On the Power of Two-Local Random Reductions

Lance Fortnow*
Mario Szegedy†
University of Chicago
Computer Science Department
1100 E. 58th Street
Chicago, IL 60637

Abstract

We show that any language that has a two-locally-random reduction in which the target functions are boolean is in NP/poly∩co-NP/poly. This extends and simplifies a result by Yao.

1 Introduction

Suppose Frank wanted to access a database. Frank had access privileges to this database but for security reasons Frank could not reveal his question to this database. What can Frank learn under this requirement? What if Frank had access to several copies of the same database?

Abadi, Feigenbaum and Kilian [1] looked at the following game based on this scenario: Suppose a probabilistic polynomial-time player has access to a trustworthy oracle. This player wishes to use this oracle to determine the value of some complex function of some input but does not wish to reveal any information about the input besides its length. Abadi, Feigenbaum and Kilian [1] showed that any language reducible to an oracle in this fashion lies in NP/poly∩co-NP/poly.

Beaver and Feigenbaum [2] looked at the power of having the polynomial-time player query several separated oracles, i.e. oracles that can not communicate among themselves or listen to the conversation between a different oracle and the player. Beaver and Feigenbaum show the surprising result that, given $n+1$ different oracles, *any* function has such a *locally-random reduction.* Beaver, Feigenbaum, Kilian and Rogaway [3] improved this result to show that $n/(c\log n)$ oracles suffice for any positive constant c.

Virtually nothing was known about the complexity of two-locally-random reduction. Perhaps one could use two separate and trustworthy oracles to determine the value of any function without revealing more than the input length. Extending an idea of Yao [6], we give a partial negative result: Any language with a two-locally-random reduction with boolean oracles is in NP/poly∩co-NP/poly.

2 The Main Theorem

First we formally define local-random reductions:

*Supported in part by NSF Grant CCR-9009936.
†AT& T Bell Laboratories, 600 Mountain Avenue, P.O. Box 636, Murray Hill, NJ 07974

Definition 1 *A function f has a k-locally-random reduction if there exist polynomial-time functions σ and ϕ and a polynomial $q(n)$ such that for input x and every r of length $q(|x|)$, there exists arbitrary oracle functions $g_1, \ldots, g_{k(n)}$ such that*

$$f(x) = \phi(x, r, g_1(\sigma(1, x, r)), \ldots, g_{k(n)}(\sigma(k(n), x, r)))$$

and for each i, $\sigma(i, x, r)$ and $\sigma(i, y, r)$ are distributed identically when $|x| = |y|$ and r is chosen uniformly at random over all strings of length $q(|x|)$.

We say a language has a k-locally-random reduction if its characteristic function has.

The outputs of the σ functions are the questions asked to the oracles, the r is the random coins of the querier, the g functions are the oracle responses and the ϕ function is the computation done after the response. If k is a constant, we will often use $\sigma_i(x, r)$ for $\sigma(i, x, r)$.

Locally-random reductions are a restriction of instance hiding schemes where the oracles can flip coins and more importantly interact with the polynomial-time player including having future answers depend on previous questions. See Beaver and Feigenbaum [2] for further details.

We can also look at *random-self reductions* as a restriction of local-random reductions by requiring $g_1 = \ldots = g_k = f$. For more precise definitions and theorems about random-self reductions see [1, 5, 4].

We can now state the main theorem:

Theorem 2 *If L has a two-local random reduction with oracles g_1 and g_2 where g_1 and g_2 output a single bit then L is in NP/poly$\cap$co-NP/poly.*

Our proof was inspired by a weaker result by Yao [6]: Any language with a two-local-random reduction with oracles that output only a single bit each is in PSPACE/poly. Besides obtaining stronger consequences our proof is also reasonably simpler than Yao's original proof.

Yap [7] shows that if NP$\subseteq$co-NP/poly then the polynomial-time hierarchy collapses to the third level. From this fact we immediately get

Corollary 3 *If SAT (or any other NP-hard language) has a two-local-random-reduction where the oracles only output single bit responses then the polynomial-time hierarchy collapses to the third level.*

3 Proof of Main Theorem

To prove Theorem 2, we need only show L is in NP/poly because by Definition 1 a language has a k-local-random reduction if and only if its complement also has one.

