SETS THAT DON'T HELP*

Nancy Lynch
Tufts University, Medford, Mass.

Albert Meyer and Michael Fischer
M. I. T., Cambridge, Mass.

Abstract

This paper contains several results yielding pairs of problems which don't help each other's solution, and therefore which may be said to be complex for "different reasons." Statements are formalized and results proved within Blum complexity theory, generalized to relative algorithms. The approach is fairly intuitive; all details appear in [1] and [2].

I. Introduction

There are many known interesting problems for which any computed solution must use very large amounts of time on infinitely many instances of the problem. Several decision procedures for logical theories [3][4] fall into this category, as well as certain natural problems involving regular expressions [5][6]. However, the method of proof of difficulty for all of these problems is essentially the same: all are shown to be sufficiently expressive to encode the computations of a diagonalizing Turing machine, on some infinite set of instances of the problem. It is therefore possible that these problems might be difficult "for the same reason." At any rate, we cannot yet prove that having a table of answers for one of these problems does not reduce the solution of any other of these problems to triviality.

As another example, all of the NP-complete problems of Cook [7] and Karp [8], if they are difficult, must be difficult for the "same reason." This is because the time required to compute the solution to any one of these problems can be lowered to a polynomial bound if we have a precomputed table of the answers to any other of the problems.

It would be most desirable to have natural problems which are provably difficult for "different reasons." We begin by proving the existence of some pair of recursively solvable problems (not necessarily interesting ones) with this property. This is then strengthened in two ways to enable us to fix one of the problems, presumably as a natural one.

We follow a machine-independent approach similar to that of Blum [9], although the results may also be stated and proved in terms of Turing machine time or space.

*Work reported herein was supported in part by Project Mac, an M.I.T. research program sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract Number N00014-70-A-0362-0001. Reproduction in whole or in part is permitted for any purpose of the United States Government.

II. Notation

We use [10] for the notation of recursive function theory. In addition:

"a.e." ("almost everywhere") will mean "for all but a finite number of arguments." Similarly, "i.o." ("infinitely often") will mean "for infinitely many arguments".

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

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

To discuss ways in which one problem lowers the complexity of another, we require a formalism for computation using a set for help. We use "relative algorithms" [10], which are partial recursive functions of one set variable and one integer variable. We assume an effective enumeration of relative algorithms, writing "$\varphi_i^{(\ )}$" for the i^{th} function in this enumeration. We write "$\varphi_i^{(X)}$" for the partial X-recursive function computed by $\varphi_i^{(\ )}$ using set X. We write $\varphi_i^{(\emptyset)}$ as simply "φ_i", and thus obtain an acceptable Gödel numbering for the partial recursive functions. The standard model for a relative algorithm is Davis' oracle Turing machine [10].

We use the following to define a measure of complexity of a relative algorithm:

Definition: A relative complexity measure $\{\Phi_i^{(\ )}\}$ is a sequence of relative algorithms satisfying:

(1) $(\forall i,X)$ domain $\Phi_i^{(X)}$ = domain $\varphi_i^{(X)}$,

and (2) There exists $\psi^{(\ )}$, a relative algorithm, such that:

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

These two properties are similar to the axioms of Blum for the complexity of partial recursive functions [9], and are used in the study of relative complexity in [1] and [2]. The most natural examples of relative complexity measures are the time and space measures on oracle Turing machines. We show in [1] and [2] that these simple axioms are sufficiently powerful to imply an invariance theorem, stating that any pair of relative complexity measures is related by a fixed recursive function. This allows us to prove results about any convenient measure, and then conclude related results for other measures. The three theorems in this paper may be proved in this way, with the Turing machine space measure a very convenient one, or directly from the axioms.

We write $\Phi_i^{(\phi)}$ as simply "Φ_i", and obtain a complexity measure on the set of partial recursive functions, in the sense of Blum. We call all of the functions Φ_i "running times."

Assume A is a set, $f \in R_1$, and b is a total function of one variable. Then:

"$\text{Comp}^{(A)} f \leq b$ i.o." means:

$$(\exists i)[(\varphi_i^{(A)} = f) \text{ and } (\Phi_i^{(A)} \leq b \text{ i.o.})].$$

Similarly,

"$\text{Comp}^{(A)} f > b$ i.o." means:

$$(\forall i)[(\varphi_i^{(A)} = f) \Rightarrow (\Phi_i^{(A)} > b \text{ i.o.})].$$

We use analogous definitions for "a.e." in place of "i.o." Also, we write "Comp f" instead of "$\text{Comp}^{(\phi)} f$." And finally, if f is 0-1 valued, so that $f = C_B$ for some set B, then we write "$\text{Comp}^{(A)} B$" instead of $\text{Comp}^{(A)} C_B$.

