Uniformly Hard Languages*

Rod Downey†
Department of Mathematics
Victoria University
P. O. Box 600, Wellington
New Zealand

Lance Fortnow‡
Department of Computer Science
University of Chicago
1100 E. 58th St.
Chicago, IL 60637
U. S. A.

February 16, 1998

Abstract

We examine the class of "uniformly hard languages," where a language is just not merely infinitely often not computable in polynomial-time but in fact the languages are hard nearly everywhere. We conjecture that all NP-complete languages have this property.

We examine some properties of uniformly hardness. In particular we examine Ladner's proof of the existence of incomplete NP sets if P $\neq$ NP. His proof produces sets that are not even close to uniformly hard. We give evidence that any Ladner-type proof will not give us incomplete uniformly hard sets.

*To appear, *Complexity Theorey, 1998*

†Email: downey@math.vuw.ac.nz.

Partially supported by New Zealand Marsden Fund for basic science via grant 95-VIC-MIS-0698 under contract VIC-509.

‡Email: fortnow@cs.uchicago.edu. Web: http://www.cs.uchicago.edu/~fortnow.

Partially Supported in part by NSF grant CCR 92-53582.

Currently on leave at CWI, AA1, P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands.

1 Introduction

Complexity theorists generally believe that P $\neq$ NP and thus there exist sets in NP like SAT that are not computable in polynomial-time. This leaves open the possibility, however, that for many input lengths instances of SAT have polynomial-time computable satisfying assignments. We suspect most complexity theorists would also believe that SAT does not observe this kind of behavior either.

We call a language A *uniformly hard* if for every B in P there are not arbitrarily large polynomial input ranges where A and B agree.

Conjecture 1.1 *There exists uniformly hard languages in NP.*

We show that if a uniformly hard set A honestly reduces to another set B than B is also uniformly hard. Thus Conjecture 1.1 is equivalent to any of the traditional NP-complete sets, like SAT, being themselves uniformly hard.

Uniformly hardness has an important role in the area of "hardness versus randomness" [20, 5, 14]. One way to state the recent result of Impagliazzo and Wigderson [14] is

Theorem 1.2 (Impagliazzo-Wigderson) *If some language in E cannot be accepted by any $2^{o(n)}$-size circuit family then every language in BPP is not uniformly hard.*

Ladner [16] shows that if P $\neq$ NP then there exist NP sets not in P and not NP-complete. Ladner's proof works by creating a set B which alternates between looking like A and looking like some set in P such as $\emptyset$. These alternations may take a very long time and thus cause B to look like an easy set for very long sequences of input lengths.

The set B created by Ladner's construction is not even close to uniformly hard. This leads to the following conjecture.

Conjecture 1.3 *If there exist uniformly hard sets in NP then there exists incomplete uniformly hard sets.*

To examine Conjecture 1.3 we consider the more general result that follows from Ladner's proof.

Theorem 1.4 ([16]) *For every recursive set A not in P, there exists a set B such that*

1. *B is not in P*
2. *B polynomial-time honest many-one reduces to A*
3. *A does not polynomial-time honest many-one reduce to B.*

We show that under a hard to disprove assumption there exists minimal recursive uniformly hard sets under polynomial-time honest many-one reductions. This shows that Ladner's proof techniques must produce sets with long sequences of easy instances. Finding nonuniform incomplete sets will require new techniques.

1.1 An Historical Perspective

The realization that virtually all combinatorial reductions arising from concrete problems were honest (such as those of Karp [15] and Garey and Johnson [11]) came very early, and strong forms of this observation gave rise to conjectures such as the Isomorphism Conjecture of Berman and Hartmanis [8].

Examining the structure of (the degrees of) computable sets under various polynomial-time reductions has a long history. The fundamental paper here is Ladner [16], who demonstrated considerable richness in the structures, showing that, for instance for honest and standard polynomial time $m-, tt-$, and $T-$ reducibilities, there were minimal pairs and that the structure was dense. In that paper Ladner introduced what has proven to be the fundamental technique of "delayed diagonalization" which in many instances is applied by making a language essentially trivial "long enough" to cause some desired diagonalization. For instance, in the density theorem, as we observed above, Ladner's proof works by creating a set B which alternates between looking like A and looking like some set in P such as $\emptyset$. These alternations may take a very long time and thus cause B to look like an easy set for very long sequences of input lengths. This technique is ubiquitous in the area. For instance, see Balcázar et. al. [7], Balcázar and Diaz [6], Landweber et. al. [18], Chew and Machtey [9], Homer [12], Regan [21], Schöning [22], and Ambos-Spies [1, 3]. It is even the case that most applications of the so-called "speedup technique" have at their heart a Ladner-type strategy. We refer to, for instance, Downey [10], Shinoda-Slaman [23], Ambos-Spies [2], or Shore-Slaman [24].

