CS254: Computational Complexity Theory
Prof. Luca Trevisan

Final Project
Ananth Raghunathan[*]

Toda's Theorem: $\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}}$

1 Introduction

The class $\mathbf{NP}$ captures the difficulty of finding certificates. However, in many contexts, one is interested not just in a single certificate, but actually counting the number of certificates. In this manuscript we look at a famous result involving $\#\mathbf{P}$, (pronounced "sharp p"), a complexity class that captures this notion. Counting problems arise in diverse fields, often in situations having to do with estimations of probability. Examples include statistical estimation, statistical physics, network design, and more. Counting problems are also studied in a field of mathematics called enumerative combinatorics, which tries to obtain closed-form mathematical expressions for counting problems. To give an example, in the 19th century Kirchoff showed how to count the number of spanning trees in a graph using a simple determinant computation.

Definition 1. *A function $f : \{0,1\}^* \to \mathbb{N}$ is in $\#\mathbf{P}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial time algorithm M such that for every $x \in \{0,1\}^*$:*

$$f(x) = \left| \left\{ y \in \{0,1\}^{p(|x|)} : M(x,y) = 1 \right\} \right|$$

The biggest open problem regarding $\#\mathbf{P}$ is whether or not all problems in this class are efficiently solvable. In other words, whether $\#\mathbf{P} = \mathbf{FP}$ where $\mathbf{FP}$ is the analogue class to the class $\mathbf{P}$ for functions with more than 1 bit of output. Clearly, finding the number of certificates is at least as hard as finding whether or not there exists a certificate, therefore, if $\#\mathbf{P} = \mathbf{FP}$, certainly $\mathbf{P} = \mathbf{NP}$. We also know that $\#\mathbf{P} \subseteq \mathbf{PSPACE}$, because we can count the number of certificates given polynomial amount of space.

In 1979, when Valiant introduced the class $\#\mathbf{P}$, he also showed that finding the permanent of a matrix (even when its entries are restricted to $\{0,1\}$) is $\#\mathbf{P}$-complete. But surprisingly, we also know that we can approximately count given access to an $\mathbf{NP}$ oracle. In other words, approx$-\#\mathbf{P} \subseteq \mathbf{BPP}^{\mathbf{NP}}$. Since $\mathbf{BPP} \subseteq \Sigma_2^{(p)}$, approx$-\#\mathbf{P} \subseteq \Sigma_3^{(p)}$.

This was the state of affairs in the 1980s, where people were interested in the relative powers of $\mathbf{PH}$ and $\#\mathbf{P}$. Both were considered natural generalizations of $\mathbf{NP}$, although it is not immediately clear how each of their definitions, viz. alternation, and the ability to count certificates were comparable in any sense. However, in 1989, Seinosuke Toda [1] showed that $\#\mathbf{P}$ is a very powerful language. Indeed, with only a single call to a $\#\mathbf{P}$ oracle, one can decide membership of a quantified boolean formula in $\mathbf{PH}$. This came as a very surprising result to the community and was one of the landmark results of complexity theory. Toda's theorem therefore implies that any problem in the polynomial hierarchy is deterministic polynomial time *Turing reducible* to a counting problem. This won Toda the Gödel prize in 1998. Here is the citation for the Gödel prize. even with a single call to the oracle

!

[*] `ananthr@cs.stanford.edu, ananthr1987@gmail.com`

$PH \subseteq P^{PP} \overset{?}{\Longrightarrow} PH^{PP} \subseteq P^{PP}$ as well?

Crucial step: $PH^{PP} \subseteq (P^{PP})^{PP}$.

In the remarkable paper "PP is as hard as the polynomial-time hierarchy" Seinosuke Toda showed that two fundamental and much studied computational concepts had a deep and unexpected relationship. The first is that of alternation of quantifiers—if one alternates existential and universal quantifiers for polynomial time recognizable functions one obtains the polynomial time hierarchy. The second concept is that of counting the exact number of solutions to a problem where a candidate solution is polynomial time recognizable. Toda's astonishing result is that the latter notion subsumes the former—for any problem in the polynomial hierarchy there is a deterministic polynomial time reduction to counting. This discovery is one of the most striking and tantalizing results in computer science. It continues to serve as an inspiration to those seeking to understand more fully the relationships among the fundamental concepts in computer science.

