Computational Complexity and the Existence of Complexity Gaps

A. BORODIN

University of Toronto, Toronto, Ontario, Canada*

ABSTRACT. Some consequences of the Blum axioms for step counting functions are investigated. Complexity classes of recursive functions are introduced analogous to the Hartmanis-Stearns classes of recursive sequences. Arbitrarily large "gaps" are shown to occur throughout any complexity hierarchy.

KEY WORDS AND PHRASES: computational complexity, measures of complexity, recursive functions, tape complexity, step counting functions, axiomatic complexity theory

CR CATEGORIES: 5.21, 5.22

1. *Introduction*

During the 1930's, a great deal of research concerned the question of "what functions are computable." Subsequent understanding of the concept of computability gave rise to a new question, namely, "how difficult to compute is a given function." Our purpose is not to analyze the complexity of any specific functions but rather to continue the development of a theory of complexity.

In this paper, we shall study further the properties of complexity classification schemes, in order to develop a framework within which concepts of computational complexity (such as "difficulty" of a function) can be formulated. Our classification scheme evolves directly from the one introduced by Hartmanis and Stearns [11]. In [11, 22], "time" and "memory" are viewed as resources, which any efficient algorithm will try to conserve. Thereafter, a complexity class is formed by placing a bound (a function of the input value) on the amount of a particular resource that an algorithm may use. A function is in a given class if there is an algorithm for computing the function, in which the amount of resource does not exceed the bound.

In [11], every computable monotone-increasing recursive[1] function t induces a class of recursive sequences by limiting the number of steps for computing each

Copyright © 1972, Association for Computing Machinery, Inc.

General permission to republish, but not for profit, all or part of this material is granted provided that reference is made to this publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery.

* Department of Computer Science. This paper embodies the results contained in the author's Ph.D dissertation (Cornell University, 1969). A preliminary version of this paper appeared in the ACM Symposium on Theory of Computing, May 1969. The research has been supported in part by National Science Foundation Grants GJ96, GP6426 and GP7701, and National Research Council Grant A7631.

[1] A number theoretic function is one whose domain and range is contained in the natural numbers N. We will henceforth use the standard terminology of *recursive function* to mean computable total number theoretic function and *partial recursive* to mean computable partial number theoretic function.

digit of the sequence. Specifically, a sequence $\alpha = \alpha_0\alpha_1\alpha_2 \cdots$ is in S_t if and only if there is a multitape T.M. which outputs the nth digit α_n within $t(n)$ computation steps. We may say that the number of computation steps (= time) is a *resource* and we *measure* the complexity of a computation by the amount of resource expended.

Following Hartmanis and Stearns, similar schemes for classifying recursive sets were studied in [22, 8]. Here the classes are derived by placing a bound on the number of tape squares (tape reversals, respectively) in a computation, the bound being a recursive function of the input length. Again, membership in a class is determined by the existence of some "efficient" (relative to the measure) algorithm.

Are any of the results and concepts formulated in the previously mentioned papers valid with respect to other (abstract) models of computation? The concepts are valid. It is the work of Blum [1] which shows how to develop a theory of complexity in a formulation which is machine (model) independent.

Motivated by the work of Rabin [18], Blum chooses two axioms which provide a broad characterization for the intuitive notion of a resource or a measure of complexity. A partial function Φ_i, the *step counting function*,[2] is associated with each (algorithm) ϕ_i. Axiom 1 merely says that a measure is well defined in the sense that we get a value for the measure if and only if the computation converges (i.e. produces a value for the function being computed). Axiom 2 captures the notion of being able to determine if a particular computation exceeds a bound on the measure.

All previously studied measures of complexity (time, tape, reversals) can be viewed as examples of step counting functions. Our goal will be to introduce and investigate classification of recursive functions based on bounding (by a recursive function) the step counting function Φ_i associated with a computation (algorithm) ϕ_i. Although we will be talking about classes of functions, we note that we could have chosen to study recursive sequences or recursive sets.

In the next section, we define our notion of resource bounded complexity classes with respect to axiomatically defined resources (complexity measures). Some immediate observations are then made by generalizing results in Hartmanis and Stearns.

Section 3 concerns itself with hierarchy theorems. In particular, the gap theorem shows how essential are the restrictions in known hierarchy theorems (i.e. the use of tape constructable functions and real time countable functions). More precisely, the gap theorem shows that there is no uniform way to increase a resource bound to guarantee an increase in computing power.

We then begin to view classes of function (the set of permissible bounding functions) as names for complexity classes. In this context, it is established that no naming class can have all the properties one might desire. Specifically, it is shown that there is no class of functions (bounds) $\mathcal{S}$ for which (1) there is a uniform way [composition with some fixed function $h(x) \ll x$] to decrease the computing power and (2) $\mathcal{S}$ names all the classes.

Minimal growth rates are studied in Section 4, as an example of a "gaplike phenomena in the low end of the complexity structure." Section 5, the conclusion, reviews the distinction between "class or structural properties" as opposed to "naming properties." This distinction is made as a possible starting point for attacking the problem of "how to classify measures."

[2] Many other names for Φ_i appear in the literature, including complexity function, Φ constructible function, run-time function, difficulty function.

2. *Axiomatically Defined Complexity Classes*

2.1. BASIC DEFINITIONS. We restate the Blum formulation precisely. An effective list $\langle \phi_0, \phi_1, \phi_2, \cdots \rangle$ of all partial recursive functions is postulated, such that this list satisfies the S_n^m theorem and the universal Turing machine theorem.[3]

The list $\{\phi_i\}$ should be thought of as a list of algorithms or devices for computing the partial recursive functions. A partial function Φ_i is associated with each algorithm ϕ_i. The set of functions $\{\Phi_i\}$ satisfies the following axioms.

Axiom 1: $\Phi_i(x)$ *is defined* $\Leftrightarrow$ $\phi_i(x)$ *is defined.*

Axiom 2: The function

$$M(i, x, m) = \begin{cases} 1 & \text{if } \Phi(x) = m \\ 0 & \text{otherwise} \end{cases}$$

is total recursive.

Henceforth, we will use the standard notation $\phi_i(x)\downarrow$ to mean $\phi_i(x)$ defined at x (x in domain of ϕ_i) and $\phi_i(x)\uparrow$ to mean ϕ_i undefined at x.

As an immediate consequence of Axiom 2, we note that the following predicates are recursive for all k, i, and x:[4]

$$\Phi_i(x) = k, \qquad \Phi_i(x) < k, \qquad \Phi_i(x) > k.$$

Example 2.1: Let $\{Z_i\}$ be the class of Turing machines with a read only input tape, a write only output tape, and one "work" tape. Let ϕ_i be the function computed by Z_i. We fix our input and output convention to be the binary representation of integers and allow an arbitrary but finite number of work symbols. Define $\Phi_i(x) = m$ iff Z_i with input x stops and uses precisely m squares of the work tape for the computation. $\{\Phi_i\}$ has been defined to satisfy Axiom 1. To see that $\{\Phi_i\}$ satisfies Axiom 2, we note that either the number of tape squares used in the computation increases as the computation proceeds or the machine is in a loop. Since we can effectively determine how many machine operations may occur without entering a loop or using a new square on the work tape, Axiom 2 holds for $\{\Phi_i\}$.

