text{ and } \varphi_a^{(D_{z_2})}(x) = z_1 \text{ and}$$

$$(\forall w \leqslant h(a, x, z_3))[\Phi_b(w) \leqslant z_4 \text{ and}$$

$$(w \in D_{z_2} \Leftrightarrow \varphi_b(w) = 1)]]).$$

Now define a function $g'$ such that

$$g'(x, y, z, a, b, B) = \begin{cases} \Phi_{\gamma(a,b)}(x) & \text{if } \Phi_a^{(B)}(x) = y \text{ and } (\forall w \leqslant h(a, x, y)) \\ & [\Phi_b(w) \leqslant x \text{ and } (w \in B \Leftrightarrow \varphi_b(w) = 1)], \\ 0 & \text{otherwise.} \end{cases}$$

Clearly $g'$ is recursive, so letting $g(x, y, z) = \max_{a,b \leqslant x, B \subset N} g'(x, y, z, a, b, B)$, we conclude by Lemma 2.5.1 that $g \in R_3$.

Now let $B, f, t, t'$ be as in the hypotheses, and let $a, b$ be indices such that

$$\varphi_a^{(B)} = f \text{ and } \Phi_a^{(B)} \leqslant t \text{ a.e., and } \varphi_b = C_B \text{ and } \Phi_b \leqslant t' \text{ a.e.}$$

Then by definition of $\gamma$, $\varphi_{\gamma(a,b)} = f$ and by definition of $g$,

$$\Phi_{\gamma(a,b)}(x) \leqslant g(x, \Phi_a^{(B)}(x), \max\{\Phi_b(y) \mid y \leqslant h(a, x, \Phi_a^{(B)}(x))\})$$

for almost all $x$.

Let $g_1(x, y, z) = \max\{g(u, v, w) \mid u, v, w \leqslant \max\{x, y, z\}\}$ and let $g_2(x, y) = \max\{h(u, v, w) \mid u, v, w \leqslant \max\{x, y\}\}$. Then

$$\Phi_{\gamma(a,b)}(x) \leqslant g_1(x, t(x), \max\{t'(y) \mid y \leqslant g_2 \circ t(x)\})$$

for almost all $x$. Hence, Comp $f$ is bounded a.e. as required. Q.E.D.

We next note that for any sufficiently complex recursive set $A$, there exist arbitrarily complex recursive sets $B$ that *do* help the computation of $C_A$; in fact, $\text{Comp}^{(B)}A$ can be reduced below a small bound. We may further specify that the set $B$ be "compressed" (i.e., $B$'s complexity is very closely determined, to

within a fixed amount $g$ depending on the measure only).

THEOREM 4.3. *Let $\Phi^{(\ )}$ be any relative complexity measure. There is a function $g \in R_2$ with the following property:*

*Let $t$ be any total, unbounded, monotone nondecreasing running time.*(7)

*Let $A$ be any recursive set such that* Comp $A \leqslant t$ *a.e.*

*Then there exists a recursive set $B$ with*

$$\text{Comp } B > t \text{ a.e.}, \qquad \text{Comp } B \leqslant g \circ t \text{ a.e., and } A \leqslant_\rho B.$$

PROOF. The proof is a convergence argument. The construction is carried out in stages, using a diagonal construction similar to that of Theorem 2.8 with one modification: a smaller number of indices are tested for cancellation at each stage in constructing $B$, so most values of $B$ are not used for cancelling indices. We use the remaining arguments to encode $A$ in a primitive recursive way.

We define a function $f$ as follows:

$$f(0) = 0,$$

$$f(n) = \lfloor \sqrt{n} \rfloor - 1 \quad \text{for all } n \geqslant 1.$$

By the *s-m-n* theorem, we can define a partial recursive function $\varphi_{\alpha(a,b)}$ where $\alpha \in R_2$ according to the following construction in stages:

*Stage n*: (Define $\varphi_{\alpha(a,b)}(n)$.)

Find the smallest uncancelled $i \leqslant f(n)$ such that $\Phi_i(n) \leqslant \Phi_b(n)$.

(Diverge if $\Phi_b(n) \uparrow$.)

If no such $i$ exists, define $\varphi_{\alpha(a,b)}(n) = \varphi_a(f(n))$.

If $i$ exists, define $\varphi_{\alpha(a,b)}(n) = 1 \dot{-} \varphi_i(n)$ and *cancel i.*

Go on to stage $n + 1$.

END OF CONSTRUCTION

We define $g(x, y) = \max_{a,b \leqslant x} g'(x, y, a, b)$, where

$$g'(x, y, a, b) = \begin{cases} \Phi_{\alpha(a,b)}(x) & \text{if } (\forall w \leqslant x)[\Phi_b(w) \leqslant y \text{ and } \Phi_a(w) \leqslant y], \\ 0 & \text{otherwise.} \end{cases}$$

If $A$ and $t$ are as in the hypotheses, and we choose $a^*, b^*$ with $\Phi_{b^*} = t$, $\varphi_{a^*} = C_A$ and $\Phi_{a^*} \leqslant t$ a.e., then we claim that $C_B = \varphi_{\alpha(a^*,b^*)}$ has the desired properties.

---

(7) As mentioned in the remark following Lemma 2.7, we use the simplifying assumption that $t$ is a running time in place of a general honesty hypothesis.

For example, $A \leqslant_p B$ since for any $n$,

$$C_A(n) = \begin{cases} 1 & \text{if } \sum_{x=(n+1)^2}^{(n+2)^2-1} C_B(x) \geqslant n+2, \\ 0 & \text{otherwise.} \end{cases}$$

The remaining properties are left for the reader to verify. Q.E.D.

If we relax the upper bound on Comp $B$ in Theorem 4.3 a much simpler construction suffices:

DEFINITION 4.4. For any sets $X$, $Y$, we define $X \oplus Y$ to be the symmetric difference $(X - Y) \cup (Y - X)$.

For any functions $f$, $g$, we define $f$ *join* $g$ by

$$f \text{ join } g(2x) = f(x), \qquad f \text{ join } g(2x+1) = g(x).$$

Then if we take $t' \in R_1$ with $t'(x)$ sufficiently larger (depending on the measure) than $t(2x)$ and $t(2x+1)$, we may choose any recursive set $C$ with Comp $C > t'$ a.e. and let $B = C$ join $(A \oplus C)$. This set $B$ has the properties $A \leqslant_p B$ and Comp $B > t$ a.e. This second property is easily shown for space measure using the parallel computation property, and recursive relatedness gives the result for general measures; we omit the details.

Results in this section have so far been rather intuitive; less so are results stating "independence" of sets (for example, demonstrating the existence of pairs of recursive sets which do not help each other's computation).

Our solutions to problems of this latter type are analogous to work on degrees of unsolvability [Sa], [**Ro1**, §10.2, Chapter 13] in the following sense:

Independence proofs proceed by a diagonalization (the only general tool we have thus far for proving such results). We require a countable sequence of conditions, or perhaps two different countable sequences of conditions, to be satisfied. Satisfaction of these various conditions may cause conflict. To ensure that each condition gets satisfied, we establish before the construction a "priority ordering" of conditions; in our theorems, this is a simple numerical ordering in which lower numbered conditions are given higher priority.

We allow the satisfaction of a condition to be interrupted only by switching to an attempt to satisfy a higher-priority (lower numbered) condition. It follows that once we begin trying to satisfy some condition, we must thereafter succeed in satisfying either *that* condition or one of higher priority; thus, all conditions will eventually become satisfied.

Our arguments are more complex than the "initial segment" constructions in [**Ro1**, Chapter 13]; we do construct our sets by determining values first on

initial segments, but we also carry with us "tentative commitments" to definition of the set at arguments a finite distance beyond the defined initial segment.

Our constructions differ from those in [Sa] and [Ro1], however; we are constructing recursive sets and our constructions are always effective.

After a degree-of-unsolvability priority construction, the oracles used in the construction are usually pinpointed, thereby placing the constructed set in its proper Turing degree. We are working with a subrecursive analog of these constructions and are generally interested in the complexity of the resulting set. Thus, we follow our constructions with arguments showing what subcomputations were used in the computation constructing our set, thereby placing the set in its proper complexity class.

We now prove an independence theorem. In order to make the proof as compact as possible, we first introduce definitions designed to allow us to discuss the independence of the values of a 0-1 valued function at its different arguments. In Theorem 4.6 we give an example of a simple theorem using this definition. Theorem 4.7 shows the existence of a 0-1 valued recursive function, i.e. a set, whose values at its different arguments are independent, while Theorem 4.8 shows how to split this type of set into two sets which do not help each other's computation, thus giving a complexity-theoretic analog to the Friedberg-Muchnik theorem [Ro1, §10.2].

