The Power of Adaptiveness and Additional Queries in Random-Self-Reductions

Joan Feigenbaum*
Lance Fortnow†
Carsten Lund‡
Daniel Spielman§

*AT&T Bell Laboratories, Room 2C-473, 600 Mountain Avenue, Murray Hill, NJ 07974-0636. Email: jf@research.att.com

†University of Chicago, Computer Science Department, 1100 East 58th Street, Chicago, IL 60637. Supported in part by NSF Grant CCR-9009936. Email: fortnow@cs.uchicago.edu

‡AT&T Bell Laboratories, Room 2C-324, 600 Mountain Avenue, Murray Hill, NJ 07974-0636. Email: lund@research.att.com

§Yale University, Computer Science Department, P. O. Box 2158, Yale Station, New Haven, CT 06520. Work done as a summer intern at AT&T Bell Laboratories. Email: spielman-dan@cs.yale.edu

Abstract

We look at relationships between adaptive and nonadaptive random-self-reductions. We also look at what happens to random-self-reductions if we restrict the number of queries they are allowed to make. We show the following results:

- There exist sets that are adaptively random-self-reducible but not nonadaptively random-self-reducible. Under reasonable assumptions, there exist such sets in *NP*.
- There exists a function that has a nonadaptive $(k+1)$-random-self-reduction but does not have an adaptive k-random-self-reduction.
- For *any* countable class of functions $\mathcal{C}$ and *any* unbounded function $k(n)$, there exists a function that is nonadaptively k-uniformly-random self-reducible but is not in $\mathcal{C}/poly$.

These results were presented in preliminary form at the 7th IEEE Structure in Complexity Theory Conference, Boston MA, June 1992.

1 Introduction

Informally, a function f is *random-self-reducible* if the evaluation of f at any given instance x can be reduced in polynomial time to the evaluation of f at one or more *random* instances y_i.

Random-self-reducible functions have many applications, including:

Average-case complexity: A random-self-reduction maps an arbitrary, worst-case instance x in the domain of f to a set of random instances $y_1, \ldots, y_k$ in such a way that $f(x)$ can be computed in polynomial-time, given x, $f(y_1), \ldots, f(y_k)$, and the coin-toss sequence used in the mapping. Thus the average-case complexity of f, where the average is taken with respect to the induced distribution on instances y_i, is the same, up to polynomial factors, as the worst-case randomized complexity of f. An important special case is that in which each random instance y_i is uniformly distributed over all elements in $Dom(f)$ that have length $|x|$. In this case, f "is as hard on average as it is in the worst case." For example, it follows from a result in [Lip] that the PERM (permanent of integer matrices) function is random-self-reducible. The PERM function is also #P-complete [Val]; thus, if PERM could be computed efficiently *on average* (with respect to the target distribution of the reduction), then *every* function in #P could, with a randomized algorithm, be computed efficiently in the *worst case*. Furthermore, the random-self-reduction for PERM is very simple, whereas standard average-case hardness proofs are often complicated.

Lower bounds: The random-self-reducibility of the parity function is used in [Bab] to obtain a simple proof that a random oracle separates the polynomial hierarchy from PSPACE. (An earlier proof of this result in [Cai] does not use random-self-reducibility.)

Interactive proof systems and program checkers: Random-self-reductions are crucial ingredients in many of the original examples of interactive proof systems and program checkers [BK, GMR]. Intuitively, this is because the verifier/checker interrogates the prover/program by comparing its output on the specific input of interest to its outputs on other correlated random instances. Several variations of this relationship between random-self-reducibility and proof systems/checkers are stated formally in [BLR, LFKN, TW]. These ideas play a crucial role in the characterization of the language-recognition power of interactive proof systems [BFL, LFKN, Sha]. Currently, one of the most important open questions about checkability is whether *NP*-complete sets are checkable. Feigenbaum and Fortnow [FF] use results about random-self-reducibility to show that if *NP*-complete sets are checkable, their checkers must use radically different techniques from those used by existing checkers.

Cryptographic protocols: The fact that certain number-theoretic functions are random-self-reducible (and hence hard on average if they are hard at all) is used extensively in the theory of cryptography – e.g., to achieve *probabilistic encryption* [GM] and *cryptographically strong pseudorandom number generation* [BM]. Random-self-reductions also provide natural examples of *instance-hiding schemes* [AFK, BF, BFKR], in which a weak, private computing device uses the resources of a powerful, shared computing device without revealing its private data.

