Some New Consequences of the Hypothesis That P Has Fixed Polynomial-Size Circuits

Ning Ding[1,2(✉)]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
dingning@sjtu.edu.cn
[2] NTT Secure Platform Laboratories, Tokyo, Japan

Abstract. We present some new consequences of the hypothesis that **P** can be computed by fixed polynomial-size circuits since [Lipton SCTC 94]. For instance, we show that the hypothesis implies that some small circuit family and BPP machines cannot be fooled by any complexity-theoretic pseudorandom generator $G : \{0,1\}^{\Theta(\log n)}$ to $\{0,1\}^n$, which means the known derandomization argument of $\mathbf{BPP} = \mathbf{P}$ no longer works. It also implies the existence of 2-round public-coin zero-knowledge proofs for **NP**.

1 Introduction

Proving non-uniform general circuit lower bounds for complexity classes is one of the most fundamental and challenging tasks in complexity theory. Let $\mathbf{SIZE}(n^c)$ denote the class of languages that can be determined by $O(n^c)$-size circuit families. Let $\mathbf{P}/poly = \cup_c \mathbf{SIZE}(n^c)$. With the notions of $\mathbf{SIZE}(n^c)$ and $\mathbf{P}/poly$, a typical lower bound result is of the form that some uniform class $\mathcal{C}$ cannot be compute by $\mathbf{SIZE}(n^c))$ or $\mathbf{P}/poly$.

For $\mathbf{P}/poly$ lower bounds, the best separation result we do know so far is the exponential-time version of Merlin-Arthur games is not in $\mathbf{P}/poly$ due to Buhrman *et al.* [3]. Karp and Lipton [13] showed that if $\mathbf{NP} \subset \mathbf{P}/poly$, the polynomial hierarchy collapses. However, currently we do not have any techniques for proving $\mathbf{NEXP} \not\subseteq \mathbf{P}/poly$. Williams [21] showed any algorithm for Circuit-SAT or for Circuit Acceptance Probability Problem slightly faster than exhaustive search implies $\mathbf{NEXP} \not\subseteq \mathbf{P}/poly$.

As for $\mathbf{SIZE}(n^c)$ lower bounds, Kannan [12] showed that $\Sigma_2 \cap \Pi_2 \not\subseteq \mathbf{SIZE}(n^c)$ for any constant c, Vinodchandran [20] showed $\mathbf{PP} \not\subseteq \mathbf{SIZE}(n^c)$ and Santhanam [17] showed $\mathbf{promiseMA} \not\subseteq \mathbf{SIZE}(n^c)$ for any $c \in \mathbb{N}$. When considering lower bounds for **P** and **NP**, however, currently the best known lower bound is $5n - o(n)$ due to Iwama and Morizumi [11].

After long-time failure to present non-linear lower bounds for **P**, some researchers thought possibly $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$. As mentioned in [16] Levin pointed out that Kolmogorov even believed $\mathbf{P} \subseteq \mathbf{SIZE}(n)$, and Lipton then investigated what can be implied if $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ and provided some interesting results e.g. $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ implies $\mathbf{NP} \neq \mathbf{P}$. Two decades passed since then and we still cannot prove or disprove the hypothesis.

R. Jain et al. (Eds.): TAMC 2015, LNCS 9076, pp. 75–86, 2015.
DOI: 10.1007/978-3-319-17142-5_8

Our Results. We continue the research of [16] by presenting some new consequences of the hypothesis $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$, some of which are about the topics emerging posterior to [16]. More concretely, our results are as follows.

Basic Consequences. If $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$, we have the following two conclusions (which are elementary but did not appear in literature to our knowledge).

1. $\mathbf{E} \subseteq \mathbf{SIZE}(2^{o(n)})$. It follows from this result that the assumption that $\mathbf{E}$ has a language that requires $2^{\Omega(n)}$ circuit lower bound is false. Recall that the known derandomization argument of $\mathbf{BPP} = \mathbf{P}$ in many works e.g. [10,18,19] requires this assumption. So our result means the known derandomization of $\mathbf{BPP} = \mathbf{P}$ no longer works under the hypothesis.
2. $\mathbf{BPP} \subseteq \mathbf{SIZE}(n^{c+\epsilon})$ for any constant $0 < \epsilon < 1$ if one-way functions exist.

$\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ vs Pseudorandom Generators. We show $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ implies the following negative results on complexity-theoretic pseudorandom generators $G : \{0,1\}^{l(n)=d\log n}$ to $\{0,1\}^n$ in polynomial-time for any $d \in \mathbb{N}$. This kind of generators is used to derandomize $\mathbf{BPP}$ in literature.

