THEORY OF COMPUTING, Volume 5 (2009), pp. 135–140
www.theoryofcomputing.org

NOTE

A Simple Proof of Toda's Theorem*

Lance Fortnow†

Received: January 21, 2009; published: July 3, 2009.

Abstract: Toda in his celebrated paper showed that the polynomial-time hierarchy is contained in $\mathrm{P}^{\#\mathrm{P}}$. We give a short and simple proof of the first half of Toda's Theorem that the polynomial-time hierarchy is contained in $\mathrm{BPP}^{\oplus\mathrm{P}}$. Our proof uses easy consequences of relativizable proofs of results that predate Toda.

For completeness we also include a proof of the second half of Toda's Theorem.

ACM Classification: F.1.3

AMS Classification: 68Q15

Key words and phrases: Toda's Theorem, Relativization

1 Introduction

In 1991, Toda proved his celebrated theorem [7].

Theorem 1.1 (Toda)**.** $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}}$.

Here PH is the set of languages in the polynomial-time hierarchy.

The proof of Theorem 1.1 follows from the following two lemmas (since $\mathrm{BPP}^A \subseteq \mathrm{PP}^A$ for all A).

Lemma 1.2 (Toda)**.** $\mathrm{PH} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}$.

Lemma 1.3 (Toda)**.** $\mathrm{PP}^{\oplus\mathrm{P}} \subseteq \mathrm{P}^{\#\mathrm{P}}$.

In this paper we give a short proof of Lemma 1.2 using relativizable versions of results that predate Toda's Theorem. For completeness we will give a proof of Lemma 1.3 as well.

*This result first appeared in the *Computational Complexity* weblog [5].

†Supported in part by NSF grants CCF-0829754 and DMS-0652521.

© 2009 Lance Fortnow
Licensed under a Creative Commons Attribution License
DOI: 10.486/toc.2009.v005a007

2 Preliminaries

The Complexity Zoo [1] and the Arora-Barak textbook [2] are good sources for descriptions of the complexity classes used in this note.

To relativize Satisfiability to an oracle A, we allow our CNF formulas to have predicates $A_0, A_1, A_2, \ldots$ where A_n is an n-ary predicate defined so $A_n(x_1, \ldots, x_n)$ is true exactly when $x_1 \ldots x_n$ is in A. For every A, SAT^A is NP^A-complete.

If $\mathcal{C}$ and $\mathcal{D}$ are relativizable classes, $\mathcal{C}^{\mathcal{D}} = \cup_{A \in \mathcal{D}} \mathcal{C}^A$. If $\mathcal{D}$ has a complete set D (such as $\mathcal{D} = \oplus\mathrm{P}$) then $\mathcal{C}^{\mathcal{D}} = \mathcal{C}^D$.

When we relativize a class like $\mathrm{BPP}^{\oplus\mathrm{P}}$ to an oracle A, both the BPP and the $\oplus$P machines should have access to the oracle A. The BPP machine can make its queries to A via the $\oplus\mathrm{P}^A$ oracle so we have $(\mathrm{BPP}^{\oplus\mathrm{P}})^A = \mathrm{BPP}^{(\oplus\mathrm{P}^A)}$ which we will write simply as $\mathrm{BPP}^{\oplus\mathrm{P}^A}$.

We define the polynomial-time hierarchy relative to A recursively:

- $\Sigma_0^A = \mathrm{P}^A$.
- $\Sigma_{i+1}^A = \mathrm{NP}^{\Sigma_i^A}$.
- $\mathrm{PH}^A = \cup_i \Sigma_i^A$.

The class GapP is the set of #P functions closed under subtraction. In particular GapP functions may take on negative values. Like #P, GapP functions are closed under uniform exponential-sized sums and polynomial-sized products and unlike #P, GapP functions are also closed under subtraction [3].

3 Proof of Toda's first lemma

We start with the following three results, all of which have proofs that easily relativize.

Theorem 3.1 (Valiant-Vazirani [8])**.** *There is a probabilistic polynomial-time procedure that, given a Boolean formula ϕ, will output formulas $\psi_1, \ldots, \psi_k$ such that*

- *if ϕ is not satisfiable then, for every i, ψ_i is not satisfiable;*
- *if ϕ is satisfiable then, with high probability, for some i, ψ_i has exactly one solution.*

Theorem 3.2 (Papadimitriou-Zachos [6])**.** $\oplus\mathrm{P}^{\oplus\mathrm{P}} = \oplus\mathrm{P}$.