$\boxed{\textbf{Theorem 1 (Toda [1]).}\ \mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}}}$

1.1 Preliminaries

We use $\oplus$ to denote a parity quantifier that is formally defined below:

Definition 2. *The quantifier $\oplus$ is defined as follows: For every Boolean formula φ on n variables, $\bigoplus_{(x\in\{0,1\}^n)} \varphi(x)$ is true iff the number of x's such that $\varphi(x)$ is true is odd. The language $\underline{\oplus\mathsf{SAT}}$ denotes the set of all true quantified Boolean formulae of the form $\bigoplus_{(x\in\{0,1\}^n)} \varphi(x)$ where φ is an unquantified Boolean formula (not necessarily in CNF).*

Note 1. If we identify true with 1 and false with 0, $\bigoplus_{(x\in\{0,1\}^n)} \varphi(x) = \sum_{x\in\{0,1\}^n} \varphi(x) \pmod 2$. Also, $\bigoplus_{(x\in\{0,1\}^n)} \varphi(x) = \bigoplus_{(x_1\in\{0,1\})} \bigoplus_{(x_2\in\{0,1\})} \cdots \bigoplus_{(x_n\in\{0,1\})} \varphi(x_1, \ldots, x_n)$

Let φ be a boolean formula. We denote by $\#(\varphi)$ the number of solutions to φ. Given two unquantified formulae $\varphi(x)$ and $\varphi(y)$, define:

- $\varphi \cdot \varphi'$ as the formula $\varphi(y) \wedge \varphi'(z)$ where y and z are two disjoint sets of variables. It is easy to see that $\#(\varphi \cdot \varphi') = \#(\varphi) \cdot \#(\varphi')$.
- $\varphi + \varphi'$ as the formula $(w \wedge \varphi(y)) \vee (\overline{w} \wedge \varphi'(y))$, where w is a single additional boolean variable. Then, $\#(\varphi + \varphi') = \#(\varphi) + \#(\varphi')$.
- 1 is an arbitrary formula with exactly one satisfying assignment.

Therefore, it follows immediately that:

$$\left(\bigoplus_x \varphi(x)\right) \wedge \left(\bigoplus_y \psi(y)\right) \Leftrightarrow \bigoplus_{x,y} (\varphi \cdot \psi)(x, y)$$

$$\neg\left(\bigoplus_x \varphi(x)\right) \Leftrightarrow \bigoplus_{x,z} (\varphi + 1)(x, z)$$

$$\left(\bigoplus_x \varphi(x)\right) \vee \left(\bigoplus_y \psi(y)\right) \Leftrightarrow \bigoplus_{x,y,z} ((\varphi + 1) \cdot (\psi + 1) + 1)(x, y, z)$$

2 Valiant-Vazirani Theorem

One of the key constructions that will be used (repeatedly) in the proof of Toda's theorem is the randomized reduction of Valiant-Vazirani of SATto Unique-SAT. We will use a weaker corollary of the Valiant-Vazirani theorem as stated below:

Theorem 2 (Valiant-Vazirani [2]). *There exists a probabilistic polynomial time algorithm A such that for every n-variable Boolean formula φ, we have:*

$$\varphi \in \mathsf{SAT} \Rightarrow \Pr[A(\varphi) \in \text{Unique-}\mathsf{SAT}] \geq \frac{1}{8n}$$
$$\varphi \notin \mathsf{SAT} \Rightarrow \Pr[A(\varphi) \text{ is satisfiable}] = 0.$$

In particular, since $A(\varphi)$ has a unique solution, it is a member of $\oplus\mathsf{SAT}$. Also, if $A(\varphi)$ has no solutions, then it does not belong to $\oplus\mathsf{SAT}$. Thus, the Valiant-Vazirani theorem implies the following Lemma.

Lemma 1. *There exists a probabilistic polynomial time algorithm A such that for every n-variable Boolean formula φ, we have:*

$$\varphi \in \mathsf{SAT} \Rightarrow \Pr[A(\varphi) \in \oplus\mathsf{SAT}] \geq \frac{1}{8n}$$
$$\varphi \notin \mathsf{SAT} \Rightarrow \Pr[A(\varphi) \in \oplus\mathsf{SAT}] = 0.$$

One interpretation of the Valiant-Vazirani theorem is that given an algorithm to distinguish whether or not a formula had exactly one or zero satisfying assignments, then we can construct a randomized algorithm that would decide NP-complete problems, which would imply that $\mathbf{NP} = \mathbf{RP}$.

Extending upon this interpretation, if we could somehow determine if a formula had an even number or an odd number of satisfying assignments, then we can construct a randomized algorithm that would solve everything in the polynomial hierarchy.

Although this does not prove Toda's theorem (because the result is deterministic) it does show that $\mathbf{PH} \subseteq \mathbf{BPP}^{\oplus\mathbf{P}}$. We require oracle access to a $\#\mathbf{P}$ oracle to derandomize this reduction.