III. Theorems about Sets that Don't Help

Our first theorem produces two recursive sets that don't help each other's computation (i.e. which are complex for "different reasons.") It is a subrecursive analog to the Friedberg-Muchnik theorem of recursion theory [10], which produces two sets not permitting each other's computation. The function h in the statement of the theorem results from overhead involved in simulation, and should be thought of as small relative to t_B and t_C:

Theorem 1: There exists $h \in R_2$ satisfying the following:
For all sufficiently large running times t_B and t_C, there exist recursive sets B and C such that:

$$\text{Comp } B \leq h \circ t_B \text{ a.e.},$$

$$\text{Comp } C \leq h \circ t_C \text{ a.e.},$$

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

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

The basic proof method of abstract complexity theory is diagonalization. Sacks, Spector, etc. [10][11] have developed extensive diagonalization machinery for theorems about degrees of unsolvability, much of which has not yet been used in complexity theory. This theorem and also Theorem 3 have proofs which appear to require the use of priority constructions, as originated by Friedberg and Muchnik.

We omit the proof of Theorem 1 in favor of an outline of the proof of Theorem 3. We note, however, that one method of proof for Theorem 1 is to simultaneously construct the two sets B and C using diagonalization and a finite-injury priority construction. An alternative proof follows from Trachtenbrot's construction [12] of a "nonautoreducible set," reconstructed in [1] and [2], which also used a finite-injury priority argument. In either case, there is a small recursive bound on the number of injuries to any condition.

We note that two corollaries follow from the proof of Theorem 1:

Corollary 1.1: There exists $h \in R_2$ satisfying the following:
For any sufficiently large running time t, there exists an infinite collection of recursive sets $\{A_i\}$ such that:

$$(\forall i)\ \text{Comp } A_i \leq h \circ t \text{ a.e.},$$

and

$$(\forall i,j)\ \text{Comp}^{(A_i)} A_j > t \text{ a.e.}$$

Corollary 1.2: Assume $\{\Phi_i^{(\ )}\}$ represents Turing machine space measure.
For all sufficiently large total tape-constructable functions t_B and t_C, there exist recursive sets B and C such that:

$$\text{Comp } B \leq 2^{t_B} \text{ a.e.},$$

$$\text{Comp } C \leq 2^{t_C} \text{ a.e.},$$

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

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

In Theorem 1 and its Corollaries, both sets are constructed by diagonalization. As a step toward making the result more applicable to natural problems, we would like to be able to fix one of the sets arbitrarily (i.e. as some natural set). Theorems 2 and 3 require a "compression" condition on the complexity of set A, but otherwise allow us to fix A arbitrarily.

Theorem 2: There exists $h \in R_2$ satisfying the following:
For any recursive set A and any recursive function t with the property that Comp $A > h \circ t$ i.o., there exist arbitrarily complex recursive sets B such that:

$$\text{Comp}^{(B)}A > t \text{ i.o.}$$

Thus, it is impossible to use set B to lower the complexity of A below the bound t.

The proof, which we again omit, is based on an idea of Machtey [13], and is a diagonalization essentially similar to the initial segment constructions in [10]. There is no priority involved. Once again, we give the sharper bound for the Turing machine space measure:

Corollary 2.1: Assume $\{\Phi_i^{(\ )}\}$ represents Turing machine space measure.
For any recursive set A and any recursive function t with the property that Comp A > t i.o., there exist arbitrarily complex recursive sets B such that:

$$\text{Comp}^{(B)}A > t \text{ i.o.}$$

The third theorem is similar to Theorem 2, but with a stronger type of lower bound on the complexity of A:

Theorem 3: There exists $h \in R_2$ satisfying the following:
For any recursive set A and any total running time t, if Comp A > h o t a.e., then there exist arbitrarily complex recursive sets B such that

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

Proof: We present an intuitive outline. Complete details may be found in [1] and [2].

The method of proof is a finite-injury priority argument with no apparent recursive bound on the number of injuries for each condition. Briefly, the construction of B proceeds as follows:

The set B must satisfy two conditions. We must have $\text{Comp}^{(B)}A > t$ a.e., and B must have a given minimal complexity. The second of these conditions is achieved by interweaving a Rabin diagonalization construction [9] with the main construction, and poses no particular problems. The first condition is much more difficult.