One of the key motivations for the study of these structures is to give insight into the natures of computation, nondeterminism, and feasibility. From this point of view much of this work can have a somewhat unsatisfactory flavor. Wolfgang Maass coined the phrase "punching large holes in sets." In terms of concrete complexity of natural problems, it seems unsatisfactory to say that if $P \neq NP$ then there are problems of intermediate degree because we punch large holes in the language simply because we can exploit the guaranteed totality of the reductions. (The point is that precisely the same argument works in the case of the classical truth table degrees above the recursion-theoretical degree of the halting problem.) Often one feels that one is doing elementary set theory rather than complexity-theory. In this light the first motivation of the present paper is to ask what the universe looks like is we only restrict our attention to languages where we won't punch large holes in the languages.

A second related motivation comes from the point of view that the current evidence suggests that not only does $P \neq NP$, but *hard instances occur fairly often.* We have tried to capture this idea in the notion of *uniform hardness.* Here we ask that not only is the language not feasibly computable but it has no long easy intervals. From the point of view that SAT is like this, it is very natural to study such languages.

Our final motivation is to try to relate complexity-theoretical hypotheses to structural ones for the appropriate reducibilities. Homer [13] has one example for the noncomputable languages under honest polynomial time reductions, but his techniques do not apply to the computable languages. In our setting we get a nice consequence of the hypothesis that $P = PSPACE$. It seems that getting the consequence to fail is related not to punching holes in languages, but to the existence of certain types of one-way functions, although we do not explore this here.

2 Uniformly Hard Sets

We define "uniformly hard sets" to capture the notion of sets that do not have large gaps where they may be easily computable.

Definition 2.1 Let $m, n \in \mathbb{N}$. For $A \subseteq \Sigma^*$, we let $A \upharpoonright [m, n]$ denote

$$\{x \in A : m \leq |x| \leq n\}.$$

If $n = m$ we write $A \upharpoonright [n]$.

Definition 2.2 We say that $A \subseteq \Sigma^*$ is *uniformly hard* iff for any language $B \in \mathcal{P}$, there is a $k \in \mathbb{N}$ such that for all $m \geq 2$,

$$A \upharpoonright [m, m^k] \neq B \upharpoonright [m, m^k].$$

Definition 2.3 A set A is *polynomial-time many-one honest reducible* to B if there exists a polynomial-time computable function f and a polynomial p such that

1. $f : \Sigma^* \rightarrow \Sigma^* \cup \{T, F\}$
2. If $f(x) \in \Sigma^*$ then $p(|f(x)|) \geq |x|$ (honesty)
3. $x \in A$ if and only if $f(x) \in B \cup \{T\}$.

We require honesty so we can find inverses to f using our P = PSPACE assumption. We require T and F because in Ladner's proof one gets long stretches of nothingness and we would like an honest reduction to have something to map a positive instance to.

Uniformly hard sets are upward closed under these reductions.

Lemma 2.1 *If A is uniformly hard and polynomial-time many-one honest reducible to B then B is also uniformly hard.*

The polynomial-time many-one honest reductions give a partial order to the uniformly hard sets. A "polynomial-time many-one honest degree" is an equivalence class under this ordering.

One might expect to prove that if SAT is not uniformly hard we can derive some interesting collapse. But the possibility remains that at those inputs where SAT is hard, it could be very hard. This possibility is exactly captured by generic oracles.

Theorem 2.2 *Relative to any generic oracle, SAT and every other language in PSPACE is not uniformly hard yet the polynomial-time hierarchy is infinite.*

3 Main Result

Theorem 3.1 *If $P = PSPACE$ then there is a minimal uniformly hard polynomial-time many-one honest degree of a computable set.*

Proof. We must construct a uniformly hard set A so that for all $B <_m^h A$, B is not uniformly hard. To achieve this goal, we use an infinitary priority argument and satisfy the requirements below.

