On the Importance of Being Π_2-Hard

Juris Hartmanis*

TR 89-961
January 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This research was supported by NSF Research Grant DCR 85-20597.

The Structural Complexity Column

J. Hartmanis*

Department of Computer Science

Cornell University

Ithaca, New York 14853

On the Importance of Being Π_2-Hard

Abstract

In this column, we show how a variety of interesting results in theory of computation all follow from a simple observation about Π_2-complete sets of total machines. We easily derive:

a) representation independent independence results,

b) non-recursive succinctness relations between different representations of languages,

c) the existence of incomplete languages in various complexity classes.

Introduction

Goedel's 1931 paper "Über formal unetscheidbare Sätze der Principia Mathematica und verwandter Systeme" sent shock waves through the mathematical community by showing that sufficiently rich mathematical theories can not be completely and consistently axiomatized (contrary to the belief of many mathematicians, including Hilbert and von Neumann). An consistent axiomatization had to be incomplete and leave undecidable propositions. This was the end of a heroic but innocent mathematical era, epitomized by Hilbert's intellectual battlecry "Wir müssen wissen, wir werden wissen!", and the beginning of an intensive search for what is and is not effectively computable and formally provable. From this effort came the

This research was supported by NSF Research Grant DCR 85-20597.

The Structural Complexity Column

J. Hartmanis*

Department of Computer Science

Cornell University

Ithaca, New York 14853

On the Importance of Being Π_2-Hard

Abstract

In this column, we show how a variety of interesting results in theory of computation all follow from a simple observation about Π_2-complete sets of total machines. We easily derive:

a) representation independent independence results,

b) non-recursive succinctness relations between different representations of languages,

c) the existence of incomplete languages in various complexity classes.

Introduction

Goedel's 1931 paper "Über formal unetscheidbare Sätze der Principia Mathematica und verwandter Systeme" sent shock waves through the mathematical community by showing that sufficiently rich mathematical theories can not be completely and consistently axiomatized (contrary to the belief of many mathematicians, including Hilbert and von Neumann). An consistent axiomatization had to be incomplete and leave undecidable propositions. This was the end of a heroic but innocent mathematical era, epitomized by Hilbert's intellectual battlecry "Wir müssen wissen, wir werden wissen!", and the beginning of an intensive search for what is and is not effectively computable and formally provable. From this effort came the

This research was supported by NSF Research Grant DCR 85-20597.

fundamental concept of a Turing machine, the Church-Turing thesis that the intuitive concept of effective computability is precisely captured by Turing machine computability or equivalently, by the class of recursive partial functions, and the subsequent development of recursive function theory.

A milestone in this development was Emil Post's 1944 address to the American Mathematical Society, "Recursively enumerable sets of positive integers and their decision problems " [Po]. This beautiful, easily readable paper (which I urge all of you to read) clearly reveals the essence of recursively enumerable sets and their relation to Goedel's incompleteness result. It introduced the concept of reductions between problems, defined complete problems, and started the classification of recursively enumerable (r.e.) sets. It also defined what came to be known as Post's problem. This problem motivated much good work in recursive function theory and was solved only in 1956 by Friedberg and Muchnik [Fr, Mu].

The development of computer science has been strongly influenced and aided by logic and recursive function theory. Computer science borrowed heavily from this rich intellectual arsenal and it owes a great debt of gratitude to the scientist who developed this field. In particular, theoretical computer science was very strongly influenced by Turing's work and it is very close to the spirit of Post's 1944 paper. This paper beautifully reveals the computational nature of Goedel's incompleteness results and shows how "natural" and simple these results are. Post's own enthusiasm about these ideas is summarized at the end of his paper: "Indeed, if general recursive function is the formal equivalent of effective calculability, its formulation may play a role in the history of combinatory mathematics second only to that of the formulation of the concept of natural number."

Although Post may have been a bit over enthusiastic about this new field, we are deeply convinced that the concept of effective computability is one of the most important ideas in mathematics and that every mathematician and computer scientist should have a thorough understanding of what is and is not effectively computable and that every computer scientist should, furthermore, have a good understanding of computational complexity.

In this column, we illustrate how simply and elegantly we can now derive a variety of interesting results in theory of computation. In the following we make a simple observation bout Π_2-complete (or Π_2-hard) sets of total machines and show that from this observation we can derive representation independent independence results, non-recursive succinctness results between different language representations and, finally, the existence of incomplete languages in various complexity classes.