We need to insure that, for any index i, $\Phi_i^{(B)} \leq t$ i.o. implies $\varphi_i^{(B)} \neq C_A$. We thus have an infinite sequence of conditions to satisfy, one for each i. To prevent conflict, we assign smaller indices higher priority than larger indices.

B is constructed in an effective sequence of stages executed in numerical order, with membership of n in B determined at stage numbered n. Thus, B will be a recursive set.

Three major devices are used in the proof. At any time during the construction, we may have:

(1) One tentative commitment for some index i, to an extension of the part of B already defined,

(2) Any number of tentatively cancelled indices i,

and

(3) Any number of permanently cancelled indices i.

They indicate the following:

If we have a tentative commitment for i, it means we plan to extend the definition of B in such a way as to satisfy

$$\Phi_i^{(B)}(x) \leq t(x)$$

for a certain argument x. If i is tentatively cancelled, it means that we have succeeded in defining B in this way (i.e. there has been no interference from indices of higher priority than i). If i is permanently cancelled, it means that it was already tentatively cancelled, and we have discovered that $\varphi_i^{(B)}(x) \neq C_A(x)$ for the argument x used in i's tentative commitment. (If we instead discover that $\varphi_i^{(B)}(x) = C_A(x)$, we will remove i's tentative cancellation and try again to find a new tentative commitment for i.)

A conflict may arise if it becomes desirable to make tentative commitments for two different indices at the same time; they might require different definitions of B. To resolve such conflicts, we always choose to satisfy the condition corresponding to the index of higher priority.

We choose a monotone increasing total running time t_B to be an a.e. lower bound on B's complexity. That is, interwoven into the following construction will be a Rabin diagonalization insuring that Comp B > t_B a.e. We now describe the general stage of the construction.

We assume without loss of generality that $t > \lambda x[x]$.

Stage n: (Define $C_B(n)$)

(a) Setting up tentative commitments

See if there exists an index i, an argument x, and a finite extension E of the current definition of B such that:

(a1) $i \leq n$, i is not permanently cancelled or even tentatively cancelled, and i is of higher priority than any index for which there is a current tentative commitment,

(a2) $t_B(n-1) < t(x) \leq t_B(n)$, and

(a3) $\Phi_i^{(E)}(x) \leq t(x)$.

If so, consider the smallest such index i and, for i, the x with the smallest such t(x), and for i and x, the first such E in the lexicographic ordering. Establish E as a new tentative commitment for i, and remove any previous tentative commitment.

In either case, define $C_B(n)$ according to whatever is now the current tentative commitment. If there is no current tentative commitment, let $C_B(n) = 0$.

Go on to substage (b).

(b) Converting tentative commitments to tentative cancellations

See if $n \geq$ the largest value in the current tentative commitment. If so, we have succeeded in defining B consistently with the current tentative commitment, so we change the tentative commitment for i to a tentative cancellation of i.

If not, we make no change. In either case, we go on to substage (c).

(c) Converting tentative cancellations to permanent cancellations

For any current tentative cancellation of an index i, established via an argument x and an extension E, see if $C_A(x)$ can be computed within measure $t_B(n)$. If so, and if $\varphi_i^{(E)}(x) \neq C_A(x)$, we have succeeded in insuring that $\varphi_i^{(B)} \neq C_A$, and so we convert the tentative cancellation of i to a permanent cancellation of i. On the other hand, if $\varphi_i^{(E)}(x) = C_A(x)$, we have failed, so we just remove the tentative cancellation of i, leaving i open for a new tentative commitment at a later stage.

END OF CONSTRUCTION

We note that stage n requires measure not much greater than $t_B(n)$.

The key fact in the verification is that no index i can become tentatively cancelled infinitely many times. Assuming this fact for the moment, we see that all the conditions

$$\Phi_i^{(B)} \leq t \text{ i.o. implies } \varphi_i^{(B)} \neq C_A$$

will eventually be satisfied. This is because eventually all higher priority indices will be unable to interfere with a tentative commitment for i being made and converted to a tentative cancellation of i. Then, since

$$\Phi_i^{(B)} \leq t \text{ i.o.},$$

such tentative cancellations will be made repeatedly, each either becoming a permanent cancellation or being removed in substage (c). Since i cannot become tentatively cancelled i.o., it will eventually become permanently cancelled, satisfying the condition.