For every fixed n the characteristic function of the words of L having length n is a boolean function $f_n = f$.

Suppose f has a two-local-random reduction as required by the theorem. Consider the multisets $M_1 = \{\sigma_1(\mathbf{0}, r) \mid r \in \{0,1\}^{q(n)}\}$, $M_2 = \{\sigma_2(\mathbf{0}, r) \mid r \in \{0,1\}^{q(n)}\}$ (where $\mathbf{0}$ denotes the n bit string of zeros). The distributional equivalence of σ_1 and σ_2 imply that for every input x of f the multisets $\{\sigma_1(x, r) \mid r \in \{0,1\}^{q(n)}\}$ and $\{\sigma_2(x, r) \mid r \in \{0,1\}^{q(n)}\}$ can be identified with M_1 and M_2 respectively. The elements of these multisets will be called points.

Note 4 *We are dealing with multisets instead of sets to insure that $\sigma_i(x, r_1)$ and $\sigma_i(x, r_2)$ map to distinct points of M_i.*

We suppose that M_1 and M_2 are disjoint and introduce the following convention:

When talking about the value of g on a given point, we mean the value of g_1 if the point is in M_1 and the value of g_2 if the point is in M_2.

Let j be an element of $\{1, 2\}$. We say that the values $x, r, g_j(\sigma_j(x, r))$ *sets* $f(x)$ if the value of $\phi(x, r, g_j(\sigma_j(x, r)), w)$ does not depend on w.

In the case $x, r, g_j(\sigma_j(x, r))$ does not set $f(x)$, we can obtain the value of $w = g_{3-j}(\sigma_{3-j}(x, r))$ from $x, r, g_j(\sigma_j(x, r)), f(x)$.

Definition 5 *For some x and r let $y = \sigma_j(x, r)$, $y' = \sigma_{3-j}(x, r)$. We say that $g_j(y)$* forces *$g_{3-j}(y')$ through x and r if $x, r, g_j(\sigma_j(x, r))$ does not set $f(x)$.*

Forcing can be iterated. A sequence of length n points $y_0, \ldots, y_m$ is a *forcing path with respect to a subset of length-n inputs S* if for every i, $0 \leq i < m$

1. i is even and there exists $x \in S$ and $r \in \{0, 1\}^{q(|x|)}$ such that $y_i = \sigma_1(x, r)$ and $y_{i+1} = \sigma_2(x, r)$ and $g_1(y_i)$ forces $g_2(y_{i+1})$ through x and r.

2. i is odd and there exists $x \in S$ and $r \in \{0, 1\}^{q(|x|)}$ such that $y_i = \sigma_2(x, r)$ and $y_{i+1} = \sigma_1(x, r)$ and $g_2(y_i)$ forces $g_1(y_{i+1})$ through x and r.

The *description* of the forcing path consists of the points $y_0, \ldots, y_m$ and the corresponding x's and r's used to force g_1 and g_2 along the path.

If the value of f is known for a subset of inputs, then the values of g along any forcing path with respect to this subset are forced by the value of the first point. These values can be computed in polynomial time if given the description.

From now on any forcing path will start at $\sigma_1(\mathbf{0}, \mathbf{0})$.

The idea of the NP/poly algorithm is that the value of g at the point $\sigma_1(\mathbf{0}, \mathbf{0})$ and the value of f on a small, but appropriate set of inputs will force enough values of g to compute f. Recall that with the help of ϕ we can compute $f(x)$ in polynomial time for an arbitrary choice of r from the values $g_1(\sigma_1(x, r))$ and $g_2(\sigma_2(x, r))$.

For some x the nondeterministic guess will include an r with the property that both $g_1(\sigma_1(x, r))$ and $g_2(\sigma_2(x, r))$ are forced (or in some cases only one of them) and the description of the corresponding forcing paths.

The case in which we need only one of the above values is when f is set by x, r and this value. In this case it is enough to give the forcing path to the corresponding point.