1. General such pseudorandom generators fooling small circuits do not exist. That is, there is no such G such that for all circuits D of size n, $|\Pr[D(G(U_{l(n)})) = 1] - \Pr[D(U_n) = 1]| \leq \frac{1}{n}$. Note that such generators are required in many works e.g. [10,18,19].
2. Some small circuit family $\{D_n\}_{n\in\mathbb{N}}$ is unfoolable against all G. That is, for each such G it holds for any constant $0 < \epsilon < 1$, $|\Pr[D_n(G(U_{l(n)})) = 1] - \Pr[D_n(U_n) = 1]| \geq \epsilon$ for infinitely many n. Note that this result is stronger than the first one.
3. Some BPP machines are unfoolable against all G if one-way functions exist. That is, for each $L \in \mathbf{BPP}$, there is a BPP machine M for L such that for each G there are instances x satisfying $M(x, G(U_{l(n)}))$ outputs wrong decisions with high probability (but in contrast $M(x, U_n)$ outputs wrong decisions with small probability).

The first result eliminates the existence of such general G which can fool all small circuits, but it does not eliminate the possibility that for any specific small circuit, there may exist a specific G which can fool the circuit (and may not fool other small circuits). However, the second result eliminates such possibility. Despite these two results, there is still a possibility that for each $L \in \mathbf{BPP}$ and some BPP machine M for L, there is such G such that we can derandomize M with G. The third result says for any $L \in \mathbf{BPP}$, some BPP machine for it cannot be derandomized by any G.

2-round Public-coin Zero-knowledge Proofs for NP. Zero-knowledge proofs [8] are of extreme importance in cryptography. Currently we have a 5-round construction in [6] and some impossibilities on fewer round numbers in [6,7,14]. There is no constant-round *public-coin* zero-knowledge proof for $\mathbf{NP}$ ever known. We show under the hypothesis there is a 2-round public-coin zero-knowledge proof for $\mathbf{NP}$. The simulator of the protocol is non-uniform. The non-triviality of such

a simulator is despite being non-uniform, it is able to simulate the interaction for all public inputs.

Then we present a witness-extractor for the protocol from program obfuscation, i.e. indistinguishability obfuscators recently proposed by e.g. [5,15], which can work for all bounded-size provers.

Our Techniques. Basically, the core technique in each consequence is to first define a problem/function and then show it is in **P** and thus gain an $O(n^c)$-size circuit family solving the problem which can then be used to establish the consequence. Here we sketch it in more detail with respect to unfoolable circuits against all pseudorandom generators.

Recall that our goal is to present some circuit family that can tell U_n from $G(U_{l(n)})$ for any G. So we first define a problem L_i: given an n-bit string r, decide if there is a string s with length $|s| \leq i \log n$ such that there is a G among the first n^i machines (in lexicographical order, say) within $n/2$-bit size such that $G(s)$ halts in n^i-time and $r = G(s)$. It can be seen $L_i \in \mathbf{P}$ for any i. So there is an $O(n^c)$-size circuit family $\{C_n^i\}_{n\in\mathbb{N}}$ determining $L_i, i \in \mathbb{N}$.

Let us investigate the output of C_n^i on input U_n or $G(U_{l(n)})$. First we can show C_n^i's output is almost always 0 when the input is U_n. On the other hand, for $G(U_{l(n)})$, for large enough i C_n^i can indeed output 1, indicating it can tell $G(U_{l(n)})$ from U_n. Lastly, we carefully choose such circuits over infinitely many n such that the circuit family can tell U_n from $G(U_{l(n)})$ for all G.

Organizations. Section 2 presents very short preliminaries. In Sects. 3 to 5 we present the consequences of the three parts respectively.

2 Preliminaries

Let $T : \mathbb{N} \to \mathbb{N}$ be some function. A language L is in $\mathbf{DTIME}(T(n))$ iff there is a Turing machine that runs in time $O(T(n))$ and determines L. Let $\mathbf{P} = \cup_{c\geq 1}\mathbf{DTIME}(n^c)$ and $\mathbf{E} = \cup_{c\geq 1}\mathbf{DTIME}(2^{cn})$.

Let $\mathbf{SIZE}(T(n))$ denote the class of languages satisfying for each L in it there is a circuit family $\{C_n\}_{n\in\mathbb{N}}$ such that $|C_n| = O(T(n))$ and for every $x \in \{0,1\}^n$, $x \in L \Leftrightarrow C_n(x) = 1$.