Let W_e denote the e-th P-time language so that "$x \in W_e$" is computable in time $|x|^e$.

In the following the variables n, m, k, s range over $\mathbb{N}$ and z, x, y over Σ^*. To make A uniformly hard we meet the requirements.

$$\mathcal{R}_e : (\forall^\infty\ n)[A \restriction [n] \neq W_e \restriction [n]].$$

Let φ_e denote the e-th polynomial time honest m-reduction, so that for all x, $\varphi_e(x)$ is computable in time $|x|^e$, and $(|\varphi_e(x)|)^e > |x|$. To make A have minimal degree we must meet the requirements below.

$$\mathcal{M}_e : (A \leq_m^h \varphi_e^{-1}(A)) \vee (\varphi_e^{-1}(A) \text{ is not uniformly hard}).$$

Initially we make $A = \Sigma^*$ and every string "active."

Meeting $\mathcal{R}_e$ in isolation. Meeting the $\mathcal{R}_e$ requirements is not too difficult. In isolation for each length $n > 2^e$, we pick an active x of length n and make $W_e(x) \neq A(x)$.

Meeting $\mathcal{M}_e$ in isolation. The construction will proceed in stages $s \in \mathbb{N}$. We will more or less at stage s decide the fate of z with $|z| = s$. Associated with $\mathcal{M}_e$ will be a *current parameter* $k = k(e, s)$. We guarantee that $k(e, s) \geq k(e, s+1)$ for all e and s. It might be that $k(e, s) \to \infty$ or it might come to a limit. The value $k(e, s)$ goes to ∞ indicates that $\varphi_e^{-1}(A)$ is not uniformly hard.

Suppose that we have just reset k and now have a particular value in mind. At some stage we will begin to look at $\mathcal{M}_e$ again. Suppose that s_0 is such a stage. We choose s_0 large enough such that $s_0^k < 2^{\sqrt{s_0}}$ and $s_0 > 2^e$.

We look at those strings x with $s_0 \leq |x| < s_0^k$. Let t_m be the number of active strings of length m and $t = \min_{s_0 \leq m < s_0^k} t_m$. At the end of the stage we will guarantee that there are at least $w = t/(s_0^k + 1)$ active strings remaining at each of these lengths.

We see if there is any length n with $s_0 \leq n < s_0^k$ such that

- there exist at least $t/(1 + (1/s_0^k))$ active strings x such that there exist z with $|\varphi(z)| = n$.

There are two cases depending on whether the answer to • is *yes* or *no*.

Case 1. *The answer to • is yes*

For each $m \neq n$ with $s_0 \leq m < s_0^k$, delete all but the lexicographically least w many strings and make these deleted strings inactive.

Now in order of $m \neq n$, and in lexicographic order of $x \in B(m, s)$, define the *e-associate* $a_e(x)$ to be lexicographically least active string of length n. Then make $a_e(x)$ inactive. That leaves at least $2w$ strings of length n. Associate in the same way the last w strings of length n with the first w and make the first w inactive. Delete and make inactive all remaining strings of length n.

Our promise is the following. For every active x,

$$x \in A \text{ iff } a_e(x) \in A.$$

Now we reset s_1 (the new s_0) to be s_0^k, and at stage s_1 repeat the above with s_1 in place of s_0. If the answer remains *yes* then we use $s_2 = s_1^k$, etc.

Outcome k. Suppose then that the answer is *always yes*. In that case, we claim that $A \leq_m^h \varphi_e^{-1}(A)$. To see this, to decide if $z \in A$ from the $\varphi_e^{-1}(A)$ oracle, if $|z| < s_0$ then use a table lookup. Otherwise, compute i such that $s_i \leq |z| \leq s_i^k$. Using the assumption P = PSPACE, we can determine the active strings and their associates. Now if z is not active or an associate then $z \notin A$. If z is an associate then $z \in A$ iff $\varphi_e^{-1}(z) \in \varphi_e^{-1}(A)$, where $\varphi_e^{-1}(z)$ is any element with $\varphi_e(\varphi_e^{-1}(z)) = z$ found using P = NP. If z is active then $z \in A$ iff $\varphi_e^{-1}(a(z)) \in \varphi_e^{-1}(A)$. We call this process the outcome k.