Definition 2.2: Let $\langle \phi_0, \phi_1, \phi_2, \cdots \rangle$ and $\langle \Phi_0, \Phi_1, \Phi_2, \cdots \rangle$ be as stated for the Blum axioms. We will call $\Phi = \langle \{\phi_i\}, \{\phi_i\} \rangle$ a *dynamic complexity measure* or *resource* (and for brevity, we will often refer to Φ as a measure). Informally, this notation allows us to denote both the measure and the "devices" upon which the measure is imposed. For notational convenience, the set of step counting functions $\{\hat{\Phi}_i\}$ would be associated with the measure $\hat{\Phi}$. Φ is dynamic in the sense that it describes the behavior of the computations of an algorithm; a computation being the execution of an algorithm when applied to an input. This is in contrast to a static or definitional measure such as the size [2] or "structure" of the ith algorithm.

Rogers [20] shows that acceptable indexings are recursively isomorphic. That is, if $\{\phi_i^*\}$ and $\{\phi_i\}$ are acceptable then there exists a recursive 1-to-1 onto function f

[3] The S_n^m theorem asserts the existence of a total recursive function σ such that $\phi_{\sigma(i,m)}(n) = \phi_i(m, n)$. The universal Turing machine theorem states that for any recursive 1-to-1 onto map $J: N \times N \to N$, there exists a universal machine Z_i with the property that $\phi_i(J(x, y)) = \phi_x(y)$ for all x and y. Indexings, satisfying the above theorems, are called *acceptable* by Rogers [21]. It should be clear that any effective enumeration of Turing machines would yield an acceptable indexing.

[4] $\Phi_i(x) > k$ iff it does *not* hold that ($\Phi_i(x) < k$ or $\Phi_i(x) = k$). Specifically, $(\forall k)\ \Phi_i(x) > k$ when $\Phi_i(x)\uparrow$ [equivalently, $\phi_i(x)\uparrow$].

such that $\phi_i \simeq \phi^*_{f(i)}$ (and $\phi_i^* \simeq \phi_{f^{-1}(i)}$).[5] A complexity measure $\Phi = \langle\{\phi_i\}, \{\Phi_i\}\rangle$ may be viewed as a complexity measure $\Phi^* = \langle\{\phi_i^*\}, \{\Phi_i^*\}\rangle$ with $\Phi_i^* \simeq \Phi_{f^{-1}(i)}$. Φ and Φ^* determine the same set of step counting functions, and in particular, for any function g, $\{\Phi_i \mid \phi_i \simeq g\} = \{\Phi_j^* \mid \phi_j^* \simeq g\}$. Informally, we may say that the existence of a complexity measure with certain properties will not be unique to a particular list $\{\phi_i\}$.

The framework for our investigation will be classes of recursive functions. We generalize the Hartmanis-Stearns concept of classes of recursive sequences S_t as follows:

Definition 2.3: Let Φ be a complexity measure, t a recursive function.

$R_t^\Phi = \{\text{recursive functions } f \mid (\exists i)[\phi_i \simeq f \wedge \Phi_i(x) \leq t(x) \textit{ for almost all } x]\}$.[6]

Informally, f is in R_t^Φ if f is recursive and there exists an algorithm for computing f such that the difficulty (relative to the measure or recource Φ) is bounded almost everywhere by the function t.

We call R_t^Φ a Φ *resource bounded class* or more briefly a *complexity class.*

Regarding Definition 2.3, one may wonder why we did not require $\phi_i(x) \leq t(x)$ for *all* x. The "almost everywhere" criterion is consistent with the concept of a "bounding function" (e.g. Ackerman's function bounds the primitive recursive functions in this "for almost all sense"). It is also consistent with the Blum speedup and compression results, which are discussed later in this paper. Moreover, the distinction between "for almost all x" and $(\forall x)$ disappears for a complexity measure if both

(1) the computation "device" has some sort of "finite control," capable of minimizing a function's complexity at a finite number of argument values;

(2) input-output conventions do not "contribute" to the complexity measure.

The measure of Example 2.1 satisfies these properties. "Time" on a Turing machine would not satisfy the second condition since, the number of steps needed merely to read the input or write the output contributes to the total number of steps (= time). Thus if we used the $(\forall x)$ criteria for "time," no class R_t^Φ could contain all constant functions, which we intuitively feel are easy to compute.

Henceforth, we will abbreviate statements such as $\Phi_i(x) \leq t(x)$ for almost all x by $\Phi_i \leq t$ almost everywhere (a.e.) or $\Phi_i(x) \leq t(x)$ a.e. x. Similarly, $\Phi_i(x) \leq t(x)$ for infinitely many x will appear as $\Phi_i \leq t$ infinitely often (i.o.).

We will sometimes have need to denote the class of algorithms (more precisely the indices) which "behave" within some bound.

Definition 2.4: Let Φ be a complexity measure, t a recursive function.

$$I_t^\Phi = \{i \mid \Phi_1 \leq t \text{ a.e. and } (\forall x)\Phi_i(x)\downarrow\}.$$

It should be apparent that $I_t^\Phi = I_{t'}^\Phi$ implies that $R_t^\Phi = R_{t'}^\Phi$ but not necessarily conversely. When it is clear from the context, we will sometimes omit the superscript Φ, and write I_t and R_t.

2.2. ANALOGS TO HARTMANIS-STEARNS. It is easy to show that many of the properties of the S_t classes proved in [11], generalize to our R_t^Φ classification. The proofs have been omitted in the following straightforward development of results pertaining to the enumerability of R_t^Φ classes.

[5] $\simeq$ represents strong equality of partial functions. That is, $\phi \simeq \Psi$ iff $(\forall x)[\phi(x)\downarrow \Leftrightarrow \Psi(x)\downarrow \wedge \phi(x)\downarrow \Rightarrow \phi(x) = \Psi(x)]$.

[6] "For almost all x" $P(x) \equiv (\exists x_0)(\forall x \geq x_0)P(x)$.

Definition 2.5: A class C of [partial] recursive functions is recursively enumerable (r.e.) $\Leftrightarrow \exists$ [partial] recursive $h(i, m)$: $C = \{\lambda x h(i, x) \mid i \geq 0\}$.[7]

In the context of a listing $\{\phi_i\}$ of partial recursive functions, it is equivalent to say that C is r.e. $\Leftrightarrow \exists$ recursive h': $C = \{\phi_{h'(i)}(x) \mid i \geq 0\}$. We will say that h' enumerates C. We may think of h' as enumerating a list of machines and the set of functions computed by these machines is exactly C. Note that if C is a class of recursive functions, then each $\phi_{h'(i)}$ is a recursive function.

Example 2.6: The class C of functions of finite support is r.e. That is, $C = \{f \mid f$ recursive and $f(x) = 0$ a.e. $x.\}$.