Let $L(x)$ denote the indicator function that outputs 1 if $x \in L$ and outputs 0 otherwise. Let **BPP** denote the class in which each language L admits a PPT machine M such that for each x, $\Pr[M(x) = L(x)] > \frac{1}{10}$ where the probability is taken over all choices of the coins of M. We call M a BPP machine for L.

3 Some Basic Consequences of $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$

3.1 E and $\mathbf{SIZE}(2^{o(n)})$

Theorem 1. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$, *then* $\mathbf{E} \subseteq \mathbf{SIZE}(2^{o(n)})$.

Proof. We use the padding argument to show this. Suppose $\mathbf{E}-\mathbf{SIZE}(2^{o(n)}) \neq \phi$ and L is a language in it. This means $L \in \mathbf{DTIME}(2^{c_1 n})$ for some $c_1 \in \mathbb{N}$, but

there exists $0 < \epsilon < 1$ such that L requires circuit lower bound $\Omega(2^{\epsilon n})$ for infinitely many n. Choose a sufficiently small constant δ satisfying $\epsilon/\delta > c$.

Consider the language L' that consists of all instances of form $x \circ 0^{2^{\delta n}-n}$ for $x \in L$ where $n \leftarrow |x|$. Then L' can be determined in $O(2^{\delta n \cdot c_1/\delta})$-time when inputs are of $2^{\delta n}$ bits. Thus by translation $L' \in \mathbf{DTIME}(n^{c_1/\delta}) \subseteq \mathbf{P}$. On the other hand, L' requires circuit lower bound $\Omega(2^{\epsilon n})$ when inputs are of $2^{\delta n}$ bits for infinitely many n. Thus by translation L' requires circuit lower bound $\Omega(n^{\epsilon/\delta})$ for infinitely many n and so it is not in $\mathbf{SIZE}(n^c)$. This is a contradiction. □

The theorem immediately asserts the following assumption is conditionally false which is used to establish the derandomization result $\mathbf{BPP} = \mathbf{P}$.

Assumption 2. $\mathbf{E}$ *has a language of deterministic circuit complexity* $2^{\Omega(n)}$.

Corollary 1. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$, *then Assumption 2 is false.*

3.2 BPP and SIZE($n^{c+\epsilon}$)

Theorem 3. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$ *and one-way functions exist, then* $\mathbf{BPP} \subseteq \mathbf{SIZE}(n^{c+\epsilon})$ *for any constant* $0 < \epsilon < 1$.

Proof. First if one-way functions exist, for any constant $0 < \delta < 1$ there exists a pseudorandom generator $G : \{0,1\}^{n^\delta} \to \{0,1\}^{\mathrm{poly}(n)}$ such that G is computable in time $\mathrm{poly}(n)$ and for all polynomial-size circuits D, $|\Pr[D(U_{\mathrm{poly}(n)}) = 1] - \Pr[D(G(U_{n^\delta})) = 1]| \leq 1/\mathrm{poly}(n)$ [9]. Thus for any $L \in \mathbf{BPP}$ and a BPP machine for L, there is another BPP machine for L which uses only n^δ coins: The machine first runs G with n^δ coins to get polynomial pseudorandom coins and then feeds the original BPP machine the pseudorandom coins to make decisions. Let M_1 denote such a BPP machine using n^δ coins with error $\frac{1}{n}$.

Let M denote a machine that runs M_1 $8n$ times independently and outputs the majority. Then there exists a specific value for all the coins used by M, denoted r_n, such that $M_{r_n}(x)$ outputs the correct decision for all $x \in \{0,1\}^n$ (as the proof of $\mathbf{BPP} \subset \mathbf{P}/poly$ shows). Note that $|r_n| = 8n^{1+\delta}$.

Now we define a language L_1 which consists of all instances (x, y) satisfying $M_y(x) = 1$. Thus $L_1 \in \mathbf{P}$. Then there is an $O(m^c)$-size circuit family $\{C_n\}_{n\in\mathbb{N}}$ deciding $m = |(x, y)| = O(n^{1+\delta})$-bit instances of L_1. Then we construct an $O(n^{(1+\delta)c})$-size circuit family $\{C'_n\}_{n\in\mathbb{N}}$ determining L. Actually, C'_n has r_n hardwired and on input x outputs $C_n(x, r_n)$. Since $C_n(x, r_n) = M_{r_n}(x)$ that equals the correct decision and $|C'_n| = O(n^{c+\epsilon})$ for $\epsilon = c\delta$, $L \in \mathbf{SIZE}(n^{c+\epsilon})$. □

4 $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ vs Pseudorandom Generators

In this section we investigate the relations between the hypothesis and complexity-theoretic pseudorandom generators. We focus on the polynomial-time generators $G : \{0,1\}^{\Theta(\log n)}$ to $\{0,1\}^n$, which are used to derandomize $\mathbf{BPP}$ and can result in $\mathbf{BPP} = \mathbf{P}$.