Case 2. *The answer to • is no.* In this case our action is to delete all active strings z with $s_0 \leq |z| \leq s_0^k$ such that $\exists x[\varphi_e(x) = z]$. This leaves at least w active strings at each length.

We reset $k(e, s+1) > k(e, s)$. We will later attend $\mathcal{M}_e$, again repeating these steps with a suitably longer s_0.

The outcome ∞. Suppose that this *no* case repeats itself infinitely often. The we claim that $\varphi_e^{-1}(A)$ is not uniformly hard. Let k be given. Compute a k_i and s_i sufficiently large that

$$(s_i + 1)^{eke} < s_i^{k_i},$$

and for which the outcome is *no*. Consider the interval of lengths $[(s_i + 1)^e, (s_i + 1)^{ek}]$. We claim that

$$\varphi_e^{-1}(\emptyset) \upharpoonright [(s_i + 1)^e, (s_i + 1)^{ek}] = \varphi_e^{-1}(A) \upharpoonright [(s_i + 1)^e, (s_i + 1)^{ek}].$$

The point is that by the $|x|^e$-honesty of φ_e, it can only be that

$$\varphi_e^{-1}([(s_i+1)^e,(s_i+1)^{ek}]) \subseteq \Sigma^* \upharpoonright [s_i,(s_i+1)^{eke}] \subseteq \Sigma^*[s_i,s_i^{k_i}].$$

But since we make sure that there are no z in $[s_i,s_i^{k_i}]$ with z in the range of φ_e, we get the claim.

Therefore, for each k there is a m with $\varphi_e^{-1}(A) \upharpoonright [m,m^k] = \varphi_e^{-1}(\emptyset) \upharpoonright [m,m^k]$. Hence, $\varphi_e^{-1}(A)$ is not uniformly hard.

In summary In summary either we get the outcome k for some $k \in \mathbb{N}$, in which case $A \leq_m^h \varphi_e^{-1}(A)$, or $k(e,s) \to \infty$, the so-called Π_2 outcome which witnesses the fact that $\varphi_e^{-1}(A)$ is not uniformly hard.

Combining Requirements.

Basically there is no problem with any number of $\mathcal{R}_j$ type requirements combining with a single $\mathcal{M}_e$. This is because we retain at least at $2^{-\sqrt{m}}$ fraction of active strings every time length m is processed. Length m is processed for at most $2\log m$ requirements.

The problems become more subtle when we consider more than one $\mathcal{M}_e$ type requirements, say $\mathcal{M}_e$ and $\mathcal{M}_f$ with $e<f$. (Two requirements are representative, and the inductive strategies continue in the obvious way.) First we will always ensure compatibility of the $k(e,s)$ and $k(f,s)$ by ensuring that at each stage s they are towers of 2, and hence either $k(e,s)$ is a root of $k(f,s)$ or vice versa. We will ask that they be distinct. We make the initial value of $k(i,0)$ a tower of 2's of height $2i+1$.

Now what turns out to be important is the relative sizes of $k_e = k(e,s)$ and $k_f = k(f,s)$ rather than their priorities, since we will use a dynamic priority ordering, in a sense to be described below. To begin with suppose that we have $k_e < k_f$. Then we will have $k_f = k_e^d$ for some d. Both $\mathcal{M}_e$ and $\mathcal{M}_f$ will deal with the same s_0 in the sense of the isolated strategy. But, assuming that $\mathcal{M}_e$ has the k_e outcome each time, the f-interval $[s_0,s_0^{k_f}]$ will consist of d e-intervals of the form

$$[s_0,s_0^{k_e}],[s_1=s_0^{k_e},s_1^{k_e}],...,[s_d,s_d^{k_e}].$$

Call these intervals $I_1,...,I_d$.

Now we regard the output of the basic e-module applied to I_t as producing equivalence classes of strings with two strings, one active and one its associate. Consider the active string the representative of this class.

In future applications of the basic modules for the $\mathcal{R}$ and $\mathcal{M}$ requirements, treat any query to an element of an equivalence class as a query to the active representative. If the representative is deleted then so should all of the other elements in the class. This may in turn yield larger equivalence classes with still one active representative.