Theorem 4.7 was announced by Trahtenbrot [T1], and easily implies Theorem 4.8 which was discovered independently and announced by the second two authors [MeF2]. The proofs given here are our own.

DEFINITION 4.5. Assume $A$ is a recursive set and $g$ is a total function of one variable. Then:

$$\underline{\mathrm{Comp}^{[A]}A > g \text{ a.e.}} \text{ means:}$$

$$(\forall i)[(\forall x)(\varphi_i^{A-\{x\}}(x) = C_A(x)) \Rightarrow (\forall^\infty x)(\Phi_i^{A-\{x\}}(x) > g(x))]$$

$$\underline{\mathrm{Comp}^{[A]}A \leqslant g \text{ a.e.}} \text{ means:}$$

$$(\exists i)[(\forall x)(\varphi_i^{A-\{x\}}(x) = C_A(x)) \text{ and } (\forall^\infty x)(\Phi_i^{A-\{x\}}(x) \leqslant g(x))].$$

The following theorem pointed out to us by M. Paterson shows the abundance of 0-1 valued functions whose values at different arguments are strongly dependent. This essentially settles a question raised by Trahtenbrot [T1].([8])

([8]) Trahtenbrot asked whether Theorem 4.6 held for the space measure with $g$ approximately linear. A positive solution follows by a straightforward implementation on a Turing machine of the axiomatic construction given above.

THEOREM 4.6 (Paterson). *There exists* $r \in R_1$, $g \in R_2$ *with the following property*:

*Whenever* $t$ *is an increasing running time, there exists a recursive set* $A$ *such that*

$$\text{Comp } A > t \text{ a.e.}, \quad \text{Comp } A \leqslant g \circ t \text{ a.e., and } \text{Comp}^{[A]} A \leqslant r \text{ a.e.}$$

PROOF. We define the set $A$, depending on $t$; we indicate how to construct $r$ and $g$ afterwards.

We define $A$ by a construction in stages. Indices $i$ are cancelled as the construction ensures that $\varphi_i \neq C_A$.

Let $f(y) = \lfloor y/4 \rfloor$.

*Stage* $x$. (Define $C_A(x)$.)

See if there exists an uncancelled $i < f(x)$ such that $\Phi_i(x) \leqslant t(x)$.

1. If so, choose the least such $i$ and define $C_A(x) = 1 \dot{-} \varphi_i(x)$.
   *Cancel* $i$.
   Go on to stage $x + 1$. (In this case we say that $x$ was *used to cancel* index $i$.)

If no such $i$ exists,

2.1. If $|\{y \mid y < x \text{ and } y \in A\}|$ is even, we define $C_A(x) = 0$.
   Go on to stage $x + 1$.

2.2. Otherwise, define $C_A(x) = 1$.
   Go on to stage $x + 1$.

(In case 2, we say that $x$ was *used to maintain parity*.)

END OF CONSTRUCTION

$A$ is clearly recursive. Substage 1 ensures that Comp $A > t$ a.e. as follows:

Assume for some index $i$ that $\Phi_i \leqslant t$ i.o. There exists a stage $x_0$ in the construction of $A$ after which all indices smaller than $i$ which are ever cancelled have already been cancelled. Thereafter, when it happens for some $x > \max\{x_0, 4i\}$, that $\Phi_i(x) \leqslant t(x)$, the procedure defines $C_A(x) \neq \varphi_i(x)$.

Verification of the second claim depends on the construction of the proper $g$, which may be done as in previous proofs by a convergence argument.

To verify the third claim, we use the following procedure for obtaining $C_A(x)$ from $C_A$ on other arguments:

Let $j$ be chosen so that for all $x$, $B$,

$$\varphi_j^{(B)}(x) = \begin{cases} 0 & \text{if there are more integers } y,\ x+1 \leqslant y \leqslant 2x \text{ for which} \\ & |(\{0, \ldots, y\} - \{x\}) \cap B| \text{ is even than for which} \\ & |(\{0, \ldots, y\} - \{x\}) \cap B| \text{ is odd,} \\ 1 & \text{otherwise.} \end{cases}$$

Then we have that $\varphi_j^{(A-\{x\})}(x) = C_A(x)$ for all $x$. This follows from the fact that among arguments $y$ with $x + 1 \leqslant y \leqslant 2x$, fewer than $f(2x)$ were used to cancel indices, so more than half were used to maintain parity.

From this procedure, it is easy to construct the function $r$, namely $r(x) = \max_{B \subset N}(\Phi_j^{(B)}(x))$. Q.E.D.

THEOREM 4.7 (Trahtenbrot [T1]). *There exists $g \in R_2$ with the following property: For any total running time $t$, there exists a recursive set $A$ with*

$$\text{Comp } A \leqslant g \circ t \text{ a.e., and } \text{Comp}^{[A]} A > t \text{ a.e.}$$

PROOF. We would like to ensure

$$(\forall i)[(\Phi_i^{(A-\{x\})}(x) \leqslant t(x) \text{ i.o.}) \Rightarrow (\exists y)(\varphi_i^{(A-\{y\})}(y) \neq C_A(y))].$$

As before, we use cancellation; an index $i$ is cancelled when the construction has ensured that

$$(\exists y)(\varphi_i^{(A-\{y\})}(y) \neq C_A(y)).$$

In addition, at any time during the construction, a single index may be "tentatively cancelled". If an index $i$ is tentatively cancelled, it means that the process of attempting to cancel $i$ by defining $A$ according to an appropriate "tentative commitment" is underway. If the process succeeds in defining $A$ in this way, then $i$ will be cancelled; otherwise, the tentative cancellation of $i$ will be removed.

We construct $A$ in stages beginning at stage 0. A parameter "$a$" is used effectively in the construction, so by the *s-m-n* theorem there is a recursive function $\alpha$ such that $\varphi_{\alpha(a)}$ is the function defined by the construction. For $\Phi_a = t$, the function $\varphi_{\alpha(a)}$ will turn out to be the $C_A$ of our theorem. (We use the parameter "$a$" to allow us later to obtain the desired recursive function $g$ by a convergence argument.)

If $m = \langle n, \Phi_a(n)\rangle$, then at stage $m$ we will define $\varphi_{\alpha(a)}(n)$. This guarantees that, for any $n$, the value $\varphi_{\alpha(a)}(n)$ is defined at most at one stage, and that if $\Phi_a(n) \downarrow$, then $\varphi_{\alpha(a)}(n)$ will be defined.

*Stage m.* Let $n = \pi_1(m)$; if $\Phi_a(n) \neq \pi_2(m)$, go to stage $m + 1$. Otherwise, define $\varphi_{\alpha(a)}(n)$ as follows:

Find the smallest $i \leqslant n$ that is not yet cancelled and such that

(a) if some index $j$ is tentatively cancelled, then $i < j$, and

(b) there exists $E$ such that

(b1) $E \subset \{x \mid x \leqslant h(i, n, \Phi_a(n))$ or $\varphi_{\alpha(a)}(x)$ has already been defined$\}$, where $h$ is the function whose existence is asserted by Lemma 4.2.1,

(b2) $(\forall x \mid \varphi_{\alpha(a)}(x)$ has already been defined$)[x \in E \Leftrightarrow \varphi_{\alpha(a)}(x) = 1]$,

(b3) $n \notin E$, and

(b4) $\Phi_i^{(E)}(n) \leqslant \Phi_a(n)$.

1. If such an $i$, $E$ exist, remove any previous tentative cancellation and tentative commitment. (There can only be one.)

Define $\varphi_{\alpha(a)}(n) = 1 \dot{-} \varphi_i^{(E)}(n)$. *Tentatively cancel* $i$ and let $(E, n)$ be a *tentative commitment* (the aim will be to define $\varphi_{\alpha(a)}$ so that

$$(\forall x \mid x \leqslant h(i, n, \Phi_a(n)) \text{ and } x \neq n)[\varphi_{\alpha(a)}(x) = C_E(x)].$$

Go on to stage $m + 1$.

2. If no such $i$ exists,

2.1. If some index $j$ is tentatively cancelled, let $(E', n')$ be the associated tentative commitment.

Define $\varphi_{\alpha(a)}(n) = C_{E'}(n)$.

2.1.1. If $(\forall x \leqslant h(j, n', \Phi_a(n')))$ [$\varphi_{\alpha(a)}(x)$ has already been defined], then remove $j$'s tentative commitment. Change $j$'s tentative cancellation to a *cancellation*. Go on to stage $m + 1$.