4.1 On General Pseudorandom Generators Fooling Small Circuits

Recall the derandomization argument of **BPP** in [10,18,19] that basically proceeds in two steps: first assume Assumption 2 to deduce Assumption 4 in the following is true; second use the pseudorandom generator G to derandomize any BPP machine for a language in **BPP**. Conversely, we also know Assumption 4 implies Assumption 2.

Assumption 4. *There exists a pseudorandom generator* $G : \{0,1\}^{l(n)} \to \{0,1\}^n$ *such that* G *maps inputs of length* $l(n) = \Theta(\log n)$ *to length* n *in time poly*(n)*, and for all circuits* D *of size* n, $|\Pr[D(G(U_{l(n)})) = 1] - \Pr[D(U_n) = 1]| \leq \frac{1}{n}$.

However, due to Corollary 1, we immediately have the following result.

Proposition 1. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$*, Assumption 4 is false.*

Proposition 1 eliminates the existence of such general G which can fool all small circuits. However, it does not eliminate the possibility that for any specific small circuit, there may exist a specific G which can fool the circuit (and may not fool other small circuits). So a further question is whether for each small circuit there is such a specific generator G that can fool it. In the next subsection, unexpectedly, we will answer this question negatively.

4.2 Unfoolable Circuit Families Against All Pseudorandom Generators

We now present a circuit family that cannot be fooled by any pseudorandom generator that stretches $\Theta(\log n)$-bit coins to n-bit pseudorandom coins.

Theorem 5. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$*, there is an* n^{c+1}*-size circuit family* $\{D_n\}_{n\in\mathbb{N}}$ *such that for any pseudorandom generator* G *that maps inputs of length* $l(n) = d \log n$ *for arbitrary* $d \in \mathbb{N}$ *to length* n *in time poly*(n)*, it holds for any constant* $0 < \epsilon < 1$, $|\Pr[D_n(G(U_{l(n)})) = 1] - \Pr[D_n(U_n) = 1]| \geq \epsilon$ *for infinitely many* n.

Proof. To present the circuit family $\{D_n\}_{n\in\mathbb{N}}$ such that for any generator G the result holds, we first define the following problems.

Problems L_i**.** For each $i \in \mathbb{N}$, we define problem L_i as follows. Given $r \in \{0,1\}^n$, decide if there is s of length no more than $i \log n$ such that for at least one machine G among the first n^i machines (in lexicographical order, say), G is at most $n/2$-bit long and $G(s)$ halts in n^i-time and r is equal to $G(s)$.

It can be seen that an exhaustive search algorithm can run each one of the first n^i machines at most n^i steps on input a string s of length no more than $i \log n$ and check if there are G and s satisfying the requirement. Since we only need to check the first n^i machines and emulate $G(s)$ n^i steps and the number of all s is $O(n^i)$, the algorithm can output a correct decision in polynomial-time. So L_i is in **P** for all $i \in \mathbb{N}$. Thus if $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$, there is an $O(n^c)$-size circuit family $\{C_n^i\}_{n\in\mathbb{N}}$ determining $L_i, i \in \mathbb{N}$.

Note that C_n^i is of $O(n^c)$-size. Set $k = c+1$, which means the size of $C_n^i, i \in \mathbb{N}$, is bounded by n^k for sufficiently large n. Let n_1 denote the least integer satisfying $|C_n^1| < n^k$ for each $n \geq n_1$ and for $i = 2, 3, \cdots$, let n_i denote the least integer satisfying $|C_n^i| < n^k$ for each $n \geq n_i$ and $n_i > n_{i-1}$. Define the following distinguisher $\{D_n\}_{n\in\mathbb{N}}$: for each $n \in \{n_1, \cdots, n_i, \cdots\}$, let D_n be $C_{n_i}^i$; for all other n, let it be any n^k-size circuit. Thus $|D_n| \leq n^k$.

For $\{D_n\}_{n\in\mathbb{N}}$, let us consider an arbitrary pseudorandom generator G which stretches $d \log n$ bits to n bits in polynomial-time for some d. It can be first seen that for all machines of length $n/2$ and all inputs s of length no more than $i \log n$, there are at most $\text{poly}(n) \cdot 2^{n/2}$ different outputs of all these machines with input s. So for truly random U_n, $U_n \notin L_i$ except for probability $\text{poly}(n)/2^{n/2}$ for any i. Thus $\Pr[D_n(U_n) = 1] = \text{poly}(n)/2^{n/2}$.