Theorem 3.3 (Zachos [9])**.** *If* $\mathrm{NP} \subseteq \mathrm{BPP}$ *then* $\mathrm{PH} \subseteq \mathrm{BPP}$.

We first need the following easy consequence of Theorem 3.1 noted by Toda [7].

Lemma 3.4 (Valiant-Vazirani, Toda)**.** $\mathrm{NP} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}$.

Proof Sketch. Given a Boolean formula ϕ, randomly choose $\psi_1, \ldots, \psi_k$ (as in Theorem 3.1) and accept if any of the ψ_i have an odd number of satisfying assignments. Lemma 3.4 now follows from Theorem 3.1. $\square$

Proof of Lemma 1.2.

1. By relativizing Lemma 3.4, we have
$$\mathrm{NP}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}^{\oplus\mathrm{P}}}.$$

2. Now apply Theorem 3.2 to get
$$\mathrm{NP}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}.$$

3. By relativizing Theorem 3.3, we have
$$\mathrm{NP}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}} \Rightarrow \mathrm{PH}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}.$$

4. Combining (2) and (3) we have
$$\mathrm{PH} \subseteq \mathrm{PH}^{\oplus\mathrm{P}} \subseteq \mathrm{BPP}^{\oplus\mathrm{P}}.$$

□

If we had replaced the use of Theorem 3.3 with the relativizable proof of it, we would essentially recover Toda's original proof.

4 Proof of Toda's second lemma

For completeness we include a proof of Lemma 1.3 in this section. We give a GapP-based variant of Toda's original proof [7] originally given in a survey paper by the author [4].

We will use the following GapP characterization of $\oplus$P [3].

Lemma 4.1 (Fenner-Fortnow-Kurtz)**.** *A language B is in $\oplus$P if and only if there is a* GapP *function f such that*

- *if $x \in B$ then $f(x) \equiv 1 \pmod 2$;*
- *if $x \notin B$ then $f(x) \equiv 0 \pmod 2$.*

We can define PP^A using P^A predicates.

Lemma 4.2. *A language L is in PP^A if and only if there is a language $B \in \mathrm{P}^A$ and a polynomial q such that*

- *if $x \in L$ then*
$$\left|\left\{y \in \Sigma^{q(|x|)} \,\middle|\, (x,y) \in B\right\}\right| \geq \left|\left\{y \in \Sigma^{q(|x|)} \,\middle|\, (x,y) \notin B\right\}\right|.$$
- *if $x \notin L$ then*
$$\left|\left\{y \in \Sigma^{q(|x|)} \,\middle|\, (x,y) \in B\right\}\right| < \left|\left\{y \in \Sigma^{q(|x|)} \,\middle|\, (x,y) \notin B\right\}\right|.$$

Combining Lemmas 4.1 and 4.2 with Theorem 3.2 (which implies $\mathrm{P}^{\oplus\mathrm{P}} = \oplus\mathrm{P}$) we have the following characterization of $\mathrm{PP}^{\oplus\mathrm{P}}$.

Lemma 4.3. *A language L is in* $\mathrm{PP}^{\oplus\mathrm{P}}$ *if and only if there is a* GapP *function $f(x,y)$ and a polynomial q such that*

- *if $x \in L$ then*
$$\left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 1 \pmod 2\right\}\right| \geq \left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 0 \pmod 2\right\}\right|.$$
- *if $x \notin L$ then*
$$\left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 1 \pmod 2\right\}\right| < \left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 0 \pmod 2\right\}\right|.$$

We give an $\mathrm{FP}^{\mathrm{GapP}}$ algorithm to compute
$$\left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 1 \pmod 2\right\}\right|$$
and
$$\left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 0 \pmod 2\right\}\right|.$$
Lemma 1.3 follows since $\mathrm{FP}^{\mathrm{GapP}} = \mathrm{FP}^{\#\mathrm{P}}$ [3].

Consider the polynomial $g(m) = 3m^2 - 2m^3$. Let $g^{(k)}(m) = \overbrace{g(g(\ldots g}^{k}(m)\ldots))$.