2.1.2. Otherwise, just go on to stage $m + 1$.

2.2. If no index $j$ is tentatively cancelled, just define $\varphi_{\alpha(a)}(n) = 0$.

Go on to stage $m + 1$.

END OF CONSTRUCTION

Now assume we have $t$ as in the hypotheses. If we choose $a^*$ with $\Phi_{a^*} = t$, then we claim that $C_A = \varphi_{\alpha(a^*)}$ has the desired properties:

*$A$ is a recursive set.* Since $\Phi_{a^*}$ is total, all stages are executed. The definition of each stage is clearly effective.

$\mathrm{Comp}^{[A]}A > t$ *a.e.* We make an observation about cancellations which may be proved by induction: If an integer $k$ is tentatively cancelled at some stage, then at that stage or later, some integer $\leqslant k$ will become cancelled.

Now for any index $i$, suppose that $(\forall x)[\varphi_i^{(A-\{x\})}(x) = C_A(x)]$.

Then we observe that $i$ is never cancelled. For if $i$ were cancelled, there was some last stage $m = \langle n, \Phi_{a^*}(n)\rangle$ at which some tentative commitment $(E, n)$ for $i$ was made. At stage $m$, substage 1 defines $C_A(n) \neq \varphi_i^{(E)}(n)$. Moreover, by Lemma 4.2.1, the definition of $E$, and the fact that only substage 2 can be performed between stage $m$ and the stage at which $i$ is finally cancelled, we can conclude that $A$ will be defined so that $\varphi_i^{(E)}(n) = \varphi_i^{(A-\{n\})}(n)$, hence $C_A(n) \neq \varphi_i^{(A-\{n\})}(n)$. Thus, $i$ can never be cancelled.

Hence, there is some stage in the construction such that all cancellations of indices smaller than $i$ that will ever occur have already occurred by that stage. By the observation above about cancellations, it follows that $i$ satisfies condition (a) so that at all subsequent stages numbered $\langle x, \Phi_{a^*}(x)\rangle$ for some $x$, condition (b4) must fail to be satisfied for index $i$. But then Lemma 4.2.1 implies that $\Phi_i^{(A-\{x\})}(x) > t(x)$ a.e.

Comp $A \leqslant g \circ t$ *a.e.* We see that $(\forall a, x)[\varphi_a(x) \downarrow \Rightarrow \varphi_{\alpha(a)}(x) \downarrow]$.

Thus, by the combining lemma, an appropriate function $g$ exists. Q.E.D.

As a result of Theorem 4.7, we now obtain the desired independence result:

THEOREM 4.8. *There exists $h \in R_2$ with the following property: For any total running times $t_B$ and $t_C$, there exist recursive sets B and C with*

$$\text{Comp } B \leqslant h \circ t_B \text{ a.e.}, \quad \text{Comp } C \leqslant h \circ t_C \text{ a.e.},$$

$$\text{Comp}^{(C)} B > t_B \text{ a.e.}, \quad \text{Comp}^{(B)} C > t_C \text{ a.e.}$$

PROOF. We prove the theorem for space measure on oracle Turing machines; recursive relatedness will then give the general result.

If $t_B$ and $t_C$ are space functions, then $t_B$ join $t_C$ may also easily be shown to be a space function. We may apply Theorem 4.7 to $t_B$ join $t_C$ and obtain a recursive set $A$ with

$$\text{Comp } A \leqslant g \circ (t_B \text{ join } t_C) \text{ a.e.}, \quad \text{and} \quad \text{Comp}^{[A]} A > t_B \text{ join } t_C \text{ a.e.}$$

We choose $B$, $C$ so that $A = B$ join $C$, and define $h(x, y) = g(2x, y) + g(2x + 1, y)$. We claim that $B$, $C$ and $h$ have the required properties, and leave the verification to the reader. Q.E.D.

The earliest independence result in the literature which can be stated in terms of existence of two recursive sets not helping each other is due to Axt [A]. Axt states the existence of recursive sets $A$ and $B$ such that neither is primitive recursive in the other. His proof does not use the complexity formulation of "primitive recursive in"; it is an initial segment diagonal construction similar to Theorem IV in [Ro1, Chapter 13]. With the complexity formulation (Proposition 3.2) we immediately obtain Axt's result as a corollary of Theorem 4.8.

By methods similar to those above, and using the concept of a "complexity sequence" [MeF1], we may obtain the following related result announced in [MeF2]. This result is also claimed by M. K. Vasilyev [T2]. We omit the proof.

PROPOSITION 4.8.1. $(\forall t \in R_1)(\exists$ *recursive A, B) such that*

$$\text{Comp } A > t \text{ a.e.}, \quad \text{Comp } B > t \text{ a.e.},$$

$$\neg(A \text{ s-improves } B \text{ i.o.}), \text{ and } \neg(B \text{ s-improves } A \text{ i.o.})$$

*where* $s = \lambda x, y[y]$.

We now use Theorem 4.8 to obtain counterexamples to transitivity and symmetry of helping.

COROLLARY 4.8.2. *For any measure $\Phi^{(\ )}$ there exists $g \in R_2$ such that*

*for any function $k \geqslant g$ the relations "k-improvement a.e." and "k-improvement i.o." are neither transitive nor symmetric on the recursive sets.*

PROOF. We outline the construction of three sets which provide counter-examples to all four properties.

We choose running times $t_B$ and $t_C$ with $t_B$ much larger than $t_C$ and with $t_C$ much larger than $k$. The precise definition of "much larger" is implicit in the argument below.

By Theorem 4.8, we may obtain $B$, $C$ and $h$. We then consider the three sets $B$, $B$ join $(B \oplus C)$, and $C$. We note the following relationships between the sets, providing $k \in R_2$ is sufficiently large:

(1) $\neg$($B$ $k$-improves $C$ i.o.).

(2) $\neg$($C$ $k$-improves $B$ i.o.).

(3) $B$ $k$-improves $B$ join $(B \oplus C)$ a.e.

(4) $B$ join $(B \oplus C)$ $k$-improves $B$ a.e.

(5) $B$ join $(B \oplus C)$ $k$-improves $C$ a.e.

(6) $\neg$($C$ $k$-improves $B$ join $(B \oplus C)$ i.o.).

(1) and (2) are clear by Theorem 4.8, if $k > h$.

(3) is true because a $B$-oracle machine reduces the complexity of $B$ join $(B \oplus C)$ on even arguments to triviality, and on odd arguments to the complexity of $C$. Since $t_B$ is much larger than $t_C$ and the complexity of $B$ join $(B \oplus C)$ is compressed around $t_B(\lfloor x/2 \rfloor)$, this is a large reduction in the complexity of $B$ join $(B \oplus C)$ a.e.

(4) is clear since $t_B$ is much larger than $k$, and an oracle for $B$ join $(B \oplus C)$ reduces the complexity of $B$ to triviality.

(5) is true since $t_C$ is much larger than $k$, and an oracle for $B$ join $(B \oplus C)$ reduces the complexity of $C$ to triviality because $C_C(x)$ equals the mod two sum of $C_{B\text{join}(B \oplus C)}(2x)$ and $C_{B\text{join}(B \oplus C)}(2x + 1)$.

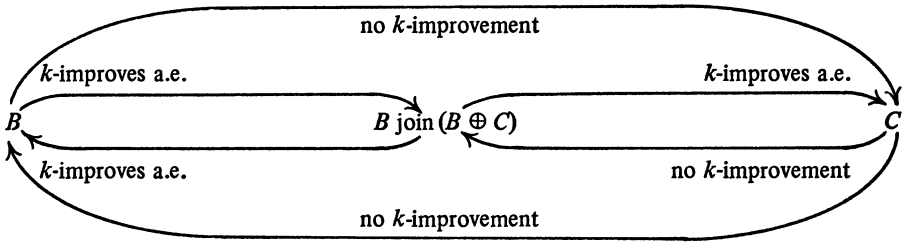
(6) If $C$ $k$-improves $B$ join $(B \oplus C)$ i.o., then either $C$ $k$-improves $B$ join $(B \oplus C)$ on infinitely many even arguments or infinitely many odd arguments. Theorem 4.8 rules out the first possibility (if $k$ is sufficiently large), so improvement must occur on infinitely many odd arguments.

It then follows that $C$ $k'$-improves $(B \oplus C)$ i.o. for some $k'$ which is only slightly smaller than $k$.

But from an i.o. "fast" program for $B \oplus C$ using oracle $C$, it is easy to see how to construct an i.o. "fast" program for $B$ using oracle $C$, so again $C$ $k''$-improves $B$ i.o. for some $k''$ only slightly smaller than $k$, contradicting Theorem 4.8.