Although random-self-reducibility had been used for a long time in the design and analysis of cryptographic protocols (e.g., [GM, BM]), it was first defined formally and studied from a complexity theoretic point of view by Abadi, Feigenbaum, and Kilian [AFK]; they considered reductions that map the given instance x to *one* random instance y. It is a corollary of the main result in [AFK] that no *NP*-hard function is random-self-reducible in this sense, unless the polynomial-time hierarchy collapses at the third level.

Random-self-reductions that produce several, correlated random instances y_1, ..., y_k were defined formally by Feigenbaum, Kannan, and Nisan [FKN]; however, they only considered reductions that produce y_i's that are uniformly distributed over $\{0,1\}^{|x|}$. Their main result is that self-reductions that map x to two instances y_1 and y_2, each of which is uniformly distributed over $\{0,1\}^{|x|}$, do not exist for *NP*-hard functions, unless the polynomial-time hierarchy collapses at the third level.

Feigenbaum and Fortnow [FF] continue the study of random-self-reductions from a complexity-theoretic point of view. They further generalize the formal definition of random-self-reducibility. (It is the definition in [FF] that we study here.) Specifically, they look at reductions that map a given instance x to a sequence of random instances y_1, ..., y_k, with the property that the induced distribution on each y_i depends only on the length of x. They consider both *nonadaptive* k*-random-self-reductions*, in which the k random instances are produced in one pass, and *adaptive* k*-random-self-reductions*, in which the instance y_i may depend not only on x and the coin-toss sequence used in the reduction, but on $f(y_1)$, …, $f(y_{i-1})$ as well. They show that sets that are complete for any level of the polynomial hierarchy are not nonadaptively random-self-reducible, unless the hierarchy collapses. In particular, *NP*-complete sets are not nonadaptively random-self-reducible, unless the hierarchy collapses at the third level. By contrast, they show that sets complete for the classes PP and MOD_mP are random-self-reducible.

In this paper, we analyze the relationships between adaptive and nonadaptive random-self-reducibility and the effect of changing the number of queries available to the random-self-reduction. Our main results are:

- There exists a language that is adaptively random-self-reducible but not nonadaptively random-self-reducible. We can make such a language sparse and recognizable in slightly more than polynomial space.
- If Probabilistic Triple-Exponential Time does not contain Nondeterministic Triple-Exponential time, then there exists a set in *NP* that is adaptively random-self-reducible but not nonadaptively random-self-reducible.
- For any polynomially bounded $k(n)$, there exists a function that is nonadaptively $(k(n)+1)$-random-self-reducible but not adaptively k-random-self-reducible.
- For *any* countable class of functions $\mathcal{C}$ and *any* unbounded function $k(n)$, there exists a function that is nonadaptively k-uniformly-random self-reducible but not in $\mathcal{C}$/poly.

For example, there exists a function that is nonadaptively $k(n)$-uniformly-random self-reducible but not in FPSPACE/*poly*, for any unbounded $k(n)$. This should be contrasted with Feigenbaum, Kannan, and Nisan's theorem that all nonadaptively 2-uniformly-random self-reducible sets are in *NP*/*poly* (cf. [FKN]).

These results first appeared in our Technical Memorandum [FFLS].

2 Definitions and Notation

Throughout this paper, x is the input to a randomized reduction, n is the size of x, and r is a random variable distributed uniformly on $\{0,1\}^{w(n)}$, where w is some polynomially bounded function of n. The parameter $k = k(n)$ is also a polynomially bounded function of n.

Definition 2.1 *A* nonadaptive $k(n)$-random-self-reduction *(abbreviated "k-rsr") for f is a pair of polynomial-time computable functions σ and ϕ with the following properties.*

1. *For all n and all $x \in \{0,1\}^n$,*

$$f(x) = \phi(x, r, f(\sigma(1, x, r)), \ldots, f(\sigma(k, x, r))),$$

with probability at least 2/3, and

2. *For all x_1 and x_2 such that $|x_1| = |x_2|$ and all $i \in \{1, \ldots, k\}$, $\sigma(i, x_1, r)$ and $\sigma(i, x_2, r)$ are identically distributed.*

Note that the random queries $\sigma(1, x, r)$, …, $\sigma(k, x, r)$ are in general dependent.

Observation 2.2 (Szegedy – see [FF]): *In a nonadaptive rsr, we can assume without loss of generality that all of the random variables $\sigma(1, x, r)$, …, $\sigma(k, x, r)$ are identically distributed.*

The word *nonadaptive* in Definition 2.1 refers to the fact that all of the random queries are produced in one round; that is, $\sigma(i, x, r)$ does not depend on $f(\sigma(j, x, r))$ for $j \neq i$. The next definition covers the case in which queries are produced in multiple rounds – that is, the rsr follows an adaptive strategy.

