Are Cook and Karp Ever the Same?

Richard Beigel*
Temple University
NEC Research Institute

Lance Fortnow†
NEC Research Institute

Abstract

We consider the question as to whether there exists a set A such that every set polynomial-time Turing equivalent to A is also many-one equivalent to A. We show that if $\mathbf{E} = \mathbf{NE}$ then no sparse set has this property. We give the first relativized world where there exists a set with this property, and in this world the set A is sparse.

1 Introduction

In their seminal papers on **NP**-completeness, Cook [Coo71] and Karp [Kar72] gave different definitions on the reductions they use. Cook allowed Turing-reductions, where one can decide an instance of one problem by asking an arbitrary collection of witnesses to another problems. Karp looked at the more restrictive many-one completeness that required mapping an instance of one problem into an instance of the other. Computational complexity theory has often looked at the relationship between these two notions of reducibility.

In this paper, we will focus on a fundamental question along these lines: Does there exist a set such that all sets Turing-equivalent to it are also many-one equivalent to it? While there has been some work done on this problem, most notably by Selman [Sel79] and Ambos-Spies, Bentzien, Fejer, Merkle and Stephan [ABF$^+$00], three decades after the work of Cook and Karp this question remains open.

Selman [Sel79] shows that no tally set has this *collapsing* property. Under the assumption $\mathbf{E} = \mathbf{NE}$ we show that in fact no sparse set collapses either.

Some assumption is necessary: We give a relativized world where there is a sparse set A that collapses. This is the first relativized world showing that any set collapses.

In Section 2 we give some background and formal definitions of our problem. Section 4 shows that no sparse set collapses unless $\mathbf{E} \neq \mathbf{NE}$. Section 5 describes the relativized world giving a sparse collapsing set.

2 Preliminaries

Definition 2.1 *A set $A \leq_m^p B$ ("A many-one reduces to B") if there exists a polynomial-time computable function f such that $x \in A \Leftrightarrow f(x) \in B$.*

*Address: Dept. of Computer and Information Sciences, 1805 N. Broad St. Room 303, Philadelphia, PA 19122-6094, USA. Phone: (215)204-8450, fax: (215)204-5082, email: beigel@joda.cis.temple.edu, http://www.cis.temple.edu/~beigel. Supported in part by the National Science Foundation under grants CCR-9996021 and CCR-0049019.

†Address: 4 Independence Way, Princeton, NJ 08540 USA. Email: `fortnow@research.nj.nec.com`. `http://www.neci.nj.nec.com/homepages/fortnow`.

Definition 2.2 *A set* $A \equiv_m^p B$ *("A is many-one equivalent to B") if* $A \le_m^p B$ *and* $B \le_m^p A$*.*

Definition 2.3 *A set* $A \le_T^p B$ *("A Turing reduces to B") if there exists a polynomial-time oracle Turing machine* M *such that* $A = L(M^B)$*.*

Definition 2.4 *A set* $A \equiv_T^p B$ *("A is Turing equivalent to B") if* $A \le_T^p B$ *and* $B \le_T^p A$*.*

Hypothesis 2.5 *There exists an* A *not in* **P** *such that for all* B*, if* $A \equiv_T^p B$ *then* $A \equiv_m^p B$*.*

We call a set A fulfilling Hypothesis 2.5 a *collapsing* set. The condition that A is not in **P** is not strictly necessary—For any $A \neq \emptyset$ in **P**, A is Turing equivalent to $\emptyset$ but not many-one equivalent to it. Nevertheless we will carry along this condition for clarity.

Collapsing is invariant under Turing equivalence: If A is collapsing and B is Turing-equivalent to A then B is also collapsing.

The join of two sets $A \oplus B$ is the set $\{0x \mid x \in A\} \cup \{1x \mid x \in B\}$. The set $A \oplus B$ is many-one reducible to any other set C such that A many-one reduces to C and B many-one reduces to C. A similar statement holds for Turing reductions.

A set A is *sparse* if for some polynomial p, there are at most $p(n)$ elements of A of length n for all n. A set A is *P-printable* if there is a function f computable in polynomial-time such that $h(1^n)$ outputs a list of strings in A. Every **P**-printable set is sparse.

The class $\mathbf{E} = \mathbf{DTIME}(2^{O(n)})$ and $\mathbf{NE} = \mathbf{NTIME}(2^{O(n)})$.

We use Σ to represent our alphabet $\{0, 1\}$.