PROPOSITION 2.7. *Let C be an r.e. class of recursive functions. Let Φ be a complexity measure. Then $\exists$ recursive functions b_1 and b_2:*

$$f \text{ in } C \Rightarrow f \leq b_1 \text{ a.e.},$$

$$f \text{ in } C \Rightarrow f \text{ in } R_{b_2}^{\Phi}.$$

THEOREM 2.8. *Let Φ be a complexity measure. Using Proposition 2.7, let b be such that R_b^{Φ} contains all functions of finite support. Then for any recursive function t,*

$$t \geq b \text{ a.e.} \Rightarrow R_t^{\Phi} \text{ is r.e.}$$

In [13, 14], it is shown that there are measures $\hat{\Phi}$ for which some of the "small" classes (i.e. $R_h^{\hat{\Phi}}$, $h(x) \equiv 0$) are not r.e. Following suggestions in [11] and [25], we introduce a property of some measures which allows Theorem 2.8 to follow for any nonempty class R_t^{Φ}.

Definition 2.9: Φ is *finitely invariant* iff

$$[f \text{ in } R_t^{\Phi} \wedge f' \text{ total} \wedge f = f' \text{ a.e.}] \Rightarrow f' \text{ in } R_t^{\Phi}.$$

That is, total functions that have the same value almost everywhere are in the same complexity class. Many familiar measures are finitely invariant (e.g. time, modified so that reading the input is not counted, or tape on a T.M.). However, Definition 2.9 is independent of the Blum axioms.[8]

THEOREM 2.10. *Let Φ be finitely invariant. Then for all recursive t, R_t^{Φ} is r.e.*

2.3. COMBINING ALGORITHMS. In order to achieve the generality desired, we have chosen to study complexity classification with respect to axiomatically defined resources. On the other hand, there is an obvious advantage in studying specific measures such as time and tape; namely, our familiarity with how these measures "behave." The purpose of this section is to show that in many ways, the "behavior" of an arbitrary Blum measure is not "radically" different from the well-studied time or tape.

In particular, Blum shows that "measures (resources) are recursively related."

PROPOSITION 2.11 (Blum). *Let Φ and $\hat{\Phi}$ be measures with respect to the same indexing. Then $\exists$ recursive h: $(\forall t) I_t^{\Phi} \subseteq I_{t'}^{\hat{\Phi}}$, and $I_t^{\hat{\Phi}} \subseteq I_{t'}^{\Phi}$ and thus*

$$R_t^{\Phi} \subseteq R_{t'}^{\hat{\Phi}} \quad \text{and} \quad R_t^{\hat{\Phi}} \subseteq R_{t'}^{\Phi},$$

where $t'(x) = h(x, t(x))$.

[7] λ notation denoting h as a function of x.

[8] In fact, there exists a measure Φ' such that there are arbitrarily large classes $R_t^{\Phi'}$ (i.e. t can be made arbitrarily large) such that $R_t^{\Phi'}$ is not closed under finite changes to functions which are members of $R_t^{\Phi'}$.

PROOF. Let

$$p(i, x, m) = \begin{cases} \Phi_i(x) + \hat{\Phi}_i(x) & \text{if } \Phi_i(x) \leq m \text{ or } \hat{\Phi}_i(x) \leq m \\ 0 & \text{otherwise.} \end{cases}$$

Set

$$h(x, m) = \max_{i \leq x} p(i, x, m)$$

Then, for all $x \geq i$,

$$\Phi_i(x) \leq h(x, \hat{\Phi}_i(x)),$$
$$\hat{\Phi}_i(x) \leq h(x, \Phi_i(x)).$$

The proposition follows by noting that $h(x, m)$ is nondecreasing in both variables. QED

As a direct application of the previous proof technique, we can show that for any measure, if one combines algorithmic devices in well-determined ways to form new devices, then the complexity of the new device will be recursively related to the complexity of the original devices. For example, when the measure is time or tape, we can often "combine computations by running them in parallel" and the resulting complexity does not exceed the maximum of the individual complexities.

PROPOSITION 2.12. *Let Φ be any complexity measure. Let $\phi_{h(i_1, \cdots i_n)}(x)$ be a partial function such that*

$$[\phi_i(x)\downarrow, i = i_1, i_2, \cdots, i_n] \Rightarrow \phi_{h(i_1, \cdots, i_n)}(x)\downarrow.$$

Then there exists a recursive $g(x, y_1, y_2, \cdots, y_n)$ such that

$$[\Phi_{h(i_1, \cdots, i_n)}(x) \leq g(x, \Phi_{i_1}(x), \Phi_{i_2}(x), \cdots, \Phi_{i_n}(x))] \quad a.e.$$

PROOF. Let

$$p(i_1, \cdots, i_n, x, y_1, \cdots, y_n) = \begin{cases} \Phi_{h(i_1, \cdots, i_n)}(x) & \text{if } \Phi_{i_j}(x) = y_j, \quad 1 \leq j \leq n, \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$g(x, y_1, \cdots, y_n) = \max_{\substack{i_j \leq x \\ i \leq j \leq n}} p(i_1, \cdots, i_n, x, y_1, \cdots, y_n)$$

satisfies the proposition.

Example 2.13: Let

$$\phi_{h(i,j)}(x) = \begin{cases} \phi_i(x) & \text{if } \Phi_i(x) \leq \Phi_j(x) \\ \phi_j(x) & \text{if } \Phi_i(x) > \Phi_j(x) \\ \uparrow & \text{if } \phi_i(x) \uparrow \wedge \phi_j(x) \uparrow. \end{cases}$$

If $\Phi_{h(i,j)}(x) = \min(\Phi_i(x), \Phi_j(x))$ holds, then we say that Φ has the "parallel computation" property.

3. *Hierarchy Theorems and the Existence of Complexity Gaps*

3.1. A REVIEW OF SOME KNOWN HIERARCHY THEOREMS. Let us examine some existing complexity hierarchy results.

Example 3.1:

C_L is the class of all recursive sets acceptable in tape $L(n)$ on an off-line T.M. (n = input length)

L is tape constructable iff L is total and there exists an off-line T.M. Z_i such that

$$L(n) = \max_{x:|x|=n} \{\text{tape used by } Z_i \text{ on input } x\},$$

where $|x|$ = input length of x.

Then

$$\inf_{n\to\infty} \frac{L_1(n)}{L_2(n)} \to 0 \qquad \text{and} \qquad L_2 \textit{ tape constructable} \Rightarrow C_{L_2} - C_{L_1} \neq \emptyset.$$

See [22].

Example 3.2:

S_t = class of all t computable binary sequences where t is increasing.

Then

$$\inf \frac{t_1(n) \log t_1(n)}{t_2(n)} \to 0 \qquad \text{and} \qquad t_2 \textit{ real time countable} \Rightarrow S_{t_2} - S_{t_1} \neq \emptyset.$$

See [11] and [12].

The Blum compression theorem is a hierarchy result which pertains to all complexity measures. An essential concept in this theorem is that of a "measured set of functions."

Definition 3.3: A measured set of functions $\mathcal{G} = \{\gamma_i\}$ is an r.e. set of partial recursive functions such that there exists a total recursive function

$$M(i, x, m) = \begin{cases} 1 & \text{if } \gamma_i(x) = m \\ 0 & \text{otherwise.} \end{cases}$$

That is, a measured set of functions satisfies Axiom 2 for a set of step counting functions. Thus the set $\{\Phi_i\}$ is always a measured set. Whether or not a set $\mathcal{G}$ is a measured set does not depend on the choice of formalism $\{\phi_i\}$.