The polynomial advice of our NP/poly machine will contain:

1. The value $g_1(\sigma_1(\mathbf{0}, \mathbf{0}))$,

2. a polynomial length sequence of inputs $x_0, \ldots, x_m$,

3. the sequence of values of f at these points: $f(x_1), \ldots, f(x_m)$.

It remains to show that a small subset of inputs with the desired properties exists.

Lemma 6 *For every function f that has a 2-local-random reduction to boolean g_1 and g_2 using random strings of length $q(n)$, there is a set of length n inputs $x_1, \ldots, x_m$ ($m \leq q(n)+1$) such that for every x there is an r with one of the following properties:*

1. *$\sigma_1(x,r)$ and $\sigma_2(x,r)$ are both on a forcing path (with respect to $x_1, \ldots, x_m$) of length m.*
2. *There is a $j \in \{1,2\}$ such that $\sigma_j(x,r)$ is on a forcing path (with respect to $x_1, \ldots, x_m$) of length at most m and $x, r, g_j(\sigma_j(x,r))$ sets $f(x)$.*

Proof of the lemma:

We construct an exponentially expanding set system $S_0 \subset S_1 \subset \cdots \subset S_m \subseteq M_1 \cup M_2$ and a set of inputs $x_1, \ldots, x_m$ ($m \leq l+1$) recursively such that S_i can be reached by a forcing path with respect to $x_1, \ldots, x_i$ of length at most i.

Moreover for every x there is an r such that one of the following cases holds:

1. both $\sigma_1(x,r)$ and $\sigma_2(x,r)$ are in S_m.
2. there is a $j \in \{1,2\}$ that $\sigma_j(x,r) \in S_m$ and $x, r, g_j(\sigma_j(x,r))$ sets the value of f.

$S_0 = \{\sigma_1(\mathbf{0},\mathbf{0})\}$. Suppose that $S_1, \ldots, S_i$ are already constructed, but S_i does not satisfy the properties required for S_m, i.e. there is an x that none of the conditions 1 and 2 hold for x.

Choose such an x for x_{i+1}.

For every pair $\sigma_1(x_{i+1},r), \sigma_2(x_{i+1},r)$ that coincides with one of the points of S_i (such a pair can never coincide with two points of S_i because otherwise condition 1 would hold for x_{i+1}), the value of the point of the pair that is outside S_i is forced by x_{i+1}, r and the value of g on the point that belongs to S_i.

Let the set S_{i+1} include exactly the points of the pairs that have one common point with S_i. The values of g at these points are forced in $i+1$ step by $x_0, \ldots, x_{i+1}$.

Observe that the size of S_{i+1} is exactly twice the size of S_i. This follows from the fact that for each point of S_i there is exactly one coinciding pair and that the second elements of the pairs are all distinct (see Note 4). The upper bound on m is now implied by $|M_1| = |M_2| = 2^{q(n)}$. ▮

4 Final Comments

There is still a large gap in our knowledge of local-random reductions. Here are some open questions:

- What is the power of two-local-random reductions when the query to one depends on the answer given by the other?
- What is the power of two-local-random reductions when the oracles can output any number of bits?
- What is the minimum $k(n)$ such that any function has a $k(n)$-local-random-reduction.

5 Acknowledgments

We acknowledge Andrew Yao for several important ideas taken from his talk at the DIMACS workshop [6]. Our discussions with Joan Feigenbaum and Gábor Tardos contributed significantly to obtaining the result. We also thank Joan Feigenbaum for her comments on this exposition. Finally, we would like to thank the anonymous referee for several helpful comments.

References

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

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

[3] D. Beaver, J. Feigenbaum, J. Kilian, and P. Rogaway. Security with low communication overhead. In *Advances in Cryptology – Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 62–76. Springer, Berlin, 1991.

[4] J. Feigenbaum and L. Fortnow. On the random-self-reducibility of complete sets. In *Proceedings of the 6th IEEE Structure in Complexity Theory Conference*, pages 124–132. IEEE, New York, 1991.

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

[6] A. Yao. An application of communication complexity to cryptography. Lecture at DIMACS Workshop on Structural Complexity and Cryptography, 1990.

[7] C. Yap. Some consequences of nonuniform conditions on uniform classes. *Theoretical Computer Science*, 26:287–300, 1983.