On the other hand, for this G, the order number of G in the enumeration of all machines is a constant and G's running-time is a fixed polynomial. When the input is $r = G(U_{d \log n})$, we have for each large enough i and for each $n \in \{n_i, n_{i+1}, \cdots\}$ the order number G is less than n^i and $G(s)$ outputs r for some s of length $d \log n$ (no more than $i \log n$) in n^i-time, which shows $C_{n_i}^i(r)$ outputs 1 always. Thus $|\Pr[D_n(G(U_{l(n)})) = 1] - \Pr[D_n(U_n) = 1]| \geq \epsilon$ for infinitely many n for any constant $0 < \epsilon < 1$. The theorem holds. □

4.3 Unfoolable BPP Machines Against All Pseudorandom Generators

The previous subsections show under the hypothesis, not only the general pseudorandom generator for all small circuits, but also specific generators for all specific circuits do not exist. But for the purpose of derandomizing **BPP**, both the two-type generators are not necessary. Actually, a specific pseudorandom generator that can fool a specific BPP machine for any language in **BPP** suffices. More precisely, let $L \in \mathbf{BPP}$ and M be a BPP machine for L. A pseudorandom generator G satisfying for any instance x, $G(U_{l(n)})$ can fool M with x suffices to induce a deterministic polynomial-time machine for L. Since intuitively $M(x)$ could not be the $\{D_n\}_{n\in\mathbb{N}}$ in Theorem 5, there is a possibility that for each $L \in \mathbf{BPP}$ and some M for L, there is such G such that we can derandomize M with G.

However, to do this we need to select a derandomizable one instead of any BBP machine for L, since the following theorem says that some M for L cannot be derandomized by any generator G, in the sense that on one hand M with truly random coins can decide all instances correctly with high probability and on the other hand M with pseudorandom coins from any G will output wrong decisions for some instances with high probability. When errors occur, we cannot be aware of this.

In the following for any BPP machine M, we use notation $M(x, U_{\text{poly}(n)})$ to denote the computation of M with input instance x and coins $U_{\text{poly}(n)}$.

Theorem 6. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$ *and one-way functions exist, then for all* $L \in \mathbf{BPP}$ *there is a BPP machine* M *for* L *which needs no more than* n^k *coins for* $k \in \mathbb{N}$ *such that for any pseudorandom generator* G *that maps*

inputs of length $l(n) = d \log n$ *for arbitrary* $d \in \mathbb{N}$ *to length* n^k *in time poly*(n)*, there is an instance serial* $\{x_n\}_{n\in\mathbb{N}}$ *satisfying* $\Pr[M(x_n, G(U_{l(n)})) \neq L(x_n)] \geq 1 - \frac{poly(n)}{2^n} - \frac{1}{n}$ *for all sufficiently large* n*.*

Proof. Let M_L be a BPP machine for L with error $\epsilon = \frac{1}{n}$ which uses no more than n coins (using a cryptographically pseudorandom generator constructed from one-way functions to generate poly(n) coins). We construct a BPP machine M for L that uses n^k coins and cannot be derandomized. On input any instance $x \in \{0,1\}^{n^{c+1}}$ and coins U_{n^k}, M does the following.

1. If x cannot be parsed to the form (C, r, r') where C denotes a boolean circuit of n-bit input and $|C| = n^{c+1/2}$ and $|r| = n$ and $|r'| = n$, output $M_L(x, r' \oplus r_1)$ where r_1 denotes n coins in U_{n^k}. Otherwise, move to the next step.
2. Set $t = n^3$ and let $r_1, \cdots, r_t, r_{t+1}, r_{t+2}$ be the first $t+2$ n-bit blocks in U_{n^k}. Compute $C(r_1), \cdots, C(r_t)$ and count the fraction of 1 among all outputs. If the fraction is less than $1-\epsilon$, output $M_L(x, r' \oplus r_{t+2})$. Otherwise, output $M_L(x, r' \oplus r_{t+2})$ if $C(r \oplus r_{t+1}) = 1$ and output $1 - M_L(x, r' \oplus r_{t+2})$ otherwise.