Suppose $\mathcal{S}$ is any set of recursive functions which contains arbitrarily large functions (i.e. for every total recursive function f, their exists an h in $\mathcal{S}$ such that $h > f$ a.e.). Then from elementary recursion theory we know that $\mathcal{S}$ cannot be r.e. The next observation is then immediate.

PROPOSITION 3.4. *There is a measured set $\mathcal{G}$ which contains the set of tape constructable (real time countable) functions. However, the set of tape constructable (real time countable) functions cannot be r.e. and therefore cannot be a measured set.*

THEOREM 3.5. (Blum Compression Theorem Restricted to Total Functions). *Let ϕ be a complexity measure and $\{\gamma_i\}$ a measured set. Then there exist recursive functions g and h such that for every total γ_i, $\exists$ a total recursive function f satisfying:*

(1) $\phi_j \simeq f \Rightarrow \Phi_j > \gamma_1$ *a.e.,*

(2) $\phi_{g(i)} \simeq f$ *and* $\Phi_{g(i)}(x) \leq h(x, \gamma_i(x))$ *a.e. x.*

We can easily restate this theorem in terms of our complexity classes.

THEOREM 3.6. (Weak Compression Theorem). *Let Φ be a complexity measure and $\{\gamma_i\}$ a measured set. Then $\exists$ a total h' such that for every "sufficiently large"*

total[9] γ_i,

$$R^{\Phi}_{\gamma_i} \subset R^{\Phi}_{h' \circ \gamma_i}.$$

PROOF. Let $b(x)$ be any increasing recursive function. γ_i sufficiently large will mean $\gamma_i \geq b$ a.e. Let h be as in Theorem 3.5. Then

$$h'(y) = \max_{z \leq b^{-1}(y)} h(z, y)$$

satisfies the theorem where $b^{-1}(y) = \mu z[b(z) \geq y]$. QED

Theorem 3.6 is a weak version of Theorem 3.5 in the sense that f in $R^{\Phi}_{h' \circ \gamma_i} - R^{\Phi}_{\gamma_i}$ only guarantees that $\phi_j \simeq f \Rightarrow \Phi_j > \gamma_i$ i.o. rather than $\Phi_j > \gamma_i$ a.e. We also note that Theorem 3.6 does not express the fact that an f in $R^{\Phi}_{h' \circ \gamma_i} - R^{\Phi}_{\gamma_i}$ can be found effectively from the index i.

We are now in a position to make a very *basic observation*. Namely, a salient feature of all known complexity hierarchy results is that some restriction has been placed on a function which bounds the resource. In Example 3.1, it was necessary to have L_2 be tape constructable. Example 3.2 had the requirement that t_2 be real time countable. And finally, in Theorem 3.6, the bounding functions were restricted to a measured set. In this chapter, we shall make precise just how essential these conditions are.

It is appropriate to comment briefly on the distinction between hierarchy theorems as in Example 3.1 or 3.2, and the general hierarchy result expressed in Theorem 3.6.

Using the terminology of Constable [7], the hierarchies in the former case are examples of "downward diagonalization." We may think of Examples 3.1 and 3.2 to be of the form:

$$\inf \frac{h \circ t}{\gamma_i} \to 0 \Rightarrow R_{\gamma_i} - R_t \neq \varnothing$$

and thus

$$\lim \frac{h \circ t}{\gamma_i} \to 0 \Rightarrow R_t \subset R_{\gamma_i},$$

where γ_i indicates a total function in some measured set. Intuitively, we see how much below γ_i we need to be in order to induce a class smaller than R_{γ_i}. Theorem 3.6 represents an "upward diagonalization" in that we show that increasing γ_i by some h insures that $R_{\gamma_i} \subset R_{h \circ \gamma_i}$.

This distinction is studied further in [7]. For our purposes we need only be aware that the distinction exists. It is more important for us to note than any measured set *induces a uniform (in some h) upward hierarchy*. That is, given any γ_i there is a uniform way (composition by h) to extend γ_i so as to induce a larger class of functions.

3.2. THE GAP THEOREM. Informally, Theorem 3.6 guarantees that "in certain complexity ranges, we need only increase the computing power by some fixed function in order to achieve a genuine increase in computing capability." In contrast, Theorem 3.7 will show that "there exists arbitrarily large gaps in some complexity ranges where no new computation is performed." The following theorem was first proven by Trachtenbrot [23] and independently by this author.

[9] $h' \circ \gamma_i$ denotes composition of function h' with function γ_i; that is, $(h_i \circ \gamma_i)(x) = h'(\gamma_i(x))$. $\subset$ denotes proper inclusion.

THEOREM 3.7. (Gap Theorem). *Let Φ be a complexity measure, g a nondecreasing recursive function such that $(\forall x)\ g(x) \geq x$. Then $\exists$ an increasing recursive function t such that*

PROOF. Define t as follows:

$t(0) = 1,$

$t(n) = \mu k > t(n-1)\{(\forall i \leq n)([\Phi_i(n) < k] \quad \text{or} \quad [\Phi_i(n) > g(k)])\}.$[10]

(1) $(\forall n)\ k$ exists, since $(\forall i \leq n)$ if $\Phi_i(n)\uparrow$ then $(\forall k)\ \Phi_i(n) > g(k)$ and if $\Phi_i(n)\downarrow$ then $(\exists k)\ \Phi_i(n) < k$.

(2) k can be found recursively, since Φ is a complexity measure and thus $[\Phi_i(n) < k]$ and $[\Phi_i(n) > g(k)]$ are recursive predicates.

(3) t satisfies the theorem, since $n \geq i$ implies that either $\Phi_i(n) < t(n)$ or $\Phi_i(n) > g \circ t(n)$. QED

We observe that an arbitrarily large t can be found to satisfy Theorem 3.7. Suppose we want $t(n) > r(n)$, then define

$$t(0) = r(0) + 1,$$

$$t(n) = \mu k > \max(t(n-1), r(n))\{\cdots\}.$$

COROLLARY 3.8. (Weak Gap Theorem). *Let $\hat{\Phi}$ be a complexity measure and let g be any recursive function such that $(\forall x)\ g(x) \geq x$. Then there exists an arbitrarily large, increasing, recursive function t:*

$$I_t^{\Phi} = I_{g \circ t}^{\Phi} \qquad \text{and consequently} \qquad R_t^{\Phi} = R_{g \circ t}^{\Phi}.$$

PROOF. Immediate from Definitions 2.3, 2.4, Theorem 3.7 and the preceding observation.

COROLLARY 3.9.[11] *Let Φ and $\hat{\Phi}$ be any two measures using the same indexing. Then there exists an arbitrarily large increasing recursive t:*

$$I_t^{\Phi} = I_t^{\hat{\Phi}} \qquad \text{and thus} \qquad R_t^{\Phi} = R_t^{\hat{\Phi}}.$$

PROOF. Let $h(x, m)$ be as in Proposition 2.10. Choose a monotone g such that $(\forall x)\ g \circ t(x) \geq h(x, t(x))$ for all sufficiently large t [i.e. $t(x) \geq x$]. Now use Corollary 3.8 to choose arbitrarily large t': $I_{t'}^{\Phi} = I_{g \circ t'}^{\Phi} = I_{g \circ g \circ t'}^{\Phi}$ (that is, construct a t', $g \circ g \circ t'$ gap).