Lemma 4.4. *For all m,*

1. *if* $m \equiv 0 \pmod{2^j}$ *then* $g(m) \equiv 0 \pmod{2^{2j}}$*,*
2. *if* $m \equiv 1 \pmod{2^j}$ *then* $g(m) \equiv 1 \pmod{2^{2j}}$*,*
3. *if* $m \equiv 0 \pmod 2$ *then* $g^{(k)}(m) \equiv 0 \pmod{2^{2^k}}$*, and*
4. *if* $m \equiv 1 \pmod 2$ *then* $g^{(k)}(m) \equiv 1 \pmod{2^{2^k}}$*.*

Proof. Items (1) and (2) follow from simple algebra, items (3) and (4) by induction using (1) and (2). □

Let $h(x,y) = g^{(1+\lceil \log q(|x|)\rceil)}(f(x,y))$. Since GapP functions are closed under uniform exponential-size sums and polynomial-size products, $h(x,y)$ is itself a GapP function and by Lemma 4.4

- if $f(x,y) \equiv 1 \pmod 2$ then $h(x,y) \equiv 1 \pmod{2^{q(|x|)+1}}$, and
- if $f(x,y) \equiv 0 \pmod 2$ then $h(x,y) \equiv 0 \pmod{2^{q(|x|)+1}}$.

Define $r(x)$ as
$$r(x) = \sum_{y \in \Sigma^{q(|x|)}} h(x,y),$$
also a GapP function. We then have
$$(r(x) \bmod 2^{q(|x|)+1}) = \left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 1 \pmod 2\right\}\right|$$
and
$$2^{q(|x|)} - (r(x) \bmod 2^{q(|x|)+1}) = \left|\left\{y \in \Sigma^{q(|x|)} \;\middle|\; f(x,y) \equiv 0 \pmod 2\right\}\right|,$$
completing the proof. □

Remark 4.5. Toda uses #P functions and the polynomial $g(m) = 4m^3 + 3m^4$. Lemma 4.3 now holds with each occurrence of "1" replaced by "-1."

References

[1] S. AARONSON: The complexity zoo. http://complexityzoo.com. 136

[2] S. ARORA AND B. BARAK: *Complexity Theory: A Modern Approach*. Cambridge University Press, Cambridge, 2009. 136

[3] S. FENNER, L. FORTNOW, AND S. KURTZ: Gap-definable counting classes. *J. Comput. System Sci.*, 48(1):116–148, 1994. [doi:10.1016/S0022-0000(05)80024-8]. 136, 137, 138

[4] L. FORTNOW: Counting complexity. In L. HEMASPAANDRA AND A. SELMAN, editors, *Complexity Theory Retrospective II*, pp. 81–107. Springer, 1997. 137

[5] L. FORTNOW: Making pigs fly. *Computational Complexity Weblog*, June 2006. http://weblog.fortnow.com/2005/06/making-pigs-fly.html. 135

[6] C. PAPADIMITRIOU AND S. ZACHOS: Two remarks on the power of counting. In *Proc. 6th GI-Conf. Theor. Comput. Sci.*, volume 145 of *LNCS*, pp. 269–276. Springer, Berlin, 1983. [doi:10.1007/BFb0009651]. 136

[7] S. TODA: PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. [doi:10.1137/0220053]. 135, 136, 137

[8] L. VALIANT AND V. VAZIRANI: NP is as easy as detecting unique solutions. *Theoret. Comput. Sci.*, 47:85–93, 1986. [doi:10.1016/0304-3975(86)90135-0]. 136

[9] S. ZACHOS: Probabilistic quantifiers and games. *J. Comput. System Sci.*, 36(3):433–451, 1988. [doi:10.1016/0022-0000(88)90037-2]. 136

AUTHOR

Lance Fortnow
professor
Department of Electrical Engineering and Computer Science
Northwestern University
Evanston, Illinois
fortnow@eecs.northwestern.edu
`http://lance.fortnow.com`

ABOUT THE AUTHOR

LANCE FORTNOW received his Ph. D. under Michael Sipser in Applied Mathematics at MIT in 1989. Before Northwestern he spent his academic career at the University of Chicago with the exception of a senior research scientist position at the NEC Research Institute from 1999 to 2003. In 1992 he received the NSF Presidential Faculty Fellowship and was a Fulbright Scholar visiting CWI in Amsterdam 1996-97. Fortnow studies computational complexity and its applications to electronic commerce, quantum computation, bioinformatics, learning theory and cryptography. His early work on interactive proofs precipitated the development of probabilistically checkable proofs and inapproximability theory. Fortnow co-writes the popular scientific and academic weblog Computational Complexity.