Definition 2.3 *An* adaptive $k(n)$-random-self-reduction *for f is a polynomial-time oracle Turing machine M with the following properties.*

1. *For all n and all $x \in \{0,1\}^n$,*

$$f(x) = M^f(x, r),$$

with probability at least 2/3.

2. *For all x and all r, $M^f(x, r)$ asks at most $k(n)$ queries $(q_1^f(x, r), \ldots, q_k^f(x, r))$ to f.*

3. For all x_1 and x_2 such that $|x_1| = |x_2|$ and all $i \in \{1, \ldots, k\}$, $q_i^f(x_1, r)$ and $q_i^f(x_2, r)$ are identically distributed.

For both adaptive and nonadaptive rsr's we can replace 2/3 by $1 - 2^{p(n)}$ for any polynomial $p(n)$ at the cost of increasing $k(n)$ by a polynomial factor. We can achieve this bound by using the standard technique of running several rsr's and taking a plurality vote. See [FF] for details.

Property 2 in Definition 2.1 and property 3 in Definition 2.3 are referred to as *the instance-hiding property.* This means that any one query of the rsr reveals nothing about the instance x, except its size. It should be noted that the instance x might be computable from a set of two queries. Note that the instance-hiding property need not hold for a "cheating" oracle $f' \neq f$, i.e., an adaptive reduction could, in round i, "leak" something besides the size of the instance if it is given wrong answers to one or more of its queries in rounds 1 through $i - 1$.

An rsr is *oblivious* if its queries only depend on the input size – that is, it is not just that the distributions of queries are the same for all inputs of the same size (which is true of any rsr), but moreover that the reduction ignores the input until after all queries have been made. An rsr is *uniform* if, for all inputs x, each query is uniformly distributed over $\{0,1\}^{|x|}$. We call such a reduction a k-uniformly-random self-reduction (abbreviated k-ursr) and stress that the word "uniform" describes the distribution of random queries, i.e., it is not meant to distinguish between reduction procedures that take advice and those that don't. Uniformly-random self-reductions are the type of rsr's studied in [FKN]. We say that a function f is random-self-reducible if it has an rsr. A language L is random-self-reducible if the indicator function for L is rsr.

Let *BPE* (resp. *BPEE* and *BPEEE*) be the class of languages recognizable in probabilistic exponential time (resp. doubly exponential time and triply exponential time), i.e., the union of $BPTIME(2^{cn})$ (resp. $BPTIME(2^{2^{cn}})$ and $BPTIME(2^{2^{2^{cn}}})$) for all constants c. The nondeterministic time classes *NE*, *NEE*, and *NEEE* are defined analogously.

Let $\beta(n)$ be any unbounded nondecreasing function such that $n^{\beta(n)}$ is time constructible, e.g., $\beta(n) = \log^* n$.

3 Adaptiveness versus Nonadaptiveness

In this section we study the difference between adaptive and nonadaptive rsr's. We find sets of relatively low complexity that are adaptively rsr but not nonadaptively rsr.

Theorem 3.1 *For any $1 \leq k(n) \leq n$, there exists a sparse set L such that L is in DSPACE($n^{\beta(n)}$) and has an oblivious, adaptive k-rsr but for which any nonadaptive $k'(n)$-rsr will, for infinitely many inputs, err with probability at least $1/2 - k'/(2^{k+1}-2)$.*

proof : First we will describe how to construct a non-sparse L. From there, the construction of a sparse L is straightforward. In this proof, there will be a 1-1 correspondence between strings of length n and the integers $1, 2, \ldots, 2^n$, i.e., the i^{th} string in the lexicographic ordering corresponds to i.

A *step-function* $B_{n,m,s}$ is the language

$$B_{n,m,s} := \{x \in \{0,1\}^n | s \leq x\},$$

where $0 \leq s \leq m \leq 2^n$ (i.e., m is a bound on how big s can be). Hence $B_{n,m,s}$ consists of all n-bits strings from the s^{th} such string on. We call s the *step* and m the *range*.

Note that any function on $\{0,1\}^*$ whose restriction to $\{0,1\}^n$ is equal to $B_{n,m,s}$ for some m and s has an oblivious, adaptive ($\lceil \log_2(m+1) \rceil$)-rsr. This is because it takes $\lceil \log_2(m+1) \rceil$ queries to find the step using binary search and, once the step is known, it is simple to determine membership in $B_{n,m,s}$.