3 Previous Work

Selman [Sel79] looked at whether collapsing sets could be tally sets, i.e., subsets of 1^*.

Theorem 3.1 (Selman) *No tally set is collapsing.*

Since we make use of Theorem 3.1 in this paper, we give a proof here for completeness.

Proof of Theorem 3.1: Suppose $A \subseteq 1^*$ is a collapsing set. Consider the following set B.

$$B = \{x \in \Sigma^* \mid x \le A(\epsilon)A(1)A(11)\ldots A(1^{|x|-1})\}$$

where we use a lexicographic ordering.

One can easily Turing-reduce B to A. A also Turing reduces to B by using binary search to find $A(\epsilon)A(1)\ldots$.

The sets B and $\overline{B}$ are both Turing-equivalent to A so they are Turing-equivalent to each other. By assumption this means there is a many-one reduction f reducing B to $\overline{B}$.

Note that if $x \in B$ and $y \le x$ then $y \in B$. Since exactly one of x and $f(x)$ could be in B, it must be the lexicographically least one. This lets us test membership in B in polynomial-time. Since A is equivalent to B we also have A in **P**. $\square$

Ambos-Spies, Bentzien, Fejer, Merkle and Stephan [ABF$^+$00] proved many other results about collapsing sets including the following.

Theorem 3.2 (ABFMS) *If* A *is a collapsing set then*

1. *A is computable, and*

2. *A is not hard for* **EXP***.*

4 Separations

In this section we show that under a hard-to-disprove assumption, no sparse sets can be collapsing. Some assumption is needed as we show in Section 5.

Theorem 4.1 *If* $\mathbf{E} = \mathbf{NE}$ *then no sparse set is collapsing.*

To prove Theorem 4.1 we first need the following lemma.

Lemma 4.2 *No subset of a* $\mathbf{P}$*-printable set is collapsing.*

Proof: Let B be a subset of a $\mathbf{P}$-printable set C. Let $h(1^n) = \{y_{n,1}, \ldots, y_{n,k}\}$ list the elements of C of length n. Define the set D as

$$D = \{1^{\langle n,i \rangle} \mid y_{n,i} \in B\}.$$

The set D is Turing-equivalent to B so if B is collapsing then D is collapsing. But D is a tally set so D is not collapsing by Theorem 3.1. □

We also use the following lemma from Hartmanis, Immerman and Sewelson [HIS85].

Lemma 4.3 (HIS) *If* $\mathbf{E} = \mathbf{NE}$ *then all sparse sets in* $\mathbf{NP}$ *are* $\mathbf{P}$*-printable.*

Proof: Let A be a sparse set in $\mathbf{NP}$. Consider the sets B defined as

$$\begin{aligned} B &= \{\langle n,w \rangle \mid \text{There exist } x_1 < x_2 < \cdots < x_w \text{ of length } n \text{ in } A\} \\ C &= \{\langle n,w,i,j \rangle \mid \text{There exist } x_1 < x_2 < \cdots < x_w \text{ of length } n \text{ in } A \text{ such that the } j\text{th bit of } x_i \text{ is } 1\} \end{aligned}$$

where the numbers n, w, i and j are written in binary. The sets B and E are in $\mathbf{NE} = \mathbf{E}$. In particular, in polynomial in n time we can find the maximum w such that $\langle n,w \rangle$ is in B. This tells us how many strings of length n are in A. Note that there is only one possible choice for $x_1, \ldots, x_w$. We now determine whether $\langle n,w,i,j \rangle$ in C for all of the appropriate i and j to reconstruct the strings $x_1, \ldots, x_w$ in A. □

Proof of Theorem 4.1: Suppose $\mathbf{E} = \mathbf{NE}$ and let A be a sparse collapsing set. A is Turing-equivalent to $\overline{A}$ so by assumption there is a function f reducing A to $\overline{A}$. Consider the set B defined as

$$B = \{\langle 1^n, x \rangle \mid \text{There exists a } y \text{ of length } n \text{ such that } f(y) = x\}.$$

The set B is in $\mathbf{NP}$.

We also claim that B is sparse. The number of length-n strings in B is bounded by $\sum_{m \leq n} |f(\Sigma^m)|$, so it suffices to show that $|f(\Sigma^n)|$ is polynomially bounded. We have $|f(\Sigma^n)| \leq |f(A \cap \Sigma^n)| + |f(\overline{A} \cap \Sigma^n)|$. Each of those terms is polynomially bounded because A is sparse and $f(\overline{A}) \subseteq A$.