Basic Concepts and Π_2-Completeness

Let $M_1, M_2, \ldots, N_1, N_2, \ldots, D_1, D_2, \ldots$ and $G_1, G_2, \ldots$ be, respectively, the standard enumeration of Turing machines, (TM's), nondeterministic polynomial time bounded TM's, deterministic polynomial time bounded TM's and context-free grammars.

The Kleene Hierarchy, *KH*, provides an elegant classification of undecidable problems. The class Σ_1, consists of all r.e. sets and they can be characterized as the languages

$$L = \{ x \mid (\exists\, y)\ R\ [x,y] \},$$

where R is a recursive predicate. Π_1 is the corresponding class with a universal quantifier over a recursive predicate,

$$L = \{ x \mid (\forall\, y)\ R\ [x,y] \}.$$

The Σ_2 class consists of all languages of the form

$$L = \{ x \mid (\exists\, y)\,(\forall z)\, R[x,y,z] \}$$

and the Π_2 class of languages of the form

$$L = \{ x \mid (\forall\, y)\,(\exists z)\, R[x,y,z] \},$$

with R a recursive predicate.

For example, the set of all TM's such that $L(M_i)$ is infinite, is a well known Π_2 set,

$$INF = \{M_i \mid (\forall n)\,(\exists y,t)\,[\,|y| \geq n \text{ and } M_i(y) \text{ accepts in } t \text{ steps }]\}.$$

The Σ_k and Π_k classes are defined correspondingly for $k > 2$.

We say that a set A is *hard for* Π_k (Σ_k) iff for any B in Π_k (Σ_k) there exists a recur-

sive function f such that

$$x \in B \Leftrightarrow F(x) \in A.$$

A is *complete for* Π_k (Σ_k) if A is in Π_k (Σ_k) and A is hard for Π_k (Σ_k).

The following facts can be easily verified.

FACT 1. For all $k \geq 1$,

$$\Sigma_{k+1} \neq \Sigma_k \neq \Pi_k \neq \Pi_{k+1}.$$

FACT 2. The set *INF* is Π_2-complete.

Proof: Clearly,

$$INF = \{M_i \mid (\forall n)\,(\exists y,t)\,[\,|y| \geq n \text{ and } M_i(y) \text{ accepts in } t \text{ steps }]\}$$

is in Π_2.

Let A be a Π_2 set,

$$A = \{x \mid (\forall y)\,(\exists z)\,[R[x,y,z] = True]\}.$$

Then A can be recursively reduced to *INF* as follows:

$$(\forall x)\,[f(x) = M_{\sigma(x)}]$$

where

$M_{\sigma(x)}(y)$ accepts iff for all $u \leq y$ there exists a z such that $R[x,u,z] = True$.

Thus, $M_{\sigma(x)}$ is in *INF* iff x is in A. □

Since we are interested in computer science problems, and particularly, in

$$P = \{L(D_i) \mid i \geq 1\} \text{ and } NP = \{L(N_i) \mid i \geq 1\},$$

we will concentrate on total machines (i.e., machines that halt on all inputs).

The following key lemma states that a Π_2-hard property on any set of languages accepted by total machines is too complex to have an r.e. list of total machines naming all the languages with this property.

Π_2-*Lemma*: Let $T_1, T_2, \ldots,$ be an r.e. list of total machines and R any property on r.e. sets. If

$$D = \{T_i \mid R\ [L\ (T_i)] = True\}$$

is a Π_2-hard set, then there is no r.e. list of total machines T_{i_1}, T_{i_2}, . . . , such that

$$\{L\ (T_i) \mid R\ [L\ (T_i)] = True\} = \{L\ (T_{i_j}) \mid j \geq 1\}.$$

Proof: If there would be an r.e. list of machines T_{i_j}, naming all the languages in

$$\{L\ (T_i) \mid R\ [L\ (T_i)\ ] = True\},$$

then we could write D as a Σ_2 set,

$$D = \{T_p \mid (\exists_j)\ (\forall x)\ [T_{i_j}(x) = T_p(x)\ ]\}.$$

This is a contradiction since D was assumed to be Π_2-hard and by FACT 1 no such set can be in Σ_2. Thus, no r.e. list of recursive names can exist for D. □.

In the following section, we will drive some dramatic consequences of the innocuous looking Π_2-Lemma.