We now show M is indeed a BPP machine for L. First consider x that is not of form (C, r, r'). Then M outputs $M_L(x, U_n)$. Thus it has error ϵ. Second consider $x = (C, r, r')$ with $\Pr[C(U_n) = 1] < 1 - 2\epsilon$. Due to the Chernoff bound, $\frac{1}{t}\sum_{i=1}^{t} C(r_i) < 1 - 2\epsilon + \delta < 1 - \epsilon$ except for probability $e^{-2\delta^2 t} = e^{-2n}$ for $\delta = \frac{1}{n}$. This shows M's error is at most $\epsilon + e^{-2n}\epsilon < 2\epsilon$. Third consider $x = (C, r, r')$ with $\Pr[C(U_n) = 1] \geq 1 - 2\epsilon$. Then M's error is at most $\epsilon + \Pr[C(U_n) = 0] + \epsilon < 4\epsilon$. So for any instance M's error is at most 4ϵ. That shows M is a BPP machine for L.

Consider an arbitrary G that maps inputs of length $d \log n$ to length n^k in time poly(n). We now define the following function.

Function f**.** Given $r \in \{0,1\}^n$, output 1 if there is $s \in \{0,1\}^{d \log n}$ such that r equals any one of the first n^3 n-bit blocks in the output of $G(s)$, and output 0 otherwise.

Similarly, viewed as a language, $f^{-1}(1)$ is in **P**. Thus there is an $O(n^c)$-size circuit family $\{C_n\}_{n\in\mathbb{N}}$ computing f. Note that $|C_n| < n^{c+1/2}$ and can be padded to $n^{c+1/2}$-size for large enough n. Similarly, $G(s)$ has at most poly(n) different outputs, one of which happens to contain U_n as a block with probability $\frac{\text{poly}(n)}{2^n}$. Thus $\Pr[C_n(U_n) = 1] = \frac{\text{poly}(n)}{2^n}$.

First consider the instance $x_n = (C_n, r, r')$ for uniformly random r, r'. When the coins for M is $G(U_{d \log n})$, letting $r_1, \cdots, r_{t+1}$ denote the first $n^3 + 1$ n-bit blocks in the output of $G(U_{d \log n})$, we have $C_n(r_1) = \cdots = C_n(r_t) = 1$ and $\Pr[C_n(r \oplus r_{t+1}) = 0] = 1 - \frac{\text{poly}(n)}{2^n}$ since r is uniformly random. Due to M's strategy, $M(x_n, G(U_{d \log n}))$ outputs $1 - M_L(x_n, r' \oplus r_{t+2})$ almost all the time. Thus $\Pr[M(x_n, G(U_{d \log n})) \neq L(x_n)] \geq 1 - \frac{\text{poly}(n)}{2^n} - \epsilon$. Thus there exist specific r, r' such that fixing $x_n = (C_n, r, r')$, the probability formula still holds. The theorem holds. □

We remark that even in the case $\mathbf{BPP} = \mathbf{P}$, L which admits a deterministic polynomial-time machine still admits a BPP machine that cannot be derandomized. Actually with a similar argument to that of Theorem 6 (where M_L changes to be a deterministic polynomial-time machine for L and consider x of form (C, r)) we have the following proposition which does not need one-way functions and achieves stronger probability result.

Proposition 2. *If* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$, *then for all* $L \in \mathbf{P}$ *there is a BPP machine* M *for* L *which needs no more than* n^k *coins for* $k \in \mathbb{N}$ *such that for any pseudorapndom generator* G *described in Theorem 6, there is an instance serial* $\{x_n\}_{n\in\mathbb{N}}$ *satisfying* $\Pr[M(x_n, G(U_{l(n)})) \neq L(x_n)] \geq 1 - \frac{poly(n)}{2^n}$ *for all sufficiently large* n.

5 Two-Round Public-Coin Zero-Knowledge Proofs

In this section we investigate the question of constructing constant-round public-coin zero-knowledge proofs for **NP** if $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$. An interactive proof is zero-knowledge if for any polynomial-time verifier there is a polynomial-time simulator such that what the verifier sees, i.e. random coins, the public input and prover's messages, can be computationally indistinguishably reconstructed by the simulator [8].

Currently we have a 5-round private-coin construction due to [6] and some impossibilities on fewer round numbers in e.g. [6,7,14]. Reference [2] presents a negative result on 2-round public-coin zero-knowledge proofs, but it assumes that **E** has a language of non-deterministic circuit complexity $2^{\Omega(n)}$, which is even stronger than Assumption 2. So due to the hypothesis $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$, this assumption is false and the negative result in [2] no longer works.

So there is no constant-round public-coin zero-knowledge proofs for **NP** ever known. However, we show that based on the hypothesis there exists a 2-round public-coin zero-knowledge proof for **NP** with respect to a relaxed requirement that the simulator can be non-uniform. Despite being non-uniform the simulator is able to simulate the interaction for all public inputs.

5.1 The Protocol

We first present some preparations as follows.