By Lemma 4.3, B is $\mathbf{P}$-printable. Consider $C \subseteq B$ defined as

$$C = \{\langle 1^n, x \rangle \mid \langle 1^n, x \rangle \in B \text{ and } x \in A\}.$$

The set C is Turing-equivalent to A: C is reducible to A because B is in $\mathbf{P}$; for the other direction, we have $y \in A \iff \langle 1^{|y|}, f(y) \rangle \notin C$.

Since A is collapsing C is also collapsing. But C is a subset of a $\mathbf{P}$-printable set so C does not collapse by Lemma 4.2. □

5 Collapse

In this section we present the first relativized world where there exists a collapsing set, i.e., Hypothesis 2.5 holds.

Theorem 5.1 *There exists an oracle B and a sparse set $A \notin \mathbf{P}^B$ such that for every set C if $C \equiv_T^{p^B} A$ then $C \equiv_m^{p^B} A$.*

Proof: Let $M_1, M_2, \ldots$ be an enumeration of polynomial-time oracle Turing machines. Note that $C \equiv_T^{p^B} A$ is the same as saying there exist machines M_i and M_j such that $C = L(M_i^{B \oplus A})$ and $A = L(M_j^{B \oplus C})$. Let $R_1, R_2, \ldots$ and $S_1, S_2, \ldots$ be enumerations of polynomial-time machines such that for every i and j there is a k such that $R_k = M_i$ and $S_k = M_j$. We assume without loss of generality that M_i, R_i and S_i run in time at most n^i for n the length of the input.

We need to fulfill the following requirements:

1. E_i: There is an x such that $x \in A \Leftrightarrow M_i^B(x)$ rejects.

2. F_i: If $C = L(R_i^{B \oplus A})$ and $A = L(S_i^{B \oplus C})$ then $A \leq_m^{p^B} C$ and $C \leq_m^{p^B} A$.

The E_i requirements guarantee that A is not in $\mathbf{P}^B$ while the F_i requirements guarantee that A is collapsing relative to B.

When we fulfill our F_i requirements we will allow our many-one reductions to occasionally just accept and reject. Since our E_i requirements will guarantee that A is not in $\mathbf{P}^B$ we can just map to fixed strings known to be in or out of A and C.

We will define our oracle B as a function $B : \{0,1\}^n \to \{0,1\}^{n^k}$ for some fixed k. One can use standard coding ideas to create a subset of $\{0,1\}^*$ equivalent to B.

We build the oracle in stages. In each stage s we will pick $n_s = 2^{n_{s-1}}$.

We will set $B(1^{n_{s-1}^{\log n_{s-1}}})$ to be a list of the elements of $A \cap \Sigma^{n_{s-1}}$. This guarantees that any reduction that looks at previous strings in A can just find them easily in B.

Fix s and let $n = n_s$. In stage s we will fulfill requirement E_s for some string of length n. We will also guarantee that $F_1, \ldots, F_s$ hold for strings of length between $n_{s-1}^{\log n_{s-1}}$ and $n_s^{\log n_s}$. At the end of the construction, each requirement F_i will have been fulfilled for all but a finite number of strings.

Choose a Kolmogorov random string of length n^2 and break it into n pieces of length n each and call these strings $y_1, \ldots y_n$. Let $U = \{y_1, y_1 + 1, \ldots, y_n, y_n + 1\}$ where $y_i + 1$ is the lexicographically next string after y_i. We will guarantee that $A \cap \Sigma^n$ is a subset of U.

We set $B(u) = 1$ for each u in U.

We will put conditions on A of the form $u \in A \Leftrightarrow v \notin A$. The construction will not build a contradictory set of conditions and will allow some freedom so at the end we can choose A to fulfill E_s.

We say u and v are *connected* if the value of $A(u)$ is forced by setting $v \in A$, given the other conditions we have already currently set.

The first set of conditions we use will require that $y_i \in A \Leftrightarrow y_i + 1 \notin A$ for each i.

Eventually we will connect all the members of U. For all pairs u and v of elements in U we will encode at $B(\langle u, v \rangle)$ whether $u \in A \Leftrightarrow v \in A$ or $u \in A \Leftrightarrow v \notin A$.

We will now work in substages $j = 1, \ldots, s$. In each substage we will handle requirement F_j.