The language L will be a union of step-functions, one $B_{n,m,s}$ for each input length n, where the range m will always be equal to $2^{k(n)} - 1$ and the steps will be determined by diagonalization against nonadaptive k'-rsr's.

The diagonalization is done as follows. Let $M_1, M_2, \ldots$ be an enumeration of all the nonadaptive k'-rsr's in which every k'-rsr occurs infinitely many times. Assume without loss of generality that M_i's running time is bounded by n^i.

We diagonalize against M_i on inputs of length n_i, where n_i is the least integer such that $\beta(n_i) > i$. (This choice is made in order to ensure that $n_i^{\beta(n_i)}$ is greater than the running time of M_i on inputs of length n_i.)

Since M_i asks at most k' queries, for some $1 \leq x \leq m$, the probability that one of M_i's k' queries is equal to x is at most k'/m. Look at M_i's behavior on the first such input x with oracles $B_{n_i,m,x}$ and $B_{n_i,m,x+1}$. Let

$$p_1 = Pr[M_i^{B_{n_i,m,x}}(x) = 1]$$

$$p_2 = Pr[M_i^{B_{n_i,m,x+1}}(x) = 1].$$

Then $|p_1 - p_2| \leq k'/m$. On the other hand, for M_i to err with probability at most p for both $B_{n_i,m,x}$ and $B_{n_i,m,x+1}$, it must be the case that $p_1 \geq 1-p$ and $p_2 \leq p$, by definition of the step functions. Hence $k'/m \geq 1-2p$. Let s be the one of x and $x+1$ on which M_i errs with probability at least $p = 1/2 - k'/2m$ on input x.

The above construction can be carried out in deterministic space $O(n^{\beta(n)})$.

In order to make L sparse we delete from L any string that the $k(n)$-rsr does not query during its binary search and denote the resulting language by L'. This will allow the same procedure to find the step of L and then decide membership in L'.

In the proof that we could choose a query x and a step s on which M_i errs, we only needed the fact that the set of functions that we could choose from, $\{B_{n_i,m,0}, \ldots, B_{n_i,m,m}\}$, contains, for every query x, two functions that differ only in their answer for x, e.g., $B_{n_i,m,x}$ and $B_{n_i,m,x+1}$. The languages L' have the same property. The function derived from $B_{n_i,m,x2^y}$, where x is odd, is identical to the function derived from $B_{n_i,m,(x+1)2^y}$ except in its value at $x2^y$. Thus, we can use the sparse functions of L' to diagonalize against M_i's as we did for L. ▮

Corollary 3.2 *There exists a sparse set $L \in DSPACE(n^{\beta(n)})$ such that L is obliviously, adaptively rsr but not nonadaptively rsr.*

proof : Let $k(n) = n$ and L be the language constructed in Theorem 3.1. Then for any polynomial $p(n)$, any nonadaptive $p(n)$-rsr M will err with probability at least $1/2 - \frac{p(n)}{2^{n+1}}$ for infinitely many inputs on L. Hence there exists some input for which M errs with probability greater than $1/3$, which means that M is not an rsr for L. ▮

Corollary 3.3 *For any $1 \leq k(n) \leq n$, there exists a sparse set L such that L is in DSPACE($n^{\beta(n)}$) and has an oblivious, adaptive k-rsr but for which any adaptive $k''(n)$-rsr will, for infinitely many inputs, err with probability at least $\frac{1}{2} - \frac{2^{k''}-1}{2^{k+1}-2}$.*

proof : We can assume that the adaptive k''-rsr M has the instance-hiding property for all the 2^k step-functions. Otherwise we can diagonalize against M by choosing a step-function for which M does not have the instance-hiding property.

We construct a nonadaptive $(2^k k'')$-rsr M' for L as follows: M' first chooses a random string r. For each of the 2^k step functions, M' simulates $M(r)$. The list of queries to be made by M' is the concatenation of the lists of queries of M on each step function. When given the answers to these queries, M' does what M would have done if given these answers. Since M has the instance-hiding property for each step function, M' has the instance-hiding property.

Once r is fixed, the ith query of M is determined by the answers to the previous $i-1$ queries. Thus, while the list of queries made by M' has $2^k k''$ entries, at most $2^{k''} - 1$ of these queries are distinct. The corollary then follows from Theorem 3.1. ▌

We turn next to the question of whether such sets exist in *NP*. First we prove a result about a restricted form of nonadaptive rsr's, i.e., those that are length-preserving. We then show how the proof can be modified to yield a result about general nonadaptive rsr's.