Independence Results

Since the halting problem for Turing machines is recursively undecidable, we know that for any sound, axiomatizable formal system there exist TM's which accept the empty set but that this fact is not provable in the system. Such independence results are about a specific machine and one can say that they are based on the "opaqueness" of TM computations and not necessarily on the complexity of the problem. There clearly are equivalent TM's which accept the empty set and for which this fact is provable in any reasonable system.

Machine independent independence results are true statements about languages not provable for *any representation of these languages*. Thus, machine independent independence results have to be based on the complexity of the property under consideration. We will consider only representations by total machines and particularly by D_i's, N_i's and G_i's. For related work and more general results of this type see [Ha1, KOR, Re1, Re2, Re3].

We now exploit our Π_2-Lemma to obtain some machine independent independence results.

Let F be an axiomatized, sound formal system. That is, in F we can prove only true theorems and the set of provable theorems is r.e.

Theorem 3: There exists an infinite set accepted by a deterministic polynomial time machine, $A = L(D_{i_o})$, such that for no D_j accepting A can it be proven in F that $L(D_j)$ is infinite.

Proof: First, we observe that the set,

$$\Delta = \{D_i \mid L(D_i) \text{ is infinite}\}$$

is Π_2-hard. To see this, note that $L(M_i)$ is infinite iff the set of valid computations (documenting the acceptance of inputs), $VAL(M_i)$, is infinite [HU]. But $VAL(M_i)$ is accepted by a polynomial-time machine,

$$VAL(M_i) = L(D_{\sigma(i)}).$$

Therefore, we have a reduction of INF to Δ and we see that Δ is Π_2-hard.

This implies that there must be D_i, $L(D_i)$ infinite, such that for no D_k equivalent to D_i can it be proven in F that $L(D_k)$ is infinite. Otherwise, for each D_i with $L(D_i)$ infinite, there would exist some D_k, $L(D_i) = L(D_k)$ and a proof in F that $L(D_k)$ is infinite, which would yield an r.e. list of names for languages accepted by machines in Δ, contradicting our Π_2-Lemma. □

By the same techniques, we can show several other results.

Corollary 4: For every sound, axiomatizable system F there exists a non-regular set A in P such that for no D_i with $L(D_i) = A$, can it be proven that

$$L(D_i) \text{ is not regular.}$$

Corollary 5: If $P \neq NP$ then there exists a set A in $NP - P$ such that for no N_i with $L(N_i) = A$, can it be proven in F that $L(N_i) \notin P$.

Corollary 6: For every sound, axiomatizable system F there exists a non-regular context free language $L(G_i)$ such that for no equivalent G_j can it be proven in F that $L(G_j)$ is not

regular.

Corollary 7: For every sound, axiomatizable system F there exists a recursive oracle A such that $P^A \neq NP^A$ and for no provably total machine M_i accepting A can it be proven in F that

$$P^{L(M_i)} = NP^{L(M_i)} \text{ or } P^{L(M_i)} \neq NP^{L(M_i)}.$$

From these results, we see that machine independent independence results appear naturally in many areas of computer science of practical importance.

Non-Recursive Succinctness of Representations

In this section, we will show that for some problems optimal algorithms must be much longer (more complex) than algorithms running is suboptimal time. Furthermore, this difference in size cannot be recursively bounded. Similar non-recursive succinctness bounds hold between many other pairs of representations of languages with different expressive power. For example, there is no recursive succinctness bound between representing deterministic (unambiguous) context-free languages by arbitrary context-free grammars and deterministic (unambiguous) context-free grammars.

Let $M_1^{[k]}, M_2^{[k]}, \ldots,$ denote a standard enumeration of clocked TM's which run in $n^k + k$ time. $TIME[n^k]$ denotes the family of languages accepted in *time* n^k.

Theorem 8: There is no recursive function R such that for all

$$L(M_i^{[3]}) \in TIME\ [n^2]$$

there exists an $M_j^{[2]}$ with

$$L\ (M_i^{[3]}\ ) = L\ (M_j^{[2]}\ ) \text{ and } R\ (\ |\ M_i^{[3]}\ |\ ) \geq |\ M_j^{[2]}|.$$