These arguments may conveniently be proved formally for the space measure and shown to hold for all measures by recursive relatedness. We omit the details.

The following diagram summarizes our results and exhibits the needed counterexamples:




5. **Sets that do not help.** In this and the next section, we ask if it is possible to improve Theorem 4.8, which proves the existence of two independent recursive sets by diagonalization. Perhaps independent sets are pathological; we would like to obtain a stronger result which fixes one of the two sets arbitrarily. Therefore, we ask the following (informal) question: Which is true?

(1) For any recursive set $A$, there exist arbitrarily complex recursive sets $B$ that do not help the computation of $A$.

(2) There is a recursive set $A$ whose computation is helped by all sufficiently complex recursive sets $B$ (a "universally-helped set").

REMARK 5.1. We first note that any recursive set $B$ will be universally-helped in a contrived measure: fix $g \in R_2$, increasing in both variables, such that with respect to the space measure Comp $B \leqslant \lambda x[g(x, 0]$ a.e. Define a measure $\Phi^{(\ )}$ as follows:

$$\Phi_i^{(A)}(x) = \begin{cases} S_i^{(A)}(x) & \text{if } (\exists y \leqslant x)[y \in A], \\ 1 + g \circ g \circ S_i^{(A)}(x) & \text{otherwise.} \end{cases}$$

It is easy to show that in measure $\Phi^{(\ )}$, $B$ is $g$-improved a.e. by any set $A \neq \emptyset$.

In fact, any set $B$ such that for some $t \in R_1$ Comp $B \geqslant t$ a.e. and Comp $B \leqslant g \circ t$ a.e. with respect to the space measure is also $g$-improved a.e. in measure $\Phi^{(\ )}$ by any set $A \neq \emptyset$.

This observation indicates that a certain reduction in complexity less than some recursive function $g$ depending only on the measure should not properly be interpreted as helping. Thus, if a reduction in the complexity of a set $B$ using an oracle for $A$, when compared to the complexity of $B$ without an oracle, is less than $g$, then the intended interpretation is that $B$ does *not* help $A$.

DEFINITION 5.2. We say that a property holds "for arbitrarily complex recursive sets" if

$(\forall r \in R_1)(\exists B$ recursive)[Comp $B > r$ a.e. and $B$ has the desired property].

We now present two theorems which support the first of the above conjectures about helping. Both theorems begin with an arbitrary recursive set $A$ and a given lower bound $t_A$ on the complexity of $A$, and then establish the existence of arbitrarily complex sets $B$ that "preserve" $t_A$ as a lower bound even in the presence of $B$-oracles. The first theorem covers the case when $t_A$ is an i.o. lower bound.

The method of proof is similar to that used by Machtey and this result was also observed by Machtey [Ma2, pp.621–622]. (In [Ma3] a similarly motivated but technically distinct notion of (not) helping by preserving i.o. lower bounds is considered.)

THEOREM 5.3. *There exists $g \in R_2$ with the following property: For any $t_A \in R_1$, and any recursive set $A$ with* Comp $A > g \circ t_A$ *i.o., there exist arbitrarily complex recursive sets $B$ with* $\text{Comp}^{(B)} A > t_A$ *i.o.*

PROOF. We obtain $g$ from the following

LEMMA 5.3.1. *There exists $g \in R_2$ with the following property: If $B$ is a finite set, $r \in R_1$, and $\varphi_i^{(B)} = r$ a.e., then* Comp $r \leqslant g \circ \Phi_i^{(B)}$ *a.e.*

PROOF OF LEMMA 5.3.1. Follows from a convergence argument, using the combining lemma.

PROOF OF THEOREM 5.3, CONTINUED. We choose $t_B \in R_1$ arbitrarily. $t_B$ will be a lower bound on $B$'s complexity.

We will define $B$ in stages, with $C_B(x)$ being defined at stage $x$. During the construction, we cancel indices of programs we know to differ from $C_B$. In the course of the construction, integers $a$, $b$ and $c$ will be defined and changed from stage to stage, where:

$a$ keeps count of how many conditions of a certain type have so far been satisfied;

$b = \pi_1(a)$ indicates which $B$-oracle program is currently being examined; and

$c$ keeps track of a tentative commitment to an extension of the already-defined initial segment of $B$.

We let $B_x = \{y \leqslant x \mid y \in B\}$.

Stage 0 begins with $a = b = 0$, $c$ undefined.

*Stage $x$.* See if there exists $i < a$ such that $i$ is not yet cancelled and $\Phi_i(x) \leqslant t_B(x)$.

1. If so, choose the smallest such $i$.
   Let $C_B(x) = 1 \dot{-} \varphi_i(x)$, and *cancel $i$.*
   Let $c$ become undefined.
   Go on to stage $x + 1$.

2. If no such $i$ exists, define $C_B(x) = 0$.
See if $c$ is defined.
2.1. If so, see if $c \leqslant x$.
2.1.1. If $c \leqslant x$, redefine $a = a + 1, b = \pi_1(a)$, $c$ = undefined, and go on to stage $x + 1$.
2.1.2. If $c > x$, just go on to stage $x + 1$.
2.2. If $c$ is not defined, see if there exists an argument $y$ such that $a \leqslant y \leqslant x$ and *either* $\Phi_b^{(B_x)}(y) > t_A(y)$ *or*

$$(\Phi_b^{(B_x)}(y) \leqslant t_A(y) \text{ and } \varphi_b^{(B_x)}(y) \neq C_A(y)).$$

2.2.1. If so, let $h$ be the function whose existence is asserted in Lemma 4.2.1 and define $c = h(b, y, t_A(y))$.
Go on to stage $x + 1$.
2.2.2. If no such argument $y$ exists, just retain the values of $a$ and $b$ and go to stage $x + 1$.

END OF CONSTRUCTION

VERIFICATION. The key technical remark is that the variable $a$ in the construction must increase without bound. Suppose it does not. Then the reader may verify that if $a_0$ is the maximum value achieved by $a$ and $b_0 = \pi_1(a_0)$, then $B$ is finite and $\varphi_{b_0}^{(B)} = C_A$ a.e. and $\Phi_{b_0}^{(B)} \leqslant t_A$ a.e.

But then (assuming without loss of generality that $g(x, y)$ is increasing in $y$) Lemma 5.3.1 implies Comp $A \leqslant g \circ t_A$ a.e., contradicting the hypothesis of the theorem.

Now, given that $a$ grows unboundedly, part 1 of the procedure ensures that Comp $B > t_B$ a.e. by an argument similar to that of Theorem 4.6.

It remains to show that $\text{Comp}^{(B)}A > t_A$ i.o.:

Assume the contrary: $(\exists i)[(\varphi_i^{(B)} = C_A)$ and $(\Phi_i^{(B)} \leqslant t_A$ a.e.$)]$.

Then there exists some integer $a_0$ such that $\pi_1(a_0) = i$ and $(\forall y \geqslant a_0)$ $[\Phi_i^{(B)}(y) \leqslant t_A(y)]$.

When $a$ is first set equal to $a_0$ at some stage, $c$ is undefined by 2.1.1. Since $a$ grows without bound, it follows that eventually at some later stage $x$. part 2.1.1 must change $a$ from $a_0$ to $a_0 + 1$.

But this implies that $c$ must have been last defined at part 2.2.1 of some interim stage $z < x$, so there is a $y$, $a_0 \leqslant y \leqslant z$, such that

$$\Phi_i^{(B_z)}(y) > t_A(y) \quad \text{or} \quad [\Phi_i^{(B_z)}(y) \leqslant t_A(y) \text{ and } \varphi_i^{(B_z)}(y) \neq C_A(y)].$$

But then part 2 defines $C_B = 0$ from stages $z$ through $x$, so $B_x = B_z$, and the definition of $c$ at stage $z$ ensures that

$$\Phi_i^{(B)}(y) > t_A(y) \quad \text{or} \quad \varphi_i^{(B)}(y) \neq C_A(y)$$

which contradicts the definition of $a_0$. Q.E.D.

We note that, for the space measure $S^{(\ )}$, $g = \lambda x, y[y]$ will suffice to satisfy Lemma 5.3.1 and hence Theorem 5.3.

We remark that if in Theorem 5.3 we assume that $t_B$ is chosen to be a running time, and that $t_B$ is increasing and much larger than both the complexity of $t_A$ and the complexity of $A$, we could prove that the set $B$ has its complexity "compressed" around the function $t_B$.