In substage j and later we guarantee not to add any strings to B of length less than $n^{j^{10}}$. Thus $\mathrm{DTIME}(n^j)$ machines or even a composition of five of them acting on strings of length n will not be affected by strings we add to B.

We first describe how to construct the many-one reduction from C to A. Fix an input w and simulate $S_j^{B\oplus A}(w)$, the potential Turing reduction from C to A. If S_j makes a query to B we just query it. Now consider all of the queries made to A. Since we have encoded the relationship between all pairs u and v in U, we can reduce $S_j^{B\oplus A}(w)$ to a single query y_i in A. If S_j accepts or rejects regardless of whether y_i is in A then just accept or reject respectively. If S_j accepts iff y_i is in A then we have our many-one reduction. Otherwise reduce to $y_i + 1$.

We now describe how to construct the many-one reduction from A to C. Fix a u of length n. Check with B to see if u is in U. If not then we just reject.

Simulate $R_j^{B\oplus C}(u)$, the potential reduction from A to C. Consider the strings it queries in C in order. Let w be the next string queried and consider the reduction $S_j^{B\oplus A}(w)$. Note that $S_j^{B\oplus A}$ cannot query any string in U not connected to u or we would violate the randomness of the string used to generate the y_i's.

By the above argument we have only three possibilities. If S_j accepts or rejects regardless of A then we answer R_j accordingly and go to R_j's next query. If $S_j^{B\oplus A}(w)$ accepts iff $u \in A$ then we just output w as our many-one reduction and we then go on to the next u in U.

The problem occurs if S_j accepts iff $u \notin A$. Set $w = w_u$ and we'll handle this case later. Go on and handle the next u in U.

Suppose for some u, $R_j^{B\oplus A}(u)$ runs out of queries and just accepts or rejects. We then set u in A to diagonalize against R_j and go on to the next substage.

Now consider the w_u's. If we have a w_u and a w_v (with u connected to v) such that $u \in A \iff v \notin A$, then we can just use the many-one reduction of v to w_u and u to w_v. Otherwise pick a w_u and a w_v not connected, and add the condition $u \in A \iff v \notin A$. We encode this connection in the oracle by setting $B(\langle 1^{n^{j^{10}}}, u\rangle) = v$ and $B(\langle 1^{n^{j^{10}}}, v\rangle) = u$. We then have the many-one reduction map all of the r with $r \in A \iff u \in A$ to w_v and all of the s with $s \in A \iff v \in A$ to w_u. We then remove from consideration all of these w_r's and w_s's and repeat.

If we have one w_u remaining without a w_v to connect to it then just set $u \in A$ and encode all of the r connected to u in B as $B(\langle 1^{n^{j^10}}, u\rangle) =$ TRUE or FALSE as appropriate and have the many-one reduction on u just accept.

At the beginning of stage s we have n connected components of elements of U. In each stage we shrink the number of components in at most half and possibly determine the elements of one component. Since $s \ll \log n$, we still have components remaining after substage s. Pick some u in one of these components and set u in A to diagonalize against the $M_s(u)$ to fulfill E_s. $\square$

6 Further Research

Of course, the question whether there exists a collapsing set still remains open. In particular can one modify the oracle construction in Section 5 to simply create a (nonsparse) collapsing set without needing a relativized world?

Another related question is whether all the Turing-complete sets for **NP** are many-one complete. This question is incomparable to the existence of a collapsing set since in this case we restrict ourselves to sets in the class **NP**, which not necessarily closed under Turing reductions. Nevertheless, perhaps our techniques could also shed light on that problem.

Acknowledgments

We thank Bill Gasarch, Aduri Pavan, Marcus Schaefer, Alan Selman and Frank Stephan for helpful discussions. Bill Gasarch also provided a careful proofreading of this exposition.

References

[ABF$^+$00] K. Ambos-Spies, L. Bentzien, P. Fejer, W. Merkle, and F. Stephan. Collapsing polynomial-time degrees. In *Logic Colloquium '98*, volume 13 of *Lecture Notes in Logic*, pages 1–24. Association of Symbolic Logic, 2000.

[Coo71] S. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on the Theory of Computing*, pages 151–158. ACM, New York, 1971.

[HIS85] J. Hartmanis, N. Immerman, and V. Sewelson. Sparse sets in $NP - P$: *EXPTIME* versus *NEXPTIME*. *Information and Control*, 65:158–181, 1985.

[Kar72] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP. *Mathematical Systems Theory*, 13:55–65, 1979.