It remains to see why no index may be tentatively cancelled infinitely often. Assume the contrary and let i be the smallest index that is tentatively cancelled infinitely often. We will use i to help construct a program for C_A which uses no oracle set and requires measure not much greater than t i.o., in contradiction to the hypothesized lower bound on the complexity of A. Except for a finite patch for small arguments, the new program acts as follows:

On argument x, it goes through successive stages of the given construction of B through stage n-1, where

$$t_B(n-1) < t(x) \leq t_B(n).$$

Then it checks to see if at stage n of the construction of B, a tentative commitment would be made for i, via argument x and some extension E. If so, and if x is sufficiently large, we know that this tentative commitment will eventually be converted to a tentative cancellation of i, and this tentative cancellation must eventually be removed. Thus,

$$\varphi_i^{(E)}(x) = C_A(x),$$

so the program simply computes $\varphi_i^{(E)}(x)$ and outputs the answer. If such a tentative commitment would not be made, the program will revert to an alternative method of computing C_A.

There will be infinitely many x for which an appropriate tentative commitment is made. For each of these x, the measure required to compute $C_A(x)$ is approximately given by the measure needed to simulate all stages up through stage n-1 in the construction of B (roughly $t_B(n-1)$), to recognize whether a tentative commitment would be made (roughly t(x)), and to compute $\varphi_i^{(E)}(x)$ (roughly t(x), since $\Phi_i^{(E)}(x) \leq t(x)$).

We can formally sum this up by saying that Comp $A \leq h \circ t$ i.o., contradicting the hypotheses on A.

QED

As before, careful analysis of the proof will yield a specific result for the Turing machine space measure:

Corollary 3.1: Assume $\{\Phi_i^{(\ )}\}$ represents Turing machine space measure.
For any recursive set A and any sufficiently large total tape-constructable function t, if

$$\text{Comp } A > 2^t \text{ a.e.},$$

then there exist arbitrarily complex recursive sets B such that

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

IV. Additional Questions

There are two directions in which to proceed from here. One, as we have already mentioned, is to obtain similar results in which both sets A and B are natural sets. The second is to strengthen the abstract complexity-theoretic result to refer to "helping" without any reference to a fixed lower bound.

For this second direction, the problem of defining "helping" for arbitrary recursive functions arises. In [14], several possible definitions are given, and all are shown to be equivalent. Using one of these definitions, we may formulate the following conjecture:

Conjecture: There exists $h \in R_2$ with the following property:

$(\forall A$, recursive$)(\exists B$, arbitrarily complex and recursive)

$(\forall i)[(\varphi_i^{(B)} = C_A) \Rightarrow (\text{Comp } A < h \circ \Phi_i^{(B)} \text{ a.e.})]$.

This is a generalization of Theorems 2 and 3, while Theorem 3 essentially gives the result for functions with well-determined complexities.

References

[1] Lynch, N. Relativization of the Theory of Computational Complexity. Project Mac Technical Report 99, June, 1972. PhD Thesis, MIT mathematics department, June, 1972.

[2] Lynch, N., Meyer, A., and Fischer, M. Relativization of the Theory of Computational Complexity. Submitted to Transactions of Amer. Math. Soc., July, 1972.

[3] Meyer, A. Weak Monadic Second Order Theory of Successor is not Elementary-Recursive. Preliminary Report, May, 1972.

[4] Meyer, A. Weak S1S Cannot Be Decided. Preliminary Report, Notices of the AMS Vol. 19, No. 5, August, 1972, p. A-598

[5] Meyer, A. and Stockmeyer, L. The Equivalence Problem for Regular Expressions with Squaring Requires Exponential Space. 13th Annual Symposium on Switching and Automata Theory. IEEE, 1972.

[6] Stockmeyer, L. and A.R. Meyer. Word Problems Requiring Exponential Time: Preliminary Report, appears in this volume.

[7] Cook, S. The Complexity of Theorem-Proving Procedures. 3rd Annual ACM Symposium on Theory of Computing. May, 1971.

[8] Karp, R. Reducibility Among Combinatorial Problems. Complexity of Computer Computations. Plenum Press, 1972.

[9] Blum, M. A Machine-Independent Theory of the Complexity of Recursive Functions. JACM, Vol. 14, No. 2, April, 1967.

[10] Rogers, H. Theory of Recursive Functions and Effective Computability. McGraw-Hill. 1967.

[11] Sacks, G. Degrees of Unsolvability. Annals of Mathematical Studies, No. 55, 1963, Princeton, N. J.

[12] Trachtenbrot, B. On Autoreducibility. Dokl. Akad. Nauk. SSSR. Vol. 11 (1970), No. 3.

[13] Machtey, M. Private communication.

[14] Lynch, N. "Helping": Several Formalizations. Submitted to JSL, Feb., 1973.