Then

$$I_{g \circ t'}^{\hat{\Phi}} \subseteq I_{g \circ g \circ t'}^{\Phi} = I_{g \circ t'}^{\Phi},$$

$$I_{g \circ t'}^{\Phi} = I_{t'}^{\Phi} \subseteq I_{g \circ t'}^{\hat{\Phi}}.$$

Thus, $t = g \circ t'$ satisfies the corollary. QED

Suppose $\Phi = \langle\{\phi_i\}, \{\Phi_i\}\rangle$ and $\hat{\Phi} = \langle\{\hat{\phi}_i\}, \{\hat{\Phi}_i\}\rangle$ refer to the "time" measure of two radically different computers. Recall the isomorphism between acceptable indexing. Then, using the technique of Proposition 2.10 and Corollary 3.9, it is easy to show that no matter how much "better" one computer may seem compared to

[10] $\mu k > t(n-1)\{P(k)\}$ denotes the least k such that $k > t(n-1)$ and $P(k)$ holds.

[11] Also proven independently by E. McCreight in his thesis.

the other, there will be arbitrarily large t such that the sets of functions computable in time t is the same for both computers.

Corollary 3.8 is not a "speedup" theorem. Rather, it states that there is no computation which "behaves" in the range $(t, g \circ t)$. Corollary 3.8 is a weak version of Theorem 3.7 in the sense that the condition $[\Phi_i > t \text{ i.o.} \Rightarrow \Phi_i > g \circ t \text{ i.o.}]$ would suffice to have $I_t^\Phi = I_{g \circ t}^\Phi$. Using an intricate priority argument, Constable [6] has shown that a "weak operator gap" holds for all measures. An observation due to Blum [1] shows that this "weak" gap is the best possible for many operators.

It is interesting to note how "small" (relative to g) an increasing recursive t can be found satisfying Theorem 3.7.

COROLLARY 3.10. *Let Φ, g, and t be as in Theorem 3.7. Let $f(y) = g(y + 1)$. Then $(\forall x)\, t(x) \leq b(x)$ where $b(0) = 1$ and $b(x) = f^{(x+1)} \circ b(x - 1)$ for $x \geq 1$.*

PROOF. Review the construction of t in Theorem 3.7. Look at the $n + 1$ intervals.

$$\{t(n-1)+1, g \circ [t(n-1)+1]\},$$
$$\{g \circ [t(n-1)+1]+1, g \circ [g \circ [t(n-1)+1]+1]\} \cdots,$$
$$\{f^{(n)} \circ t(n-1)+1, g \circ [f^{(n)} \circ t(n-1)+1]\}.$$

$$f^{(j)} \circ t(n-1) + 1 \leq \Phi_i(n) \leq f^{(j+1)} \circ t(n-1)$$

or

$$(\forall i \leq n)\, \Phi_i(n) \leq f^{(n+1)} \circ t(n-1).$$

The corollary then follows by the construction of t.

Example 3.11: Let Φ be any measure and let $g(x) = x + 1$. Then, by the preceding corollary, letting $f(x) = x + 2$, there exists $t(x) \leq b(x)$ such that $R_t^\Phi = R_{t+1}^\Phi$ where

$$b(0) = 1,$$
$$b(x) = b(x-1) + 2(x+1) \leq x^2 + 3x + 1.$$

Corollary 3.10 provides an estimate of the growth rate of t relative to g. In particular, if g is primitive recursive in Grzegorczyk [9] class $\mathcal{E}^n$, then t can be bounded by a function in $\mathcal{E}^{n+1}$. With a judicious choice of encoding instantaneous descriptions we know that M is primitive recursive (elementary) if Φ refers to the usual meaning of computation steps, tape or tape reversals. In general M need not be primitive recursive. It should be apparent that if M and g are in $\mathcal{E}^n$, then the t of Theorem 3.7 is in $\mathcal{E}^{n+1}$.

3.3. THE HONESTY THEOREM AND NAME INVARIANT DOWNWARD GAPS. After the gap theorem was announced, McCreight and Meyer proved a powerful theorem about "ways to name complexity classes." Their theorem, interesting in its own right, takes on added significance when viewed in contrast with the Blum compression theorem and the gap theorem.

THEOREM 3.12. (McCreight-Meyer Honesty Theorem). *Let Φ be any complexity measure. Then there exists a measured set $\mathcal{G}$ such that for every total recursive function t, there is a total t' in $\mathcal{G}$: $I_t^\Phi = I_{t'}^\Phi$ and $\therefore R_t^\Phi = R_{t'}^\Phi$.*

Let $\mathcal{G}$ be as in Theorem 3.12. Using the compression theorem, let h be such that for all sufficiently large total t' in $\mathcal{G}$, $R_{t'}^\Phi \subset R_{h \circ t'}^\Phi$. Then, even though we may find a large "gap t" such that $R_t^\Phi = R_{h \circ t}^\Phi$, $\exists$ a total t' in $\mathcal{G}$. $R_t^\Phi = R_{h \circ t}^\Phi = R_{t'}^\Phi \subset R_{h \circ t'}^\Phi$.

This leads McCreight and Meyer to suggest that "theorems like the gap theorem which appears to be making statements about the structure of t-computability classes may only be indicating the properties of badly chosen *names* for the classes."

This remark has a good deal of merit. Indeed, we shall return to this distinction between "structural" and "naming" properties. However, we should not infer that one type of property is more important than the other. We begin to view a set of functions as a "coordinate system" for the complexity classes or as a set of names for the classes. A modified gap argument will show that for some purposes, bad names are unavoidable (or equivalently, that certain naming properties hold in any "coordinate system").

We first formalize the ability of some classes to induce hierarchies.

Definition 3.13: A class of partial functions $\mathcal{G}$ is *upward extendible* (by h_u w.r.t. Φ) if for every sufficiently large total g in $\mathcal{G}$, $R_g^{\Phi} \subset R_{h_u \circ g}^{\Phi}$.

The relationship between measured sets, sets of "g-honest functions [17, 15]" and "upward extension" has been well studied [1, 3, 15]. These studies (and in particular, the compression theorem) allow us to characterize a measured set (or a set of g-honest functions) as a good naming class for the purpose of extending upward in the hierarchy. Furthermore, the honesty theorem says that for each Φ we can choose such a class and still name all the Φ resource bounded complexity classes.

Definition 3.14: A class of partial recursive functions $\mathcal{G}$ is *downward extendible* (by h_d w.r.t. Φ) iff $\exists$ recursive increasing h_d: for every sufficiently large total g *in* $\mathcal{G}$,

$$R_{h_d^{-1} \circ g}^{\Phi} \subset R_g^{\Phi} \qquad \text{where again} \qquad h^{-1}(y) = \mu z[h(z) \geq y].$$

Example 3.15: By the compression theorem, for any measure Φ, $\exists$ recursive h' and g, such that for all sufficiently large Φ_i, $R_{\Phi_i}^{\Phi} \subset R_{\Phi_{g(i)}}^{\Phi}$ where $\Phi_{g(i)} \leq h' \circ \Phi_i$ a.e. Thus $\{\Phi_{g(i)}\}$ is a downward extendible class (by h' w.r.t Φ).