3 Toda's Theorem

As discussed in the previous section, we begin by constructing a randomized algorithm to decide problems in $\mathbf{PH}$ given oracle access to a $\oplus\mathbf{P}$ oracle. We start off with the following lemma:

Lemma 2. *Let $c \in \mathbb{N}$ be a constant and TQBF be the set of all true quantified boolean formulas. There exists a probabilistic polynomial time algorithm B such that for every φ, a quantified boolean formula with c levels of alternations,*

$$\varphi \in \mathsf{TQBF} \Rightarrow \Pr[B(\varphi) \in \oplus\mathsf{SAT}] \geq \frac{2}{3}$$
$$\varphi \notin \mathsf{TQBF} \Rightarrow \Pr[B(\varphi) \in \oplus\mathsf{SAT}] = 0.$$

To derandomize, we need another lemma:

Lemma 3. *There is a deterministic polynomial time transformation T, that, for every input formula φ that is an input for $\oplus\mathsf{SAT}$, constructs $\psi = T(\varphi, 1^m)$, an unquantified boolean formula such that:*

$$\varphi \in \oplus\mathsf{SAT} \Rightarrow \#(\psi) = -1 \pmod{2^m}$$
$$\varphi \notin \oplus\mathsf{SAT} \Rightarrow \#(\psi) = 0 \pmod{2^m}.$$

Proof (Toda's theorem). Equipped with the above two lemmas, we prove Toda's theorem. In the next section, we show the proofs of the two lemmas. We are given an input quantified boolean formula φ with c alternations, for some constant c and we need to determine whether or not $\varphi \in \mathsf{TQBF}$.

First consider the reduction B from Lemma 2. Since B is randomized, we can consider it as a deterministic procedure that takes in a random string r of length $m-1$ along with the input φ and outputs a formula ψ_r depending on the input r. Applying $T(\cdot, 1^m)$ from Lemma 3 to ψ_r, we obtain a formula ψ'_r.

Now, consider the following quantity: $K = \sum_{r \in \{0,1\}^{m-1}} \#(\psi'_r)$, which counts the number of *pairs* (x, r) such that assignment x satisfies the formula ψ'_r. We look at the following two cases:

- Case 1: If $\varphi \notin \mathsf{TQBF}$, then regardless of the choice of r, $\#(\psi'_r) = 0 \pmod{2^m}$. Therefore $K = 0 \pmod{2^m}$.
- Case 2: If $\varphi \in \mathsf{TQBF}$, then for at least $2/3$ fraction of the strings r, we have $\#(\psi_r)$ is odd. Then this also implies that for at least $2/3$ fraction of the strings r, $\#(\psi'_r) = -1 \pmod{2^m}$, and for the remaining at most $1/3$ fraction, $\#(\psi'_r) = 0 \pmod{2^m}$. Thus, K must fall in the range $[-\frac{2}{3}2^{m-1}, -2^{m-1}] \pmod{2^m}$, which necessarily means that $K \neq 0 \pmod{2^m}$.

Now, we have our $\mathbf{P}^{\#\mathbf{P}}$ algorithm to decide φ. We run the reductions from Lemmas 2 and 3 and ask a $\#\mathbf{P}$ oracle to count the number of pairs (x, r) such that x satisfies ψ'_r.[1] If the answer divides 2^m, then we reject; otherwise accept. ∎

4 Proof of Lemmas 2 and 3

In this section, we prove the two lemmas used in the previous section.

Proof (Lemma 2). We can assume without loss of generality that the first quantifier is $\oplus$, because we see how a $\exists$ quantifier is converted to a $\oplus$ below. We can also use the identities $\forall_x P(x) = \neg\exists_x \neg P(x)$ to transform the boolean expression.

Let's start with the base case, when there is only one quantifier. We are given $\varphi(x, y)$ and we need to determine if $\bigoplus_x \exists y : \varphi(x, y)$ is true nor not. Let $|x| = |y| = n$. For the moment, forget about x and focus on y. What can we do? We use Valiant-Vazirani (Lemma 1) to randomly produce a formula $\varphi'(x, y)$ such that if $\varphi(x, y)$ is satisfiable, then $\varphi'(x, y)$ is uniquely satisfiable with a probability at least $1/8n$, and unsatisfiable otherwise.

Suppose, the Valiant-Vazirani reduction were deterministic then the formula $\bigoplus_y : \varphi'(x, y)$ would be equivalent to $\exists y : \varphi(x, y)$ and therefore the $\oplus\mathsf{SAT}$ instance $\bigoplus_x \bigoplus_y : \varphi'(x, y)$ and $\bigoplus_x \exists y : \varphi(x, y)$ would also be equivalent.