Proof: The set

$$\Omega = \{M_i^{[3]} \mid L\ (M_i^{[3]}\ ) \in TIME\ [n^3] - TIME\ [n^2]\}$$

is Π_2-complete. Therefore, if there would be a recursive succinctness bound R we could recursively enumerate Ω as follows. For $M_i^{[3]}$ we compute all $M_j^{[2]}$ such that

$$R\ [\ |\ M_i^{[3]}\ |\ ] \geq |\ M_j^{[2]}\ |$$

and accept $M_i^{[3]}$ if it is determined for all such $M_j^{[2]}$ that

$$L\ (M_i^{[3]}\ )\ \neq\ L\ (M_j^{[2]}).$$

Since Ω is Π_2-complete, this shows that there is no recursive succinctness bound between n^3-time machines and n^2-time machines computing languages in $TIME\ [n^2]$. □

These results clearly indicate that there exist, say n^3-time algorithms for n^2-time computations for which we will never derive an n^2-algorithm because of the size discrepancy between the n^3 and n^2 algorithms. It is possible that our inability to prove lower bounds for some problems is caused by the size discrepancy predicted by this theorem. It may be possible to obtain, in some cases, a nonconstructive proof that some computation runs in time n^2 but be incapable to exhibit this algorithm because of its immense size.

The same type of reasoning yields a variety of other results. The essence of all these proofs is that the *difference set* , for example

$$TIME\ [n^3]\ -\ TIME\ [n^2],$$

cannot have an r.e. list of names because the corresponding set of machines is Π_2-complete,

$$\{M_i^{[3]}\ |\ L(M_i^{[3]})\ \in\ TIME\ [n^3]\ -\ TIME\ [n^2]\}.$$

Corollary 9: If $P\ \neq\ NP$ then there is no recursive succinctness bound between representations of languages in P by deterministic and nondeterministic polynomial time machines, respectively.

Again, the essence of the proof is to show that, if $P\ \neq\ NP$, then the set

$$\Gamma\ =\ \{N_i\ |\ L(N_i)\ \in\ NP-P\}$$

is Π_2-complete. This can easily be seen by reducing INF to Γ, using delayed diagonalization [La].

Corollary 10: There is no recursive succinctness bound between representing deterministic (unambiguous) context-free languages by deterministic (unambiguous) context-free grammars and by arbitrary context-free grammars.

The essence of these proofs is again Π_2-completeness of the difference sets which can be derived using the fact that the complement of valid Turing machine computations is a context-free language [HU, Va, SS, Ha3].

Finally, with a bit more work, this method yields *Goedel's speed up theorem* that (roughly speaking) the addition of a new axiom to a formal system yields non-recursive shortening of some proofs [Go, Ha2].

Incomplete Sets

In his 1944 paper, Post raised the problem whether there exist non-recursive sets that are not Turing complete. This problem was solved twelve years later by Friedberg and Muchnik [Fr, Mu] and required the introduction of new proof techniques. The same problem appears in many generalizations of classic recursion theory as well as in computational complexity theory. In 1975, Ladner [La] solved the corresponding problem for P and NP by showing that there exist incomplete problems (under Cook and Karp reductions) in $NP - P$ if $P \neq NP$.

We will give a simple proof of this result by again exploiting the fact if $P \neq NP$, then

$$\{N_i \mid L(N_i) \in NP - P\}$$

is a Π_2-complete set. This implies that there is no r.e. list of N_i-machines for all languages in $NP - P$. On the other hand, we will show that there are r.e. lists of N_i-machines for the complete languages in NP which forces us to conclude that there must exist incomplete languages in $NP - P$.

For the sake of brevity, we will consider only $\leq_m^P$-complete sets. The following result was derived in [LLR].

Fact 11: There is a recursive list of clocked nondeterministic polynomial time machines $N_{\sigma(i,j)}$, such that

$$\{L(N_{\sigma(i,j)}) \mid i,j \geq 1\} = \{L \mid L \text{ is } \leq_m^P\text{-complete for } NP\}.$$

Proof: $N_{\sigma(i,j)}$ accepts $L(N_i)$ if D_j reduces SAT to $L\ (N_i)$, otherwise it accepts a finite variation of SAT. Thus, the $N_{\sigma(i,j)}$ accept all complete languages and only complete languages. This is achieved as follows: on input x $N_{\sigma(i,j)}$ searches for $|x|^3$ steps deterministically for a counter example (on shorter strings) that D_j reduces SAT to $L\ (N_i)$. If no counter example is found $N_{\sigma(i,j)}$ accepts what N_i accepts, if the counter example is found $N_{\sigma(i,j)}$ accepts x iff x is in SAT. □

We now easily derive the existence of incomplete languages.