COROLLARY 5.3.2 (implicit in [Ma2]). *For any nonprimitive recursive but recursive set $A$, there exist arbitrarily complex recursive sets $B$ such that $A \not\leqslant_p B$ and $B \not\leqslant_p A$.*

PROOF. By Corollary 3.4.2 with $(\{B_i\} = \{\varnothing\})$, there is an $f \in R_1$ such that $A$ is primitive recursive iff Comp $A \leqslant f$ a.e. in the space measure. The compression Theorem 2.8 applied to the space measure implies that for every primitive recursive function $t$, there is a primitive recursive set $P$ such that Comp $P \geqslant t$ a.e.; this implies that $f \geqslant t$ a.e. for every primitive recursive function $t$.

Let $A$ be a recursive set which is not primitive recursive; so Comp $A > f$ i.o. Use Theorem 5.3 for the space measure to obtain an arbitrarily complex recursive set $B$ such that $\text{Comp}^{(B)}A > f$ i.o. Hence $\text{Comp}^{(B)}A > t$ i.o. for any primitive recursive $t$, and so $A \not\leqslant_p B$ by Proposition 3.2.

Theorem 4.2 implies that $\text{Comp}^{(A)}B > f$ a.e. for all sufficiently complex recursive sets $B$, so $B \not\leqslant_p A$ also. Q.E.D.

The next theorem is similar to Theorem 5.3, but the lower bound $t_A$ is now assumed to be an a.e. lower bound instead of an i.o. lower bound; we also require the additional assumption that $t_A$ is a running time.

THEOREM 5.4. *There exists $g \in R_2$ with the following property: For any total running time $t_A$ and any recursive set $A$ with* Comp $A > g \circ t_A$ *a.e., there exist arbitrarily complex recursive sets $B$ with* $\text{Comp}^{(B)}A > t_A$ *a.e.*

PROOF. We assume for simplicity that $t_A(x) \geqslant \lambda x[x]$. There is no loss of generality in this assumption as the reader may verify by slightly modifying the function $g$ in the statement of the theorem.

We choose a function $t_B$ to be an a.e. lower bound on $B$'s complexity. By Lemma 2.8.2 we may assume without loss of generality that $t_B$ is an increasing running time.

We describe a construction which will give us the required set $B$, working from $t_A$, $t_B$ and $A$. We use the *s-m-n* theorem to obtain a function $\beta \in R_3$. The parameters $a$, $b$, and $c$ in the construction are to be thought of as follows:

$$\Phi_a \text{ will be } t_A, \quad \Phi_b \text{ will be } t_B, \quad \varphi_c \text{ will be } C_A.$$

For this choice of $a$, $b$, $c$, the function $\varphi_{\beta(a,b,c)}$ will be the desired function $C_B$.

DEFINITION OF $\varphi_{\beta(a,b,c)}$. $\varphi_{\beta(a,b,c)}$ will be defined in stages, with $\varphi_{\beta(a,b,c)}(n)$ being defined at stage $n$.

Stages are executed in numerical order beginning at stage 0. As in earlier constructions, we maintain the tacit convention that results in the computation of $\varphi_{\beta(a,b,c)}(m)$ for $m < n$ may be used effectively at stage $n$, and that $\varphi_{\beta(a,b,c)}(n)$ diverges if $\varphi_{\beta(a,b,c)}(m)$ diverges for any $m < n$.

During the construction, two types of cancellation occur, which we call 1-cancellation and 2-cancellation. An index $i$ is 1-cancelled when $\varphi_{\beta(a,b,c)}$ has been defined in such a way that

$$(\exists x, y)(\forall C)[(C_C \restriction \{0, \ldots, y\} = \varphi_{\beta(a,b,c)} \restriction \{0, \ldots, y\}) \Rightarrow (\varphi_i^{(C)}(x) \neq C_A(x))]\,.(^9)$$

These 1-cancellations will help to ensure $\mathrm{Comp}^{(B)}A > t_A$ a.e.

An index $i$ is 2-cancelled when it has been ensured that $\varphi_i \neq \varphi_{\beta(a,b,c)}$. Indices $i$ are 2-cancelled when $\Phi_i$ is less than $t_B$ sufficiently many times. This will ensure $\mathrm{Comp}\, B > t_B$ a.e.

Once an index is 1-cancelled or 2-cancelled, it remains so at all later stages.

Also, at any particular time during the construction, there may be some "tentatively 1-cancelled" indices. If an index $i$ is tentatively 1-cancelled, a pair of integers $(x_i, y_i)$ will be defined such that if it is ever discovered that $C_A(x_i) \neq y_i$, then $i$ will become 1-cancelled. If it is ever discovered that $C_A(x_i) = y_i$, then the tentative 1-cancellation will be removed.

The same index may become tentatively 1-cancelled and lose its tentative 1-cancellation repeatedly, the values of $(x_i, y_i)$ changing with each tentative 1-cancellation, but we will show that (in the cases of interest) any index can only become tentatively 1-cancelled finitely often.

Finally, at any time during the construction there may be a "tentative commitment for (an index) $i$". A tentative commitment for $i$ is a quadruple $(i, x_i, y_i, z_i)$, where $z_i$ is the canonical index of a 0-1 valued function $F_{z_i}$ with finite domain such that

$$(\forall C)[(C_C \restriction \text{domain } F_{z_i} = F_{z_i}) \Rightarrow (\varphi_i^{(C)}(x_i) = y_i)]\,,$$

and $F_{z_i}$ is an extension of the finite portion of $\varphi_{\beta(a,b,c)}$ defined at the time the tentative commitment to $i$ is made. The tentative commitment is designed to allow subsequent tentative 1-cancellation of $i$, if possible.

The tentative commitment for $i$ will eventually be fulfilled, at which time $i$ becomes tentatively 1-cancelled, unless it is interrupted by the 2-cancellation of an index smaller than (i.e., of higher priority than) $i$, or by a new tentative commitment for an index smaller than $i$.

---

(9) $f \restriction B$ denotes the function $f$ restricted to domain $B$.

In both the following constructions, we will speak of the "first" member of a certain collection of finite sets, it is to be understood that the lexicographically first set in the collection is to be chosen.

At the beginning of stage 0, there are no 1-cancellations, tentative 1-cancellations, 2-cancellations, or tentative commitments.

*Stage n.* (Define $\varphi_{\beta(a,b,c)}(n)$.)

1. (Make tentative commitments.) Compute $\Phi_b(n)$ and $\Phi_b(n \dot{-} 1)$. (If either diverges, then $\varphi_{\beta(a,b,c)}$ will diverge.)
   Let $X = \{x \leqslant \Phi_b(n) \mid \Phi_b(n \dot{-} 1) < \Phi_a(x) \leqslant \Phi_b(n)\}$.
   See if either of the following, (a) or (b), holds:
   (a) there exist $i$, $x$, $E$ such that:
      (a1) $i \leqslant n$, $i$ is neither 1-cancelled nor tentatively 1-cancelled, and if there is a tentative commitment for some $j$, then $i < j$,
      (a2) $x \in X$,
      (a3) $E \subset \{y \mid y \leqslant \max(n, h(i, x, \Phi_a(x)))\}$ (where $h$ is the function whose existence asserted in Lemma 4.2.1), and $(\forall y \leqslant n \dot{-} 1)[y \in E \Leftrightarrow \varphi_{\beta(a,b,c)}(y) = 1]$, and
      (a4) $\Phi_i^{(E)}(x) \leqslant \Phi_a(x)$.
   (b) There exists $i \leqslant n$, where $i$ is not 2-cancelled, and if there is a current tentative commitment for some $j$, then $i < j$, and $\Phi_i(n) \leqslant \Phi_b(n)$.

   1.1. If neither (a) nor (b) holds,
      1.1.1. If there is no current tentative commitment, define $\varphi_{\beta(a,b,c)}(n) = 0$ and go on to substage 2.
      1.1.2. If there *is* a tentative commitment $(j, x_j, y_j, z_j)$, let $\varphi_{\beta(a,b,c)}(n) = F_{z_j}(n)$. Go on to substage 2.

   1.2. If either (a) or (b) does hold, fix $i$ to be the smallest index for which either (a) or (b) is true.
      1.2.1. If $i$ arises from (a), choose the $x$ such that $\Phi_a(x)$ is smallest (if two are equal, choose the smaller $x$), and for this $x$ choose the first set $E$ such that $(i, x, E)$ satisfy (a). Remove any current tentative commitment, and make a new *tentative commitment for i*, $(i, x, \varphi_i^{(E)}(x), z_i)$, where $z_i$ is the canonical index of the function

$$F_{z_i} = C_E \upharpoonright \{y \mid y \leqslant \max(n, h(i, x, \Phi_a(x)))\}$$

($h$ as in Lemma 4.2.1).
Define $\varphi_{\beta(a,b,c)}(n) = F_{z_i}(n)$, and go on to substage 2.