Theorem 3.4 *If $NE - BPE$ is nonempty, there exists a set in NP that has a length-preserving, adaptive rsr but does not have a length-preserving, nonadaptive rsr.*

proof : This construction uses the step-function idea from the proof of Theorem 3.1.

Let L be a set in $NE - BPE$ and L' be $\{0^{1x} | x \in L\}$, where $1x$ denotes the integer whose binary representation is $1x$. Note that L' is a tally set in $NP - BPP$. Because L' is in *NP*, the question "is 0^n in L'" can be reduced to a *SAT* instance φ_n in n^c variables, for some constant c. Let

$$L'' = \bigcup_{n \geq 1} \{y \in \{0,1\}^{n^c} \ : \ \exists \text{ a satisfying assignment } s \text{ for } \varphi_n \text{ such that } s \leq y\}.$$

We will prove that L'' has the properties stated in the theorem. This follows from Claims 1 through 4.

Claim 1 *L'' is in NP.*

Claim 2 *L'' has a length-preserving, adaptive rsr.*

Claim 3 *If L'' has a length-preserving, nonadaptive rsr, then $L'' \in BPP$.*

Claim 4 *If $L'' \in BPP$, then $L \in BPE$.*

proof of claim 1 : On input x of length n, a nondeterministic procedure can compute l such that $n = l^c$. It rejects x if no such l exists. It then computes φ_l, guesses an assignment s, and accepts x if s satisfies φ_l and $s \leq x$. ∎

proof of claim 2 : This follows from the fact that L'' is a sequence of step-functions. ∎

proof of claim 3 : Let M be a length-preserving, nonadaptive $k(n)$-rsr for L''. Assume without loss of generality that, for all inputs x of length n, M fails with probability at most $1/2^n$.

We construct a probabilistic procedure M' that, with high probability on input x, finds the step among the L''-instances $\{0,1\}^n$. Once the step is known, it is trivial to decide membership in L''.

M' first generates m strings by taking m independent samples from M's query-distribution D, where m will be determined later. We call this set of m queries the *test* queries. Note that all test queries have length n, because M is length-preserving.

Because the restriction of L'' to $\{0,1\}^n$ is a step-function, there are only $m+1$ consistent ways in which to answer the test queries. We describe below a procedure M'' that finds the step s in $\{0,1\}^n$ with high probability when given the correct answers to the test queries. For now, note that M' can construct a list $s_1, s_2, \ldots, s_{m+1}$ of possible steps by running M'' on each consistent set of answers to the test queries. M' can then conclude that correct value for the step s is the lexicographically least s_i that satisfies φ_l, where $l^c = n$.

It remains to specify M''. Because it is given the answers to the test queries, M'' knows some interval I in which the step must lie: I is the interval bounded by the lexicographically greatest no-instance in the test set and the lexicographically smallest yes-instance in the test set. M'' uses the rsr M to perform a binary search for the step in the interval I. It gets the right answer with high probability for any particular input it needs in its binary search if all of the random queries M makes for that input lie outside I. Thus we must show that there is an m that is large enough, but still polynomial, to make I small enough so that all of the random queries asked by M on all inputs used in the binary search are outside of I with high probability.

If

$$Pr_{D(y)}[y \in I] < \epsilon,$$

then the probability is at most $k(n)\epsilon$ that at least one of the random queries for any given input falls in I. The binary search procedure needs at most $n+1$ inputs. Hence M'' fails to get a correct answer from the binary search with probability at most

$$(n+1)(k(n)\epsilon + 1/2^n),$$

where the $1/2^n$ term is the probability that M fails. So it remains to choose appropriate values for ϵ and m.

Let s be the step and let I^- be the smallest interval of the form $[s^-, s]$ such that $Pr_{D(y)}[y \in I^-] \geq \epsilon/2$. Similarly, let I^+ be the smallest interval of the form $[s, s^+]$ such that $Pr_{D(y)}[y \in I^+] \geq \epsilon/2$. Let I' be the open interval (s^-, s^+). From the minimality of I^- and I^+, we get that $Pr_{D(y)}[y \in I'] < \epsilon$. Furthermore if one test query belongs to I^- and another belongs to I^+ then $I \subset I'$. With probability at most $(1-\epsilon/2)^m$, none of the m random queries belongs to I^-; hence the probability that $I \not\subset I'$ is at most $2(1-\epsilon/2)^m$. Thus with probability at least $1 - 2(1-\epsilon/2)^m$, we have