Since the reduction is randomized and fails sometimes, we need to repeat the reduction several times to obtain $\varphi'(x, y)$ such that $\bigoplus_y : \varphi'(x, y)$ and $\exists y : \varphi(x, y)$ are equivalent with probability at least $1 - \frac{1}{6}2^{-n}$. Now, we can use the union bound to get:

$$\Pr\left[\text{For all } x,\ \bigoplus_y : \varphi'(x, y) \equiv \exists y : \varphi(x, y)\right] \geq \frac{5}{6}$$

[1] By the Cook-Levin theorem, we can construct such a boolean formula because the reductions all run in polynomial time.

therefore, in particular,

$$\Pr\left[\bigoplus_{x,y} : \varphi'(x,y) \equiv \bigoplus_{x} \exists y : \varphi(x,y)\right] \geq \frac{5}{6}$$

Thus, we are able to get rid of one level of quantification (probabilistically).

But, how do we construct φ' from φ. If we run the Valiant-Vazirani reduction $t = O(n^2)$ times independently, we produce formulae $\varphi'_1(x,y), \ldots, \varphi'_t(x,y)$. If $\varphi(x,y)$ is satisfiable, we know that at least one of the t formulae has a unique satisfying assignment with probability $\geq 1 - \frac{1}{6}2^{-n}$, and otherwise, none of them have a satisfying assignment.

Thus, given $\varphi'_1, \ldots, \varphi'_t$ produce a single φ' such that $\bigoplus_y : \varphi'(x,y)$ is true iff at least for one i, $\bigoplus_y : \varphi'_i(x,y)$ is true. For this, recall the construction in section 1.1

$$\left(\bigoplus_{x} \varphi(x)\right) \vee \left(\bigoplus_{y} \psi(y)\right) \Leftrightarrow \bigoplus_{x,y,z} ((\varphi + 1) \cdot (\psi + 1) + 1)(x,y,z)$$

Therefore, setting:

$$\varphi'(x,y) := 1 + \left(1 + \varphi'_1(x,y)\right) \cdots \left(1 + \varphi'_t(x,y)\right)$$

we construct φ' that has an odd number of satisfying assignments iff at least one of the φ'_i does.

This proves the result in the case there is one alternation. To extend this to c levels of alternation, we use the same argument to eliminate the outermost existential quantifier, and use the fact that $\psi \in \oplus\Pi_{k-1}\mathsf{SAT} \Leftrightarrow \overline{\psi} \in \oplus\Sigma_{k-1}\mathsf{SAT}$. Of course, this requires the probabilities to be arranged such that c repetitions of this succeeds with probability at least $2/3$. To do this, it suffices for us to succeed one step with a probability of at least $1 - 1/(6c^2)$, so that the reduction succeeds with probability:

$$1 - \sum_{c} \left(\frac{1}{6c^2}\right) \geq \frac{2}{3}.$$

∎

Proof (Lemma 3). For every pair of formulae φ and ψ recall the definitions $\varphi + \psi$ and $\varphi \cdot \psi$. Note that each of the constructions are of a size at most a constant factor larger than φ and ψ. Consider the formula $\psi^6 + 2\psi^3$ (where ψ^3, for example is $\psi \cdot (\psi \cdot \psi)$). One can easily check that:

$$\#(\psi) = -1 \pmod{2^{2^i}} \Rightarrow \#(\psi^6 + 2\psi^3) = -1 \pmod{2^{2^{i+1}}}$$
$$\#(\psi) = 0 \pmod{2^{2^i}} \Rightarrow \#(\psi^6 + 2\psi^3) = 0 \pmod{2^{2^{i+1}}}.$$

Therefore, if we set $\tau_0 = \varphi$ and $\tau_{i+1} = \tau_i^6 + 2\tau_i^3$, then after $\log m$ steps, we get that $\psi = \tau_{\log m}$ satisfies the two conditions in Lemma 3. The size of ψ is only polynomially larger than the size of φ (because at each stage the overhead is a constant factor). ∎

References

1. Toda, S.: Pp is as hard as the polynomial-time hierarchy. SIAM J. Comput. **20** (1991) 865–877
2. Valiant, L.G., Vazirani, V.V.: Np is as easy as detecting unique solutions. Theor. Comput. Sci. **47** (1986) 85–93
3. Bogdanov, A.: Lecture 8, lecture notes for 198:538. Rutgers University (2007)
4. Fortnow, L.: A simple proof of toda's theorem. Theory of Computing **5** (2009) 135–140
5. Barak, B., Arora, S.: Computational Complexity: A Modern Approach. (2009)