1.2.2. If $i$ arises from (b) but not from (a), define $\varphi_{\beta(a,b,c)}(n) = 1 \dot{-} \varphi_i(n)$, and 2-*cancel* $i$. Remove any current tentative commitment and go on to substage 2.

2. (Convert tentative commitments to tentative 1-cancellations.) See if there is a current tentative commitment $(i, x_i, y_i, z_i)$ such that $n \geqslant \max(\text{domain } F_{z_i})$.

2.1. If so, *tentatively* 1-*cancel* $i$, associating $(x_i, y_i)$ with the tentative 1-cancellation. Remove the tentative commitment and go on to substage 3.

2.2. If not, then just go on to substage 3.

3. (Convert tentative 1-cancellations to 1-cancellations.) For each tentatively 1-cancelled index $i$ with an associated pair of integers $(x_i, y_i)$, see if $\Phi_c(x_i) \leqslant \Phi_b(n)$.

3.1. If not, go to stage $n + 1$.

3.2. If so, then

3.2.1. If $\varphi_c(x_i) = y_i$, remove $i$'s tentative 1-cancellation.

3.2.2. If $\varphi_c(x_i) \neq y_i$, remove $i$'s tentative 1-cancellation and 1-*cancel* $i$.

Go to stage $n + 1$.

END OF CONSTRUCTION

It is easy to verify that if $\Phi_b$ is total, then for any indices $a,c$, it is the case that $\varphi_{\beta(a,b,c)} \in R_1$ and $\varphi_{\beta(a,b,c)}$ is 0-1 valued.

Now choose $a^*$, $b^*$ and $c^*$ such that $\Phi_{a^*} = t_A$, $\Phi_{b^*} = t_B$ and $\varphi_{c^*} = C_A$. Let $C_B \overset{\text{def}}{=} \varphi_{\beta(a^*,b^*,c^*)}$.

We claim that this set $B$ has the required properties. The key fact in the proof is the claim that no index is tentatively 1-cancelled i.o. If we assume this for the moment, the rest of the proof can be completed as follows:

It is easy to see that $\text{Comp } B > t_B$ a.e., as in earlier proofs: if $\Phi_i \leqslant t_B$ i.o., then $i$ will be 2-cancelled once all the finitely many higher priority indices which are ever going to be 2-cancelled are so cancelled, and once all the (finitely many) tentative 1-cancellations of higher priority indices have been made. When $i$ is 2-cancelled, clause 1.2.2 guarantees that $\varphi_i \neq C_B$.

We also claim that $\text{Comp}^{(B)} A > t_A$ a.e.

For if not, then there is an index $i$ such that $\varphi_i^{(B)} = C_A$ and $\Phi_i^{(B)} \leqslant t_A$ i.o.

Such an $i$ could never be 1-cancelled during the construction of $B$, for this would mean that for some finite set $E$ and some argument $x$,

$$C_A(x) \neq \varphi_i^{(E)}(x) \quad \text{by the 1-cancellation,}$$

but $\varphi_i^{(E)}(x) = \varphi_i^{(B)}(x)$ according to Lemma 4.2.1. and the tentative 1-cancellation which must have preceded the 1-cancellation.

Therefore, each tentative 1-cancellation of $i$ will eventually be removed by clause 3.2.1.

We will eventually reach some stage $e$ in the construction of $B$ such that after stage $e$, no $j < i$ becomes tentatively 1-cancelled or 2-cancelled. Beyond stage $e$, clauses (a), (b) and 1.2 ensure that no index smaller than $i$ can prevent a tentative commitment for $i$ from being made, nor can an index smaller than $i$ interrupt such a tentative commitment for $i$.

Thus, whenever $i$ satisfies clause (a) at some stage $n > e$, and $i$ is not already tentatively 1-cancelled at stage $n$, $i$ will become tentatively 1-cancelled at stage $n$.

But by Lemma 4.2.1, $\Phi_i^{(B)}(x) \leqslant t_A(x)$ implies the existence of a set $E$ such that $(i, x, E)$ satisfies clause (a). Since $\Phi_i^{(B)}(x) \leqslant t_A(x)$ i.o., $i$ will satisfy clause (a) i.o., and so must become tentatively 1-cancelled i.o.

But we have assumed that no index $i$ is tentatively 1-cancelled i.o.

Thus, $\text{Comp}^{(B)}A > t$ a.e.

It remains only to verify the fact that no index can become tentatively 1-cancelled infinitely often. In order to do this, we construct (by the *s-m-n* theorem) a function $\gamma \in R_5$ such that if $a, b, c, d, e$ are chosen so that:

$$\Phi_a = t_A, \quad \Phi_b = t_B, \quad \varphi_c = C_A,$$

$d$ = the least index which becomes tentiatively 1-cancelled infinitely many times,

$e > d$, and

$e$ = the number of a stage beyond which no index smaller than $d$ ever becomes tentatively 1-cancelled or 2-cancelled,

then $\varphi_{\gamma(a,b,c,d,e)}$ will represent a program for $C_A$ requiring measure $\leqslant g \circ t_A$ i.o. (for an appropriate function $g$). We will let this be the $g$ in the hypothesis of the theorem, so that we obtain here a contradiction to "Comp $A > g \circ t_A$ a.e."

DEFINITION OF $\varphi_{\gamma(a,b,c,d,e)}$. To compute $\varphi_{\gamma(a,b,c,d,e)}(x)$, proceed as follows: If $\Phi_a(x) \uparrow$ or $(\forall n)[\Phi_b(x) > \Phi_b(n)]$, then $\varphi_{\gamma(a,b,c,d,e)}(x) \uparrow$.

Otherwise, let $n = \mu m[\Phi_a(x) \leqslant \Phi_b(m)]$.

1. If $n \leqslant e$, let $\varphi_{\gamma(a,b,c,d,e)}(x) = \varphi_c(x)$.
2. If $n > e$, then perform stages 0 through $n - 1$ in the construction of $\varphi_{\beta(a,b,c)}$. (If the computation of any of these stages diverges, then $\varphi_{\gamma(a,b,c,d,e)}(x)$ diverges.) At the point immediately after completing stage $n - 1$, see if *either* there is a tentative commitment $(d, x_d, y_d, z_d)$ *or* $d$ is tentatively 1-cancelled.

   2.1. If either condition is true, let $\varphi_{\gamma(a,b,c,d,e)}(x) = \varphi_c(x)$.

   2.2. Otherwise, see if some tentative commitment $(d, x', y, z)$, for

$x \neq x'$ would be made at clause 1.2.1 of stage $n$ in the construction of $B$. That is, see if

$$(\exists x' \leqslant \Phi_a(x))$$
$$[((\Phi_b(n-1) < \Phi_a(x') < \Phi_a(x)) \text{ or } (\Phi_a(x) = \Phi_a(x') \text{ and } x' < x))$$
$$\text{and } (\exists E \subset \{w \mid w \leqslant \max(n, h(d, x', \Phi_a(x')))\})$$
$$[(\forall w \leqslant n-1)[w \in E \Leftrightarrow \varphi_{\beta(a,b,c)}(w) = 1] \text{ and } \Phi_d^{(E)}(x') \leqslant \Phi_a(x')]].$$

(Note that since a tentative commitment to $x'$ instead of $x$ would be made only if $\Phi_a(x') \leqslant \Phi_a(x)$, and since it is assumed that $\Phi_a = t_A \geqslant \lambda y[y]$ and so $x' \leqslant \Phi_a(x')$, it follows that the existential quantifier may be limited to $x' \leqslant \Phi_a(x)$.)