$$Pr_{D(y)}[y \in I] < \epsilon.$$

It follows that M'' fails with probability at most

$$2(1-\epsilon/2)^m + (n+1)(k(n)\epsilon + 1/2^n) \leq 1/3,$$

if we choose $\epsilon = 1/(6(n+1)k(n))$ and $m = 5/\epsilon$. ▌

proof of claim 4 : Assume that $L'' \in BPTIME(n^d)$, for some constant d. Then L' belongs to $BPTIME(n^{cd})$. This in turn implies that $L \in BPTIME(2^{dc(n+2)}) \subset BPE$. ▌
▌

A similar but more intricate argument can be used to prove the following more general result.

Theorem 3.5 *If NEEE−BPEEE is nonempty, there exists a set in NP that is adaptively rsr but not nonadaptively rsr.*

sketch of proof : The reason that the construction in Theorem 3.4 does not work for a general nonadaptive rsr M is that M may ask queries of either smaller or larger sizes than the input size n. This property implies that the proof of Claim 3 is no longer valid.

If M only asked smaller sized queries then a simple change in the proof of Claim 3 would be enough to prove the analogous claim for Theorem 3.5. We could modify the M' of the previous proof so that it computed, in a bottomup fashion, the steps for all input sizes less than n.

On the other hand, if M asks queries of size greater than n, there will be more than a polynomial number of ways to answer the test queries. For example, if each test query is of a different size and all sizes are greater than n, there will be 2^m consistent sets of answers.

We solve this problem by making sure that our *NP* language is trivial for most input sizes. We need that any polynomial time rsr M is only able to ask queries of a constant number of sizes on which M' does not already know the answers.

We can assume that M' knows the answers for all queries of size less than n, since M' can use the bottomup approach described above. For a hard input size N, there will only be $a+1$ consistent ways of answering the test queries of size N, where a is the number of queries of size N. Hence if there are only a constant number of nontrivial query sizes larger than n, M' can enumerate all the consistent answers in polynomial time.

The construction of a language with the desired property is similar to the previous construction except that we let L be in $NEEE - BPEEE$. Hence L' is a tally set in $NEE - BPEE$, φ_n has $2^{2^{cn}}$ variables and

$$L'' = \bigcup_{n \geq 1} \{y \in \{0,1\}^{2^{2^{cn}}} \ : \ \exists \text{ a satisfying assignment } s \text{ for } \varphi_n \text{ such that } s \leq y\}.$$

Let M be an rsr that runs in time n^d for some constant d. Hence M can only ask queries of size at most n^d. This implies that there are only a constant number of nontrivial sizes greater than n that are represented among M's queries, i.e., the number of such sizes is at most $\frac{\log d}{c}$.

The rest of the proof of Theorem 3.5 is the same as that of Theorem 3.4 except for small changes in the parameters. ▌

Remark 3.6 The above construction can be modified to yield a sparse set $L''' \in NP$ that is adaptively rsr but not nonadaptively rsr, using the same transformation from nonsparse to sparse set as in the proof of Theorem 3.1.

Furthermore any nonadaptive rsr M for L''' can be transformed into a nonadaptive rsr for L''. To prove this, it is enough to show that L'' is polynomial-time, truth-table equivalent to L'''. Given an input x, consider the elements encountered during a binary search for x. If we know whether each of these elements is in L''', it is easy to determine whether x is in L''. The same is true the other way – in this case, we need only the answers for x and the query that is encountered just before x in the binary search.

4 Number of queries

In this section we investigate how much each additional query increases the power of a random-self-reduction. We find a function that has relatively low complexity for which one additional query makes the difference between being rsr and not being rsr.

Theorem 4.1 *Let $k(n)$ be polynomially bounded. There is a function in $FDSPACE(n^{\beta(n)})$ that is obliviously, nonadaptively $(k+1)$-ursr but not adaptively k-rsr.*

proof : In this proof, we use a 1-1 correspondence between strings of length n and the elements of the finite field $GF(2^n)$. The function f will be a sequence of univariate polynomials over $GF(2^n)$ of degree at most $k(n)$, one for each input size n. It is clear that such a function has an oblivious, nonadaptive $(k+1)$-ursr, because, for each input size, knowing the value of f at $k+1$ points allows the reduction to interpolate the polynomial and determine the value at any other point.