Definition 1. *For each polynomial-time machine* M, *we define a function* f_M *as follows. Given* $x \in \{0,1\}^n, u \in \{0,1\}^n, i \in [1, n^{c+2}]$, *output* r_i *that is the* i*th bit of* $r \leftarrow M(x, u)$.

Note that in the definition i can be represented by a $\lceil (c+2)\log n \rceil$-bit string. The function f_M induces a problem L_M that consists all instances (x, u, i) satisfying $f_M(x, u, i) = 1$. Since M is polynomial-time, $L_M \in \mathbf{P}$. Due to the hypothesis, L_M can be determined by an $O(n^c)$-size circuit family $\{C_n\}_{n\in\mathbb{N}}$. Namely, f_M can be computed by $\{C_n\}_{n\in\mathbb{N}}$.

Let L be any language in **NP**. Then we define the following language Λ.

Public input: x;
Prover's auxiliary input: w, (a witness for $x \in L$).

1. $V \to P$: Send $r \in_{\mathrm{R}} \{0,1\}^{n^{c+2}}$, ZAP_1.
2. $P \to V$: Send ZAP_2 generated using witness w for the statement that $(x, r) \in \Lambda$.

Protocol 1 *The 2-round public-coin zero-knowledge proof for L.*

Definition 2. *We define the following language Λ: $(x, r) \in \Lambda$ where $|x| = n, |r| = n^{c+2}$ iff either there is a witness w for $x \in L$ or there are a boolean circuit C of size at most n^{c+1} and $u \in \{0,1\}^n$ such that $C(x, u, i) = r_i$ for all $1 \leq i \leq n^{c+2}$.*

Then $\Lambda \in \mathbf{NP}$ and a witness for $(x, r) \in \Lambda$ is either w for $x \in L$ or a circuit C and u satisfying the second condition.

Let ZAP denote the 2-round public-coin witness-indistinguishable (WI) proof for $\mathbf{NP}$ in [4], $(\mathsf{ZAP}_1, \mathsf{ZAP}_2)$ denote the two messages of ZAP. Let PRG denote a cryptographically pseudorandom generator in [9]. Our protocol for L is shown in Protocol 1.

Theorem 7. *Assuming $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ for some $c \in \mathbb{N}$ and the existence of ZAP, PRG, Protocol 1 is a 2-round public-coin zero-knowledge proof for L.*

Proof. We show the completeness, soundness and zero-knowledge properties are satisfied.

Completeness. For $x \in L$ P can always convince V using w.

Soundness. For each $x \notin L$ and all possible n^{c+1}-size boolean circuits C and $u \in \{0,1\}^n$, the string of $C(x, u, 1) \circ \cdots \circ C(x, u, n^{c+2})$ in which "$\circ$" means concatenation has at most $2^{n^{c+1}+n}$ different values. Now r is randomly chosen from $\{0,1\}^{n^{c+2}}$. So one of these values equals r with probability $2^{-\Omega(n^{c+2})}$, which shows $(x, r) \notin \Lambda$ with probability $1 - 2^{-\Omega(n^{c+2})}$, i.e. the statement that ZAP proves is false. Thus the soundness follows from the soundness of ZAP.

Zero-Knowledge. For each PPT verifier V^*, we present a polynomial-size simulator S which is constructed as follows.

1. Consider the following machine M. On input (x, u), $M(x, u)$ runs $V^*(x)$ and when V^* needs random coins, run $\mathsf{PRG}(u)$ and provide the output to it. For this machine M, let f_M and L_M be defined previously. Then there is an $O(n^c)$-circuit family $\{C_n\}_{n \in \mathbb{N}}$ computing f_M.
2. Sample coins $u \in \{0,1\}^n$. Let S have V^*, C_n, u hardwired. $S(x)$ runs as follows. It runs $V^*(x)$ to output $r \in \{0,1\}^{n^{c+2}}$ and ZAP_1 in which when V^* needs coins, run $\mathsf{PRG}(u)$ and provide the pseudorandom coins to it. Then S computes ZAP_2 using witness (C_n, u) and sends it to V^*.

We first show that (C_n, u) is a witness for $(x, r) \in \Lambda$. It can be seen that r is the output of $V^*(x)$ with coins from $\mathsf{PRG}(u)$. This means $r = M(x, u)$. Due to the definition of C_n, we have $C_n(x, u, i) = r_i$ for all $1 \leq i \leq n^{c+2}$. And $|C_n| < n^{c+1}$ for large enough n. This shows (C_n, u) is a witness for $(x, r) \in \Lambda$. So S can finish the interaction.