Example 3.16: In [15], a measure is called *proper* if for all total Φ_i, Φ_i in $R_{\Phi_i}^{\Phi}$. Let Φ be a proper and let Φ have the "parallel computation property" in the sense of Example 2.13. (The tape measure of Example 2.1 is such a measure.) Constable [7] notes that for all such measures, the set $\{\Phi_i\}$ is downward extendible.

We need one "technical" lemma, which is a slight modification of the compression theorem. This lemma is used twice: first for the observation that downward extendible classes are upward extendible and then in the proof of Theorem 3.20 (the existence of name invariant downward gaps).

LEMMA 3.17. (Modification of Compression Theorem). *Let Φ be any measure, $\mathcal{G}$ a measured set, and h_1 any recursive function. Then $\exists$ a recursive function $h_2(x, y)$ such that for all total g in $\mathcal{G}$, $\exists f$:*

$$\phi_i \simeq f \Rightarrow \Phi_i > h_1 \circ g \quad a.e.$$

$$\exists \phi_j \simeq f \quad and \quad \phi_j(x) \leq h_2(x, g(x)) \quad a.e.\ x.$$

As in Theorem 3.6, we can replace h_2 by a function of one variable h_2' if we restrict the Lemma to all sufficiently large g in $\mathcal{G}$. Obviously, $h_2' > h_1$ a.e.

PROOF. An essential lemma to the compression theorem is Theorem 7 of [1]. We make a simple modification of that theorem and show:

(1) To every partial recursive function g, there corresponds a 0–1 valued partial recursive function f *with the same domain as g such that if $\phi_i \simeq f$, then*

$$\Phi_i > h_1 \circ g \quad \text{a.e.}$$

(2) There exists a total recursive function r which takes indices for g and h_1 into an index for the corresponding f.

Let $\phi_l \simeq h_1$. Holding l fixed, we can go (via r) effectively from an index for g to an index for f. The Lemma then follows exactly as in the proof of the compression theorem.

We used the Lemma for the next easy observation.

PROPOSITION 3.18. *Let $\mathcal{G}$ be downward extendible (by h_d w.r.t. Φ), then $\mathcal{G}$ is upward extendible (by some h_u w.r.t. Φ).*

PROOF. Let g in $\mathcal{G}$ be sufficiently large (assume at least that $g(x) \geq x$ a.e. x.)

$$R^{\Phi}_{h_d^{-1}\circ g} \subset R_g^{\Phi} \Rightarrow \exists \Phi_k\colon \quad h_d^{-1} \circ g < \Phi_k \quad \text{i.o.}$$

$$\Phi_k \leq g \quad \text{a.e.}$$

Setting $h_1 = h_d$ in Lemma 3.17, we get that $\exists f$:

$$\phi_i \simeq f \Rightarrow \Phi_i > h_d \circ \Phi_k \quad \text{a.e.}$$

$$\Rightarrow \Phi_i > h_d \circ h_d^{-1} \circ g \geq g \quad \text{i.o.}$$

and $\exists \phi_j \simeq f$ with

$$\begin{aligned} \Phi_j(x) &\leq h_2(x, \Phi_k(x)) \quad \text{a.e. } x. \\ &\leq h_2(x, g(x)) \quad \text{a.e. } x. \\ &\leq h_2(g(x), g(x)) \quad \text{a.e. } x. \end{aligned}$$

The proposition is proved when we set

$$h_u(y) = h_2(y, y).$$

QED

Definition 3.19: A class of partial functions $\mathcal{G}$ is *class determining* w.r.t. Φ iff for every total recursive function t, there exists a total function g in $\mathcal{G}$ such that $R_t^{\Phi} = R_g^{\Phi}$.

We may think of any class determining $\mathcal{G}$ as a coordinate system for Φ. That is, the total functions in $\mathcal{G}$ name all the Φ complexity classes. As an immediate consequence of the following theorem, due to R. Constable and the author, we see that a class $\mathcal{G}$ cannot be both class determining and downward extendible w.r.t. any Φ.

THEOREM 3.20. (The Existence of Name Invariant Downward Gaps). *Let Φ be any measure. Let h_d be any recursive increasing function. Then there exist arbitrarily large, increasing, recursive t such that:*

$$R_g = R_t \Rightarrow R_{h_d^{-1}\circ g} = R_g.$$

That is, it is not possible to go down from R_t by h_d no matter what name (g) we use.

PROOF. Letting $\mathcal{G} = \{\Phi_i\}$, apply Lemma 3.17 with $h_1 = h_d$ to yield an appropriate h_2. That is, for every Φ_k, $\exists$ recursive f:

$$\phi_i \simeq f \Rightarrow \Phi_i > h_1 \circ \Phi_k = h_d \circ \Phi_k \quad \text{a.e.}$$

$$\exists\, \phi_j \simeq f \quad \text{and} \quad \Phi_j(x) \leq h_2(x, \Phi_k(x)) \quad \text{a.e. } x.$$

Construct arbitrarily large increasing recursive "gaplike" t:

$$\Phi_i(x) \leq t(x) \Rightarrow h_2(x, \Phi_i(x)) \leq t(x) \text{ for almost all } x.$$

It is easy to verify that such a t can be constructed. Suppose $\exists$ total g such that $R_g = R_t$ and $R_{h_d^{-1} \circ g} \subset R_g$. Then, $\exists$ total Φ_k such that

(i) $h_d^{-1} \circ g < \Phi_k$ i.o. which implies $g < h_d \circ \Phi_k$ i.o.

(ii) $\Phi_k \leq t$ a.e. which implies $h_2(x, \Phi_k(x)) \leq t(x)$ a.e. x.

But then there exists f:

$$\phi_i \simeq f \Rightarrow \Phi_i > h_d \circ \Phi_k \quad \text{a.e.}$$
$$\Rightarrow \Phi_i > g \quad \text{i.o.}$$

and $\exists\, \phi_j \simeq f$ with $\Phi_j(x) \leq h_2(x, \Phi_k(x)) \leq t(x)$ a.e. x. $\therefore f$ in $R_t - R_g$ which is a contradiction. QED

Intuitively, the proof of Theorem 3.20 consists of forming a $(h_2^{-1} \circ t, t)$ gap. We suggest that R_t may be thought of as a "limit" of a way of extending classes. That is, $\{f_j\}$ an r.e. class of recursive functions and for all j, $R_{f_j} \subset R_{h_2 \circ f_j} \subset R_{h_2^2 \circ f_j} \cdots$ reflects that h_2 is a means by which we can extend classes. It is obvious that this way is limited since there are functions which are much greater than $h^i \circ f_j$ for any i and j (h_2 recursive implies that $\{h_2^i \circ f_j\}$ is also an r.e. class of recursive functions and is therefore bounded.).