As in the proof of Theorem 3.1, we construct f by diagonalizing against all k-rsr's. So let M_i be the ith k-rsr. We diagonalize against M_i on inputs of length $n_i = \beta^{-1}(i)$.

We need to find a polynomial p of degree k such that, if $f = p$ on inputs of length n_i, then M_i fails on some such input with high probability. Assume that M_i has the instance-hiding property for all polynomials p; if it doesn't, we can diagonalize against M_i by choosing a p for which M_i doesn't have the property (and hence is not an rsr). We show that even for a random p and a random input, M_i fails with high probability. Specifically, we show that

$$Pr_{x,r,p}[M_i^p(x,r) = p(x)] \leq \frac{k+1}{2^n},$$

where r is the random string used by M_i on this run. This follows from the inequalities:

$$\forall x, r : Pr_p[M_i^p(x,r) = p(x) | M_i^p(x,r) \text{ does not query } x] \leq \frac{1}{2^n} \tag{1}$$

and

$$\forall p : Pr_{x,r}[M_i^p(x,r) \text{ does query } x] \leq \frac{k}{2^n}. \tag{2}$$

We prove Inequality (1) by choosing p in the following way. Assume without loss of generality that no two of M_i's queries are equal. For each query, choose a random value for p at that point. After k queries, $M_i(x,r)$'s answer on input x is determined. However, all values for $p(x)$ are still equally likely after k queries – just choose a random element of $GF(2^n)$ as the value of $p(x)$.

To prove Inequality (2), we use the fact that M_i has the instance-hiding property for p. Hence let D_i be the distribution for M_i^p's ith question. That is,

$$\forall x : Pr_r[M_i^p(x,r) \text{ queries } x \text{ in the } i\text{th step}] = D_i(x).$$

Hence

$$\forall x : Pr_r[M_i^p(x,r) \text{ queries } x \text{ in any step}] \leq \sum_{i=1}^{k} D_i(x).$$

Therefore

$$Pr_{x,r}[M_i^p(x,r) \text{ queries } x]$$
$$\leq \frac{1}{2^n} \sum_{x \in GF(2^n)} \sum_{i=1}^{k} D_i(x) = \frac{k}{2^n}. \blacksquare$$

5 Uniform Random-Self-Reductions with an Unbounded Number of Queries

Feigenbaum, Kannan, and Nisan [FKN] showed that, if S has a nonadaptive 2-ursr that never errs, then S is in *NP/poly*. We show by contrast that, for any unbounded $k(n)$, there exist functions f that are nonadaptively k-ursr but not even recursive with polynomial sized advice.

Theorem 5.1 *Let $\mathcal{C}$ be any countable class of functions. If $k(n) = \omega(1)$, then there exists a function f that is nonadaptively k-ursr but not in $\mathcal{C}$/poly.*

proof : The proof is a counting argument. For each input size we construct super-exponentially many functions that all have the same nonadaptive k-ursr, say M. By varying the polynomial advice string available to M over all possibilities, we create only an exponential number of possibilities for the function computed. Thus, the existence of an f with the desired property follows by a standard diagonalization argument.

Lemma 5.2 *Given $n \geq 0$ and $k(n) \leq \log\log n$, let $m = \lceil \log k(n) \rceil$. There exist a family F_n of functions $g : \{0,1\}^n \rightarrow \{0,1\}^m$ and a probabilistic polynomial-time machine M such that M, when restricted to inputs in $\{0,1\}^n$, is a nonadaptive k-ursr for any function in F_n, and $|F_n| > 2^{n^{k(n)-2}}$ for large n.*

proof : Any g in F_n will correspond to an l-variable polynomial p of degree at most $k(n) - 1$ over some finite field $GF(q)$. This type of function is known to have a k-ursr, as long as $k \leq q$ (cf. [BF, Lip]). To prove Lemma 5.2, we need to count the number of such functions and to embed $GF(q)^l$ into $\{0,1\}^n$. The purpose of the embedding is to create a function defined on $\{0,1\}^n$ and to have the rsr produce queries that are distributed uniformly over $\{0,1\}^n$; the natural domain for the polynomials is $GF(q)^l$. The embedding will determine the parameters q and l.