To complete the description of the construction, we remark that if the answer to • is no, we will reset $k(e,s)$ (or $k(e,s)$ as the case may be) to be the next largest unused tower of 2. This tower of 2 determines its new priority which means when it gets processed. We will not try to process this requirement again until length $s_0^{k_f}$. □

4 Future Directions

We show that Ladner's nonminimality result appears to require creating large gaps of easiness. There are several directions one can take this research including

- Weaken or eliminate the P = PSPACE assumption.
- Find a complexity theoretical hypothesis (such as a strong one way hypothesis) for which there are no languages minimal in our sense.
- Look at density issues among the uniformly hard sets. First one needs a good definition as to when one set is "uniformly harder" than another.
- Look at decidability in the uniformly hard degrees.

References

[1] K. Ambos-Spies, "On the structure of polynomial time degrees" in STACS 84, Lecture Notes in Computer Science, Vol. 166 (1984) 198-208, Springer-Verlag.

[2] K. Ambos-Spies, "On the structure of the polynomial time degrees of recursive sets," Technical report 206 (1885), Abteilung Informatik, Universität Dortmund.

[3] K. Ambos-Spies, "Minimal pairs for polynomial time reducibilities," in *Computation and Proof Theory*, Lecture Notes in Computer Science, Vol. 270 (1987) 1-13.

[4] K. Ambos-Spies and A. Nies, "The Theory of The Polynomial Time Many-One Degrees is Undecidable," to appear.

[5] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[6] J. Balcázar and J. Diaz, "A note on a theorem of Ladner," Information Processing Letters, Vol. 15 (1982), 84-86.

[7] J. Balcázar, J. Diaz, and J. Gabarro,*Structural Complexity* Vol. 1 and 2 (1987,1989), Springer.

[8] L. Berman and J. Hartmanis, *On isomorphism and density of NP and other complete sets*, SIAM Journal on Computing, Vol. 1 (1977), 305-322.

[9] P. Chew and M. Machtey, "A note on structure and looking back applied to the relative complexity of computable sets," *J. Comput. Sys. Sci.*, Vol. 22 (1981), 53-59.

[10] R. Downey, "Nondiamond theorems for polynomial time reducibility," *J. Comput. Sys. Sci.,* Vol. 45 (1992) 385-395

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, (1979), Freeman.

[12] S. Homer, "Some properties of the lattice of NP sets," in *Workshop on Recursion Theoretical Aspects of Computer Science,* Purdue University, (1981) 18-22.

[13] S. Homer, "Minimal degrees for polynomial reducibilities," *Journal of the ACM*, Vol. 34 (1987), 480-491.

[14] R. Impagliazzo and A. Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th ACM Symposium on the Theory of Computing.* ACM, New York, 1997. To appear.

[15] R. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations,* (R. Miller and J. Thatcher, eds) (1972), 85-104, Plenum Press.

[16] R. Ladner, "On the structure of polynomial-time reducibility" *J. Assoc. Comput. Mach.,* Vol. 22 (1975), 155-171.

[17] R. Ladner, N. Lynch and A. Selman, "A comparison of polynomial time reducibilities," *Theoretical Computer Science*, Vol. 1 (1975), 103-123.

[18] L. Landweber, R. Lipton and E. Robertson, "On the structure of NP and other complexity classes," *Theoretical Computer Science*, Vol. 15 (1981), 181-200.

[19] K. Melhorn, "Polynomial and Abstract Subrecursive Classes," *J. Comput. Sys. Sci.* 12 (1976) 147-178.

[20] N. Nisan and A. Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

[21] K. Regan, "On diagonalization methods and the structure of language classes," in *Fundamentals of Computation Theory,* (M. Karpinski, ed.) Springer Verlag Lecture Notes in Computer Science, Vol. 158 (1983), 368-380.

[22] U. Schöning, "A uniform approach to obtain diagonal sets in complexity classes," *Theoretical Computer Science,* Vol. 18 (1982), 95-103.

[23] J. Shinoda and T. Slaman, "On the Theory of $PTIME$ Degrees of Recursive Sets, *J. Comput. Sys. Sci.* Vol. 41 (1990), 321-366.

[24] R. Shore and T. Slaman, "The P-T-Degrees of Recursive Sets; Lattice Embeddings, Extensions of Embeddings, and the Two Quantifier Theory," in *Proc. Structures in Complexity* (1989).