Viewing $R_t = R_g$ as a limiting class for h_2 extension makes the theorem quite intuitive. If we could go downward by h_1 applied to the name g for R_t, then a total Φ_i function would lie "within h_1" of g and hence we could "exceed R_g" by h_2 extension (that is, R_g was not a limiting class for h_2 extension). Consequently, we might say that this *naming exclusion* holds because "downward extendible implies upward extendible." No such exclusion holds for class determining and upward extension because "upward extension does not necessarily imply downward extension."

COROLLARY 3.21. *If Φ is proper and has a parallel computation capability, then $\{\Phi_i\}$ is not class determining.*

PROOF. Immediate from Example 3.16 and Theorem 3.20. In fact, $\exists$ arbitrarily large t: $(\forall i)\ R_t^{\Phi} \neq R_{\Phi_i}^{\Phi}$.

A stronger version of this corollary was first proven by McCreight and Meyer. Namely, they showed that Φ proper $\Rightarrow \{\Phi_i\}$ not class determining.

4. *Minimal Growth Rates*

4.1. MONOTONE MINIMAL GROWTH RATES. In [22], the concept of "minimal tape function growths" was introduced. It was shown that for on- and off-line Turing machines, the amount of work tape used in a computation must "grow" as a function of input length in order to recognize a nonregular set. The essence of that proof was that if the amount of tape used in a computation is not bounded by some constant for all inputs, then for infinitely many inputs the computation requires an amount of tape at least proportional to $\log n$ (n = input length) for the on-line model and $\log \log n$ for the off-line model.

We now abstract this concept as follows.

Definition 4.1: We say a complexity measure Φ has a monotone minimal growth rate if there exists a nondecreasing unbounded recursive function t:

(1) Φ_i total, unbounded, and nondecreasing almost everywhere $\Rightarrow \Phi_i(x) \geq t(x)$ a.e. x.

(2) Φ_i total, and unbounded $\Rightarrow \Phi_i(x) \geq t(x)$ i.o. x.

A minimal growth rate corresponds to a "gap" in the "low end" of the complexity hierarchy. We shall show in Example 4.4 that not all complexity measures have a monotone minimal growth rate. This contrasts with an observation by McCreight and Meyer that *all* measures have a "minimal growth rate" if we do not insist on monotonicity.

One reason to desire monotonicity is that in the context of sequence generation it is natural to insist that all bounding functions be nondecreasing. In Theorem 4.3, we will give a sufficient condition for when a measure has a monotone minimal growth rate.

Example 4.2: Let Φ be as in Example 2.1. Let $P(i, k)$ be the predicate $(\forall x)\Phi_i(x) \leq k$. It is easy to show that $P(i, k)$ is decidable for all i and k. Given i and k, we may view the Turing machine as a finite automaton and construct its set of "permissible" states. We can then effectively test if there is a sequence of moves (i.e. there is an input) which causes the machine to leave the set of permissible states (i.e. the bound is exceeded).

THEOREM 4.3. *Let Φ be any complexity measure. Let $P^{\Phi}(i, k)$ be the predicate as in Example 4.2 [i.e. $(\forall x)\Phi_i(x) \leq k$]. Then $P^{\Phi}(i, k)$ decidable for all i and $k \Rightarrow \Phi$ has a monotone minimal growth rate.*

PROOF. Define t as follows: $t(0) = 0$;

$$\text{if} \quad (\forall i \leq t(x))\{(\forall z)[\Phi_i(z) < x] \quad \text{or} \quad (\exists y \leq x)[\Phi_i(y) > t(x)]\}$$

then

$$t(x + 1) = t(x) + 1$$

else

$$t(x + 1) = t(x).$$

(1) $t(x)$ is obviously nondecreasing.

(2) $t(x)$ is unbounded; for, given any k,

$$(\exists x)(\forall i \leq k)\{(\forall z)[\Phi_i(z) < x] \text{ or } (\exists y \leq x)[\Phi_i(y) > k]\}.$$

(3) Suppose Φ_i total and unbounded. Given any n: $i \leq t(n)$, let

$$k = \max_{x<n} (\Phi_i(x)) \qquad \text{and} \qquad x_0 = \mu x[t(x) = k].$$

In order for t to be unbounded, $\exists x > x_0$:

$$(\exists y \leq x)[\Phi_i(y) > t(x) = t(x_0)].$$

Let

$$y_0 = \mu y[\Phi_i(y) > t(y) = t(x_0)].$$

By definition of k, $y_0 \geq n$ and by definition of t, $\Phi_i(y_0) > t(y_0)$.

(4) Suppose Φ_i total, unbounded and nondecreasing almost everywhere. Choose n: $i \leq t(n)$, $(\forall y \geq n)\Phi_i(y) \leq \Phi_i(y + 1)$, and $\Phi_i(n) \leq t(n)$. Then

$$(\forall y \geq n)t(y + 1) = t(y) + 1 \Rightarrow \Phi_i(y) > t(y) \Rightarrow \Phi_i(y + 1) \geq t(y + 1).$$

QED

It is easy to exhibit a measure which does not have a monotone minimal growth rate.

Example 4.4: It is known that R^1 = class of primitive recursive functions con-

tains "arbitrarily slow growing" nondecreasing functions. Let Φ be any complexity measure, S an infinite recursive set: $i \in S \Rightarrow \Phi_i$ total.

Let

$$\hat{\Phi}_i(x) = \begin{cases} \Phi_i(x) & \text{if } i \notin S \\ f_j(x) & \text{if } i \text{ is the } j\text{th largest number in } S, \end{cases}$$

where $\{f_i\}$ is an enumeration of R^1. Then $\hat{\Phi}$ cannot have a nondecreasing minimal growth rate, and consequently $P^{\hat{\Phi}}(i, k)$ is not decidable for all i and k.

Definition 4.5: Φ has a *weak minimal growth rate* if there exists a recursive function t with $\liminf t \to \infty$ satisfying: Φ_i unbounded on dom $\Phi_i \Rightarrow \Phi_i \geq t$ i.o.

THEOREM 4.6. (McCreight and Meyer [15]). *Every measure Φ has a weak minimal growth rate.*

Example 4.4 and the preceding theorem provide an interesting result about the recursive functions.

COROLLARY 4.7. *There exists a recursive function t such that $\liminf t \to \infty$ but there does not exist any unbounded, nondecreasing recursive function b: $b \leq t$ a.e.*

PROOF. Let $\hat{\Phi}$ be the measure of Example 4.4. Let t be a weak minimal growth rate function for $\hat{\Phi}$. That is, $\liminf t \to \infty$ and $\hat{\Phi}_i$ unbounded $\Rightarrow \hat{\Phi}_i \geq t$ i.o. Now suppose $\exists$ an unbounded recursive $b \leq t$ a.e. Then $\exists$ a nondecreasing unbounded elementary function f_j such that $f_j < b$ a.e. $\therefore \hat{\Phi}_i = f_j$ which implies $\hat{\Phi}_i$ unbounded and $\hat{\Phi}_i < b \leq t$ a.e. contradicting the definition of t as a weak minimal growth rate function. QED

5. *Conclusion*

We want to attempt to clarify the distinction between "properties about the structure of t computability classes" and "properties about names for the classes."