2.2.1. If so, let $\varphi_{\gamma(a,b,c,d,e)}(x) = \varphi_c(x)$.

2.2.2. If not, then see if some tentative commitment $(d, x, y, z)$ would be made at stage $n$ in the construction of $B$. That is, see if

$$(\exists E \subset \{w \mid w \leqslant \max(n, h(d, x, \Phi_a(x)))\})$$
$$[(\forall w \leqslant n-1)[w \in E \Leftrightarrow \varphi_{\beta(a,b,c)}(w) = 1] \text{ and } \Phi_d^{(E)}(x) \leqslant \Phi_a(x)].$$

2.2.2.1. If not, let $\varphi_{\gamma(a,b,c,d,e)}(x) = \varphi_c(x)$.

2.2.2.2. If so, consider the first such $E$ and let: $\varphi_{\gamma(a,b,c,d,e)}(x) = \varphi_d^{(E)}(x)$.

END OF CONSTRUCTION

We now assume, as indicated before the construction of $\varphi_{\gamma(a,b,c,d,e)}$, that $a^*$, $b^*$, $c^*$, $d^*$ and $e^*$ are fixed as follows:

$$\Phi_{a^*} = t_A, \quad \Phi_{b^*} = t_B, \quad \varphi_{c^*} = C_A,$$

$d^* =$ the least index which becomes tentatively 1-cancelled infinitely many times during the construction of $\varphi_{\beta(a^*,b^*,c^*)}$,

$e^* > d^*$, and

$e^* =$ the number of a stage in the construction of $\varphi_{\beta(a^*,b^*,c^*)}$ after which no index smaller than $d^*$ ever becomes tentatively 1-cancelled or 2-cancelled.

We claim $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)} = C_A$.

For all clauses except 2.2.2.2, $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)} = \varphi_{c^*} = C_A$. We must check what happens if clause 2.2.2.2 defines $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)}(x)$ to be $\varphi_{d^*}^{(E)}(x)$.

If 2.2.2.2 is executed in defining some $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)}(x)$, then there is a stage $n > e^*$ in the construction of $\varphi_{\beta(a^*,b^*,c^*)}$ at which $d^*$, $x$ and some set $E$ satisfy the conditions in 1(a) of that construction. Now $d^*$ must be the smallest index for which either (a) or (b) is satisfied at stage $n$ because we are already past stage $e^*$.

Thus, in stage $n$ of the construction of $\varphi_{\beta(a^*,b^*,c^*)}$, clause 1.2.1 must be executed for $i = d^*$.

But since clause 2.2.1 of the construction of $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)}(x)$ was not executed, it must be the case that no other argument $x'$ could interfere with a tentative commitment $(d^*, x, y_{d^*}, z_{d^*})$ being made at stage $n$ in the definition of $\varphi_{\beta(a^*,b^*,c^*)}$, and so some tentative commitment $(d^*, x, \varphi_{d^*}^{(E)}(x), z_{d^*})$ will be made.

Eventually, this tentative commitment for $d^*$ will cause $d^*$ to become tentatively 1-cancelled, since $n > e^*$. When $d^*$ becomes tentatively 1-cancelled, it will be associated with the pair of integers $(x, \varphi_{d^*}^{(E)}(x))$.

Since $d^*$ becomes tentatively 1-cancelled infinitely often during the construction of $\varphi_{\beta(a^*,b^*,c^*)}$, this tentative cancellation must eventually be removed. This can only happen because of clause 3.2.1 at some stage $m > n$ in the construction of $\varphi_{\beta(a^*,b^*,c^*)}$. But 3.2.1 is executed only if $C_A(x) = \varphi_{d^*}^{(E)}(x)$, so $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)}(x) = C_A(x)$ even if $\varphi_\gamma$ is defined by clause 2.2.2.2.

This establishes the claim that $C_A = \varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)}$.

Finally, we would like to show that

$$\Phi_{\gamma(a^*,b^*,c^*,d^*,e^*)} \leqslant g \circ \Phi_{a^*} \ (= g \circ t_A) \text{ i.o.}$$

To do this, we must first define $g$.

Let $g(x, y) = \max_{a,b,c,d,e \leqslant x} g'(x, y, a, b, c, d, e)$, where $g'$ is given below. The idea behind the definition of $g'$ is the following: we list enough conditions, each recursive assuming the preceding ones are satisfied, to ensure that $\varphi_{\gamma(a,b,c,d,e)}(x)$, and hence $\Phi_{\gamma(a,b,c,d,e)}(x)$, converges. On the other hand, we keep the conditions weak enough so as to be satisfied in the cases of interest (i.e., we only list properties actually satisfied by $a^*$, $b^*$, $c^*$, $d^*$ and $e^*$).

Here, we basically follow the construction of $\varphi_{\gamma(a,b,c,d,e)}(x)$ and select which of the conditions on $a^*$, $b^*$, $c^*$, $d^*$ and $e^*$ were needed for the convergence of $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)}(x)$.

We define $g'(x, y, a, b, c, d, e) = \Phi_{\gamma(a,b,c,d,e)}(x)$ provided all the following conditions are satisfied:

1. $e > d$;
2. $y = \Phi_a(x)$;
3. $\Phi_b(y) \geqslant \Phi_a(x)$;
4. Let $n = \mu m[\Phi_a(x) \leqslant \Phi_b(m)]$. Then:

4.1. $n > e$,

4.2. Immediately after performing stage $n-1$ in the construction of $\varphi_{\beta(a,b,c)}$, there is no tentative commitment $(d, x_d, y_d, z_d)$, and $d$ is not tentatively 1-cancelled.

(*Note*. This is an effective test since we know for any $m < n$ that $\Phi_b(m) < \Phi_a(x)$, so that $\varphi_b(m)\downarrow$. This suffices to insure that $\varphi_{\beta(a,b,c)}(m)\downarrow$.)

4.3. The condition stated under part 2.2 of the definition of $\varphi_{\gamma(a,b,c,d,e)}(x)$ is *false*.

4.4. The condition stated under part 2.2.2 of the definition of $\varphi_{\gamma(a,b,c,d,e)}(x)$ is *true*.

If one of the conditions fails to be satisfied, we define $g'(x, y, a, b, c, d, e) = 0$.

Now these conditions are effective, so $g'$ is partial recursive. Moreover, if all the conditions on $x, y, a, b, c, d$, and $e$ are satisfied, then $\varphi_{\gamma(a,b,c,d,e)}(x)\downarrow$ so that $g'$ is total. Therefore, $g \in R_2$.

Now by definition,

$$g(x, \Phi_{a^*}(x)) \geqslant g'(x, \Phi_{a^*}(x), a^*, b^*, c^*, d^*, e^*) \text{ a.e.}$$
$$= \Phi_{\gamma(a^*,b^*,c^*,d^*,e^*)}(x)$$

for all $x$ such that a tentative commitment $(d^*, x, y_{d^*}, z_{d^*})$ is made at some stage after stage $e^*$ in the construction of $B$.

But since we have assumed that $d^*$ is tentatively 1-cancelled infinitely often, this latter equality must occur for infinitely many $x$.

Thus, we have $\Phi_{\gamma(a^*,b^*,c^*,d^*,e^*)} \leqslant g \circ \Phi_{a^*}$ i.o.

But since $\varphi_{\gamma(a^*,b^*,c^*,d^*,e^*)} = C_A$, this contradicts the hypothesis Comp $A > g \circ t_A$ a.e. Therefore, our assumption that $d^*$ was tentatively 1-cancelled infinitely often was wrong, and so we conclude that no index is tentatively 1-cancelled infinitely often in the construction of $\varphi_{\beta(a^*,b^*,c^*)}$. Q.E.D.

We note that in Theorem 5.4, if $t_B$ is chosen to be a monotone increasing running time, then it is not hard to prove that the set $B$ has its complexity "compressed" above the function $t_B$, that is, Comp $B \leqslant h \circ t_B$ a.e. for some $h \in R_2$ which depends only on the measure.

An interesting technical question is whether Theorem 5.4 can be strengthened by replacing the condition that $t_A$ be a running time by the condition that $t_A$ merely be recursive. Theorem 5.3 (for all sufficiently complex sets $A$) would then follow easily from this strengthened version of Theorem 5.4 (cf. the "complexity core" lemma from [L]).

The results of this section aim to support the first of the two opposite conjectures suggested at the beginning of the section. These conjectures have

not been resolved, however, since the set $B$ in each theorem is constructed to preserve a single lower bound on the complexity of the set $A$.

For sets with sufficient speedup [B1], [B2], it may be the case that there is no single lower bound whose preservation by $B$ ensures that $A$ is not helped by $B$ in the sense of the question below. A complete answer would provide a single set $B$ preserving *all* lower bounds for $A$'s complexity:

OPEN QUESTION. Does there exist $g \in R_2$ with the property that for every recursive set $A$ there exist arbitrarily complex recursive sets $B$ such that

$$(\forall t_A \in R_1)[\text{Comp } A > g \circ t_A \text{ a.e.} \Rightarrow \text{Comp}^{(B)} A > t_A \text{ a.e.}]?$$

Replacing "a.e." by "i.o." in the above question produces an equivalent open question, as does replacing the entire last line by "$\neg$($B$ $g$-improves $A$ i.o.)" [L].

6. **Sets which are helped.** In support of the second conjecture at the beginning of §5, we now outline the construction of a class of sets which are helped by all sets whose complexities are "nicely" compressed.