Theorem 12 (Ladner): If $P \neq NP$ then there exist incomplete languages (under $\leq_m^p$ and $\leq_T^p$-reductions) in

$$NP - P.$$

Proof: Since

$$\{N_i \mid L\ (N_i) \in NP - P\}$$

is Π_2-complete, $NP - P$ has no r.e. list of total names. One the other hand, the set of complete languages in NP has an r.e. list of NP-names and therefore we are forced to conclude that $NP - P$ contains languages which are not complete. □

Another proof of this result can be obtained from the Corollary 5 that for every sound, axiomatizable system, if $P \neq NP$, there exist sets in $NP - P$ which cannot be shown in F not to be in P for any N_i representation of them. To see this, choose a proof system F in which we have $P \neq NP$ as an axiom and in which we can prove for all $N_{\sigma(i,j)}$ (from Fact 11) that $L(N_{\sigma(i,j)})$ is NP complete and therefore not in P. Then the sets in $NP - P$, not provably in P, must be incomplete.

Conclusion

The classic concepts from recursive function theory have found a very fruitful reinterpretation in complexity theory. In this translation, recursiveness is replaced by polynomial time computability, recursively enumerable sets by NP sets and the analogue to the Kleene Hierarchy is the Polynomial Hierarchy. Though there are some haunting similarities between the

results in recursive function theory and computational complexity, there are many differenc In general, the structure of the feasible computations appears to be far more complicated than the corresponding structure of the effective computations and far from fully understood at this time.

We know that we are facing some very hard problems in our quest to understand the structure of the feasible computations. At the same time, we are convinced that we will have fully repaid our intellectual debt to recursive function theory when we will have understood *what is and is not feasible computable* as well as it is understood *what is and is not effectively computable.*

We should go all out to repay this intellectual debt as soon as possible.

References

[Fr] Friedberg, R.M. "Two Recursively Enumerable Sets of Incomparable Degrees of Unsolvability", *Proceedings of the National Academy of Sciences*, 43:236-238, 1957.

[Go1] Goedel, K. "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme", *Monatsheft fur Mathematik und Physik*, 38:173-198, 1931.

[Go2] Goedel, K. "Über die Länge der Beweise", *Ergebn. eines Math. Kolloquiums* 7:23-24, 1936.

[Ha1] Hartmanis, J. "Independence Results about Context-Free Languages and Lower Bounds", *Information Processing Letters 20* (1985), 241-248.

[Ha2] Hartmanis, J. "On Goedel Speed-up and Succinctness of Language Representations", *Theoretical Computer Science 26* (1983), 335-342.

[Ha3] Hartmanis, J. "On the Succinctness of Different Representations of Languages", *SIAM J. Computing 9* (1980), 114-120.

[HU] Hopcroft, J. and J. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, MA, 1979.

[KOR] Kurtz, K. S., M. O'Donnell and S. Royer. "How to Prove Representation-Independent Independence Results". *Information Processing Letters*, Vol. 24, pp. 5-10, 1987.

[La] Ladner, R.E. "On the Structure of Polynomial Time Reducibility", *J. ACM* 22:155-171, 1975.

[LLR] Landweber, L.H., R.J. Lipton, and E.L. Robertson, "On the structure of sets in NP and other complexity classes", *Theoretical Computer Science 15*, pp 181-200, 1981.

[Mu] Muchnik, A. "On the Unsolvability of the Problem of Reducibility in the Theory of Algorithms", *Doklady Akademii Nauk SSSR* (Russian), 108:194-197, 1956.

[Po] Post, E. "Recursively Enumerable Sets of Positive Integers and Their Decision Problems", *Bulletin AMS* 50:284-316, 1944.

[Re1] Regan, W.R. "On the Separation of Complexity Classes", Doctoral Thesis, Oxford University, September 1986.

[Re2] Regan, W.R. "Index Sets and Presentations of Complexity Classes", Mathematical Sciences Institute Technical Report #88-115, Cornell University, Ithaca, NY, 1988.

[Re3] Regan, W.R. "The Topology of Provability in Complexity Theory", *J. Comp. System Sci.*, 36:384-432, 1988.

[SS] Schmidt, E.H. and T.G. Szymanski. "Succinctness of Descriptions of Unambiguous Context-Free Language", *SIAM Journal on Computing* 6:547-553, 1977.

[Va] Valiant, L.G. "A Note on the Succinctness of Description of Deterministic Languages", *Information and Control* 32:139-145, 1976.