Important structural properties concern the partially ordered set $\langle\{R_t^{\Phi}\}, \subset\rangle$. For example, consider the following interesting fact about the $\langle\{R_t^{\Phi}\}, \subset\rangle$ structure [11, 15]:

$$(\forall \text{ sufficiently large classes } R_t^{\Phi})\ (\exists \text{ a class } R_{t'}^{\Phi})\ (\forall \text{ classes } R_f^{\Phi})[R_t^{\Phi} \cup R_{t'}^{\Phi} \neq R_f^{\Phi}].$$

On the other hand, there are properties which can not be expressed using only classes for variables; these properties require that we allow functions (names for the classes) as variables. We have seen many examples of such properties (the compression, gap, and honesty theorems).

We shall call such properties *naming properties*. It is obvious that these properties play an important role in the study of complexity classifications. In general, we can investigate naming properties w.r.t. the total functions of any class determining set $\mathcal{G}$. That is, the function variables are restricted to the total functions of $\mathcal{G}$. In particular, $\mathcal{G}$ might be (the set of all partial recursive functions) and then a function variable could be any total recursive function.

A further distinction should be made concerning naming properties. Some properties not only must be expressed using function (name) variables but, moreover, hold precisely because of the existence of certain names. The gap theorem reflects such a property. This contrasts with Theorem 3.25, which shows that the existence of downward gaps is in some sense name invariant.

We can call a naming property P name-dependent if there exist class determining

sets $\mathcal{G}_1$ and $\mathcal{G}_2$ such that P holds w.r.t. the total functions of $\mathcal{G}_1$ but does not hold w.r.t. the total functions of $\mathcal{G}_2$. A naming property P is name-invariant if P holds w.r.t. the total functions of any class determining set.

The Blum axiomization allowed us to characterize computing resources or complexity measures and consequently significant advances have been made in the study of resource bounded classification. By intent, the Blum axioms are very weak. That is, every conceivable measure should satisfy these axioms but many measures satisfy the axioms which we would agree are "pathological." A significant problem remaining in "axiomatic complexity theory" relates to the question of how to classify measures, and, in particular, how to refine the Blum axioms so as to characterize precisely the "*natural*" measures (i.e. exclude pathological measures).

Given a measure Φ, let us view any class-determining set of names for Φ as a possible set of names to use. What other properties should the class have? We are suggesting that searching for a good naming class is analogous to the development of good coordinate systems in geometry. We seek those properties which are invariant for any good naming class. It is our hope that these "coordinate invariant properties" and "structural properties" will provide the basis needed for classifying measures.

ACKNOWLEDGMENTS. I would like to express my sincere gratitude to Professors R. Constable, J. Hartmanis, and J. Hopcroft for their encouragement, insight and continual guidance, Professor P. C. Fischer for the beginning of this research, and Professors A. Meyer, P. Young, E. Robertson and Dr. E. McCreight for numerous suggestions. I also want to thank the referees for their excellent comments and especially for the present proof of Theorem 4.3.

REFERENCES

1. BLUM, M. A machine-independent theory of the complexity of recursive functions. *J. ACM 14* (1967), 322–336.
2. BLUM, M. On the size of machines. *Inform. Cont. 11* (1967), 257–265.
3. BORODIN, A. Computational complexity and the existence of complexity gaps. Dep. of Comp. Sci. Tec. Rep. No. 69–41, Cornell U. (Aug. 1969).
4. BORODIN, A., CONSTABLE, R., AND HOPCROFT, J. Dense and non-dense families of complexity classes. *IEEE Tenth Annual Symp. on Switching and Automata Theory* (Oct. 1969), pp. 7–19.
5. CONSTABLE, R. Extending and refining hierarchies of computable functions. Comp. Sci. Tech. Rep. No. 25, U. of Wisconsin (June 1968).
6. CONSTABLE, R. The operator gap. *IEEE Tenth Annual Symp. on Switching and Automata Theory* (Oct. 1969), 20–26.
7. CONSTABLE, R. Upward and downward diagonalization over axiomatic complexity classes. Dep. of Comp. Sci. Tech. Rep. No. 69-32, Cornell U. (March 1969).
8. FISCHER, R., HARTMANIS, J., AND BLUM, M. Tape reversal complexity hierarchies. *IEEE Ninth Annual Symposium on Switching and Automata Theory* (Oct. 1968), 373–382.
9. GRZEGORCZYK, A. Some classes of recursive functions. *Rozprawy Matematyczne (Warsaw) 4* (1953), 1–46.
10. HARTMANIS, J. Computational complexity of one-tape Turing machine computations. *J. ACM 15* (1968), 325–339.
11. HARTMANIS, J., AND STEARNS, R. E. On the computational complexity of algorithms. *Trans. Amer. Math. Soc. 117* (May 1965), 285–306.
12. HENNIE, F., AND STEARNS, R. E. Two-tape simulation of multitape Turing machines. *J. ACM 13* (Oct. 1966), 533–546.
13. LANDWEBER, L. H., AND ROBERTSON, E. R. Recursive properties of abstract complexity classes. *ACM Symposium on Theory of Complexity* (May 1970), pp. 31–36.

14. LEWIS, F. D. Decision problems for complexity classes of computable functions. *ACM Symposium on Theory of Computing* (May 1970), pp. 22–30.
15. MCCREIGHT, E. M., AND MEYER, A. R. Classes of computable functions defined by bounds on computation. *ACM Symposium on Theory of Computing* (May 1969), pp. 79–88.
16. MEYER, A., AND FISCHER, P. On computational speedup. *IEEE Ninth Annual Symposium on Switching and Automata Theory* (Oct. 1968), pp. 351–355.
17. MEYER, A., AND RITCHIE, D. A classification of functions by computational complexity. *Proc. Hawaii International Conf. on System Sciences* (1968), pp. 17–19.
18. RABIN, M. Degree of difficulty of computing a function and a partial ordering of recursive sets. Tech. Rep. No. 2, Hebrew U., Israel (1960).
19. RITCHIE, R. Classes of predictably computable functions. *Trans. Amer. Math. Soc. 106*, 1 (1963), 139–173.
20. ROGERS, H., JR. Gödel numberings of partial recursive functions. *J. Symbolic Logic 23*, 3 (Sept. 1958), 331–341.
21. ROGERS, H., JR. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York, 1967.
22. STEARNS, R., HARTMANIS, J., AND LEWIS, P., II. Hierarchies of memory limited computations. *IEEE Sixth Annual Symposium on Switching Circuit Theory and Logical Design* (1965), pp. 179–190.
23. TRACHTENBROT, B. A. *Complexity of Algorithms and Computation* [in Russian]. Novosibrisk, 1967.
24. YAMADA, H. Real-time computation and recursive functions not real-time computable. *IRE Trans. EC-11* (1962), 753–760.
25. YOUNG, P. Toward a theory of enumerations. *IEEE Ninth Annual Symposium on Switching and Automata Theory* (Oct. 1968), pp. 334–350.
26. YOUNG, P. A note on dense and nondense families of complexity classes. Comp. Sci. Tech. Rep. CSD TR 40, Purdue U. (August 1969).

RECEIVED JUNE 1970; REVISED JUNE 1971