DEFINITION 6.1.1. For any $h \in R_2$, define a recursive set $A_h$ as follows: Let $x_1, x_2, x_3$ be such that $x = \langle x_1, x_2, x_3 \rangle$. Then define

$$C_{A_h}(x) = \begin{cases} 1 \dot{-} (1 \dot{-} \varphi_{x_1}(x_2)) & \text{if } \Phi_{x_1}(x_2) \leqslant h(x_2, x_3), \\ 0 & \text{otherwise.} \end{cases}$$

THEOREM 6.1 (HELPED SETS). *For any $s \in R_2$, there exists $s' \in R_2$ with the following property: If $h \in R_2$, $t$ is a total running time and $B$ is any set such that*

$$\text{Comp } B > s' \circ t \text{ i.o.} \quad \textit{and} \quad \text{Comp } B \leqslant h \circ t \text{ a.e.},$$

*then*

$$B \textit{ s-improves } A_h \textit{ i.o.}$$

PROOF. We give an outline of the ideas used in the proof. $s'$ is chosen to be only slightly larger than $s$, the difference arising from the overhead required for simple subcomputations. We consider $i$ such that $\varphi_i = C_B$ and $\Phi_i \leqslant h \circ t$ a.e.

Let $y = \langle i, x, t(x) \rangle$, so $C_{A_h}(y) = C_B(x)$ by definition of $A_h$.

If for almost all $x$, $C_{A_h}(y)$ could be computed by a program using measure at most $s \circ t(x)$, then a simple modification of this program would compute $C_B$ using measure at most $s' \circ t$ a.e., contradicting the lower bound on Comp $B$. Thus, for infinitely many $x$, to compute $C_{A_h}(y)$ requires measures $s \circ t(x)$.

On the other hand, using a $B$-oracle, a program which recognizes arguments $y$ of the form above can compute $C_{A_h}(y)$ by interrogating the oracle about $x$. Since $t$ is a running time, arguments $y$ of the form above can actually be recognized using measure approximately $t(x)$, so *with* a $B$-oracle, $C_{A_h}(y)$ can be computed in measure $t(x)$ for almost all $x$.

That is, $B$ $s$-improves $A_h$ i.o. Q.E.D.

As a final comment, we note that if we choose $A_h$ for $A$ in Theorems 5.3 and 5.4, and $h$ is sufficiently large, then we may obtain sets $B$ satisfying these theorems such that Comp $B > s' \circ t_B$ a.e. and Comp $B \leqslant h \circ t_B$ a.e. for arbitrarily large running times $t_B$. Any such set $B$ $s$-improves $A_h$ i.o. by Theorem 6.1, yet simultaneously preserves the lower bound on Comp $A_h$ used in Theorems 5.3 or 5.4.

This illustrates our earlier comment that a set $B$ may preserve any given lower bound on Comp $A_h$, but fail to preserve other larger lower bounds. Indeed, it easily follows from the definition of $A_h$ that $A_h$ has i.o. speedup approximately equal to $h$ (cf. [B2]), so that preserving any given lower bound on Comp $A_h$ is insufficient to prevent $s$-improvement of $A_h$.

ACKNOWLEDGEMENT. We would like to thank Michael Machtey and Larry Stockmeyer for their detailed comments on this paper.

## REFERENCES

[A] P. Axt, *On a subrecursive hierarchy primitive recursive degrees,* Trans. Amer. Math. Soc. 92 (1959), 85–105. MR 23 #A3673.

[B1] Manuel Blum, *A machine-independent theory of the complexity of recursive functions,* J. Assoc. Comput. Mach. 14 (1967), 322–336. MR 38 #4213.

[B2] ———, *On effective procedures for speeding up algorithms,* J. Assoc. Comput. Mach. 18 (1971), 290–305. MR 44 #8063.

[Bo] Alan Borodin, *Computational complexity and the existence of complexity gaps,* J. Assoc. Comput. Mach. 19 (1972), 158–174; corrigendum, 576. MR 47 #9888.

[C] Alan Cobham, *The intrinsic computational difficulty of functions,* Logic, Methodology and Philos. Sci. (Proc. 1964 Internat. Congr., Jerusalem, 1964), North-Holland, Amsterdam, 1965, pp. 24–30. MR 34 #7376.

[Con] Robert Constable, *The operator gap,* J. Assoc. Comput. Mach. 19 (1972), 175–183. MR 47 #3159.

[D] Martin Davis, *Computability and unsolvability,* McGraw-Hill Ser. in Information Processing and Computers, McGraw-Hill, New York, 1958. MR 23 #A1525.

[HH] J. Hartmanis and J. E. Hopcroft, *An overview of the theory of computational complexity,* J. Assoc. Comput. Mach. 18 (1971), 444–475. MR 44 #5226.

[J] Carl G. Jockusch, Jr., *Uniformly introreducible sets,* J. Symbolic Logic 33 (1968), 521–536. MR 38 #5619.

[K] Stephen Cole Kleene, *Introduction to metamathematics,* Van Nostrand, Princeton, N.J., 1952. MR 14, 525.

[L] Nancy A. Lynch, *"Helping": Several formalizations,* J. Symbolic Logic 40 (1975), 555–566.

[LR] L. H. Landweber and E. L. Robertson, *Recursive properties of abstract complexity classes,* J. Assoc. Comput. Mach. 19 (1972), 296–308. MR 46 #35.

[Ma1] Michael Machtey, Private communication.

[Ma2] ———, *Augmented loop languages and classes of computable functions*, J. Comput. System Sci. 6 (1972), 603–624.

[Ma3] ———, *Helping and the meet of pairs of honest subrecursive classes*, Information and Control 28 (1975), p. 76.

[McC] Edward M. McCreight, *Classes of computable functions defined by bounds on computation*, Ph. D. Thesis, Department of Computer Science, Carnegie-Mellon University, July 1969.

[McCMe] E. M. McCreight and A. R. Meyer, *Classes of computable functions defined by bounds on computation*, Sympos. on Theory of Computing, Marina del Rey, May 1969.

[MeF1] Albert R. Meyer and Patrick C. Fischer, *Computational speed-up by effective operators*, J. Symbolic Logic 37 (1972), 55–68. MR 47 #6457.

[MeF2] Albert R. Meyer and Michael J. Fischer, *Relatively complex recursive sets*, J. Symbolic Logic 35 (1970), 607–608. (abstract).

[MeRd] A. R. Meyer and Dennis M. Ritchie, *A classification of the recursive functions*, Z. Math. Logik Grundlagen Math. 18 (1972), 71–82. MR 45 #4975.

[MoMe] R. Moll and A. R. Meyer, *Honest bounds for complexity classes of recursive functions*, J. Symbolic Logic 39 (1974), 127–138.

[Par] Charles Parsons, *Hierarchies of primitive recursive functions*, Z. Math. Logik Grundlagen Math. 14 (1968), 357–376. MR 39 #66.

[RD] Dennis M. Ritchie, *Program structure and computational complexity*, Ph. D. Thesis, Division of Engineering and Applied Physics, Harvard University, 1967.

[RRW] R. W. Ritchie, *Classes of predictably computable functions*, Trans. Amer. Math. Soc. 106 (1963), 139–173. MR 28 #2045.

[Ro1] Hartley Rogers, Jr., *Theory of recursive functions and effective computability*, McGraw-Hill, New York, 1967. MR 37 #61.

[Ro2] ———, *Godel numberings of partial recursive functions*, J. Symbolic Logic 23 (1958), 331–341. MR 21 #2585.

[Sa] Gerald E. Sacks, *Degrees of unsolvability*, Ann. of Math. Studies, no. 55, Princeton Univ. Press, Princeton, N.J., 1963. MR 32 #4013.

[Sy] David M. Symes, *The extension of machine-independent computational complexity theory to oracle machine computation and to the computation of finite functions*, Ph. D. Thesis, Department of Applied Analysis and Computer Science, University of Waterloo, Oct. 1971.

[T1] B. A. Trahtenbrot, *On autoreducibility*, Dokl. Akad. Nauk SSSR 192 (1970), 1224–1227 = Soviet Math. Dokl. 11 (1970), 814–817. MR 43 #52.

[T2] ———, Private communication.

DEPARTMENT OF ELECTRICAL ENGINEERING, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MASSACHUSETTS 02139

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF SOUTHERN CALIFORNIA, LOS ANGELES, CALIFORNIA 90007

DEPARTMENT OF COMPUTER SCIENCE, UNIVERSITY OF WASHINGTON, SEATTLE, WASHINGTON 98195