Let $q = 2^m$ and $l = \lfloor n/m \rfloor$. Note that $q \geq k$, which is required by the standard reduction in [BF, Lip]. Write down 2^{n-lm} copies of $GF(q)^l$. There are $q^l = 2^{lm}$ elements in $GF(q)^l$; so there are a total of 2^n elements in this multiset consisting of 2^{n-lm} copies. Assign an n-bit string to each element in the multiset. This assignment is the embedding of $GF(q)^l$ into $\{0,1\}^n$. We now define the rsr. Given an l-variable polynomial p, we evaluate the corresponding function g on $\{0,1\}^n$ as follows: To find $g(x)$, where $x \in \{0,1\}^n$, first find the element $y \in GF(q)^l$ that is mapped to x by the embedding. Evaluate $p(y)$ to get an element z of $GF(q)$; z corresponds to an element of $\{0,1\}^m$ in the natural way used in Theorem 4.1. To perform the rsr of g on input x, first find y and perform the [BF, Lip] ursr of p on input y; this produces uniformly random queries $y_1, \ldots, y_k$. For each y_i, pick a copy of $GF(q)^l$ uniformly at random, take the $x_i \in \{0,1\}^n$ that corresponds to y_i in that copy in the embedding, and evaluate $g(x_i)$. This procedure results in g-queries that are distributed uniformly over $\{0,1\}^n$.

In order to estimate the number of such functions, note first that two different polynomials p, p' define different functions. We use as a lower bound on the number of l-variable polynomials over $GF(q)$ of degree at most $k-1$ the expression q^t, where t is the number of multilinear monomials of degree $k-1$. There are exactly $t = \binom{l}{k-1}$ such monomials.

$$\binom{l}{k-1} > \frac{\left(\left\lfloor \frac{n}{\lceil \log k(n) \rceil} \right\rfloor - k(n)\right)^{k(n)}}{(k(n)-1)!} > n^{k(n)-2},$$

because $k(n) \leq \log\log n$. Hence our lower bound on the number of l-variable polynomials is $q^{n^{k(n)-2}} > 2^{n^{k(n)-2}}$. ▌▌

Theorem 5.1 and Feigenbaum, Kannan, and Nisan's result about 2-ursr's together suggest the following question: Is there a set that is $O(1)$-ursr but not in *NP/poly*?

References

[AFK] M. Abadi, J. Feigenbaum, and J. Kilian. On hiding information from an oracle. *Journal of Computer and System Sciences*, 39:21–50, 1989.

[Bab] L. Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Information Processing Letters*, 26:51–53, 1987.

[BFL] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.

[BF] D. Beaver and J. Feigenbaum. Hiding instances in multioracle queries. In *Proceedings of the 7th Symposium on Theoretical Aspects of Computer Science*, Springer-Verlag Lecture Notes in Computer Science, volume 415, 37–48, 1990.

[BFKR] D. Beaver, J. Feigenbaum, J. Kilian, and P. Rogaway. Security with low communication overhead. In *Proceedings of the 1990 CRYPTO Conference*, Springer-Verlag Lecture Notes in Computer Science, volume 537, 62–76, 1991.

[BK] M. Blum and S. Kannan. Designing programs that check their work. In *Proceedings of the 21st ACM Symposium on the Theory of Computing*, 86–97, 1989.

[BLR] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proceedings of the 22nd ACM Symposium on the Theory of Computing*, 73–83, 1990.

[BM] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13:850–864, 1984.

[Cai] J. Cai. With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy. *Journal of Computer and System Sciences*, 38:68–85, 1989.

[FF] J. Feigenbaum and L. Fortnow. On the random-self-reducibility of complete sets. In *Proceedings of the 6th IEEE Structure in Complexity Theory Conference*, 124–132, 1991. Final version to appear in *SIAM Journal on Computing.*

[FFLS] J. Feigenbaum, L. Fortnow, C. Lund, and D. Spielman. The Power of Adaptiveness and Additional Queries in Random-Self-Reductions (Extended Abstract). *AT&T Bell Laboratories Technical Memorandum*, January, 1992.

[FKN] J. Feigenbaum, S. Kannan, and N. Nisan. Lower bounds on random-self-reducibility. In *Proceedings of the 5th IEEE Structure in Complexity Theory Conference*, 100–109, 1990.

[GM] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.

[GMR] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[LFKN] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, 2–10, 1990. Final version to appear in *Journal of the ACM.*

[Lip] R. Lipton. New directions in testing. In *Distributed Computing and Cryptography*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, volume 2, 191–202, American Mathematical Society, 1991.

[Sha] A. Shamir. IP=PSPACE. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, 11–15, 1990. Final version to appear in *Journal of the ACM.*

[TW] M. Tompa and H. Woll. Random self-reducibility and zero-knowledge interactive proofs of possession of information. In *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science*, 472–482, 1987.

[Val] L. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.