Then we show S can reconstruct indistinguishably V^*'s view (random tape, prover's messages). Since V^*'s coins are now $\mathsf{PRG}(u)$ and S differs from $P(w)$ only in the witnesses they use, the indistinguishability is ensured by the pseudo-randomness of PRG and WI of ZAP. The zero-knowledge property holds. □

5.2 Obtaining Witness Extraction from Program Obfuscation

In this subsection we consider an enhanced property of witness extraction, which claims an extractor E such that for any polynomial-time prover P' that can convince V some $x \in L$, then $E(P', x)$ can output a witness for $x \in L$ in polynomial-time. A proof system admitting an extractor is called a proof of knowledge in cryptography. Our result is that we present a witness extractor from program obfuscation for Protocol 1 which works for bounded-size provers. For lack of space, we only sketch the construction.

Informally a program obfuscator is a PPT algorithm that given a program can output a new program such that the output program is of same functionality as the input program but hides some secrets. In particular, an indistinguishability obfuscator, denoted $i\mathcal{O}$, which was first introduced by [1] and which candidate constructions were recently proposed by [5,15] etc. is such that for any two machines (M_1, M_2) of same functionality (and same size and same running-time), $i\mathcal{O}(M_1)$ and $i\mathcal{O}(M_2)$ are computationally indistinguishable. We will employ $i\mathcal{O}$ to achieve our result.

We modify Protocol 1 with $i\mathcal{O}$. That is, we let P send a random $r_1 \in \{0,1\}^{\mathrm{poly}(n)}$ for a sufficiently large poly(n) (e.g. $\geq n^{c+3}$) and $\widetilde{Q}_1 \leftarrow i\mathcal{O}(Q_1)$ in Step 2, where Q_1 denotes the program that on input a program Π with $|\Pi| < |r_1|/2$ outputs w for $x \in L$ if Π outputs r_1 within $n^{\log\log n}$ steps and outputs 0^n otherwise. And accordingly, the first condition in Definition 2 changes to that $\widetilde{Q}_1$ is honestly generated. The modified protocol is shown in Protocol 2.

It can be seen that Protocol 2 is complete and sound. Moreover, the simulator S needs slight modification. That is, it samples r_1 and computes $\widetilde{Q}_2 \leftarrow i\mathcal{O}(Q_2)$ in Step 2, where Q_2 is equal to Q_1 except that it always outputs 0^n, and computes ZAP_2 as before. Note that Q_1, Q_2 are of same functionality except on input a program Π satisfying Π outputs r_1. However, for random r_1, since $|\Pi| < |r_1|/2$, the Π does not exist except for exponentially small probability. Thus the two programs are of same functionality and thus $\widetilde{Q}_1, \widetilde{Q}_2$ are indistinguishable. So the zero-knowledge property still holds.

Finally, let us sketch the construction of the extractor. Actually, as shown in the soundness, if some prover P' can convince V $x \in L$, then due to the soundness of ZAP, $\widetilde{Q}_1$ is honestly generated. If P''s size is bounded by $|r_1|/2$, basically its code is a valid input Π such that $\widetilde{Q}_1(\Pi)$ outputs w. So an extractor E can adopt V's strategy to send the message of Step 1 and emulates P''s computation where

Public input: x;
Prover's auxiliary input: w, (a witness for $x \in L$).

1. $V \to P$: Send $r \in_{\mathrm{R}} \{0,1\}^{n^{c+2}}$, ZAP_1.
2. $P \to V$: Send $r_1 \in_{\mathrm{R}} \{0,1\}^{\mathrm{poly}(n)}, \widetilde{Q}_1, \mathsf{ZAP}_2$.

Protocol 2 *The 2-round public-coin zero-knowledge proof of knowledge for L.*

providing P' pseudorandom coins from $\mathsf{PRG}(u')$ for random $u' \in \{0,1\}^n$. Thus P''s code, u' and PRG constitute a valid Π which size is bounded. On receiving P''s message, E runs $\widetilde{Q}_1(\Pi)$ to gain w. Thus we have the following result.

Theorem 8. *Assuming* $\mathbf{P} \subseteq \mathbf{SIZE}(n^c)$ *for some* $c \in \mathbb{N}$ *and the existence of* $\mathsf{ZAP}, \mathsf{PRG}, i\mathcal{O}$*, Protocol 2 is a 2-round public-coin zero-knowledge proof for L which admits an extractor for all bounded-size provers* ($< \frac{|r_1|}{2}$).

Acknowledgments. The author is grateful to the reviewers of TAMC 2015 for their detailed and useful comments. This work is supported by the National Natural Science Foundation of China (Grant No. 61100209) and Doctoral Fund of Ministry of Education of China (Grant No. 20120073110094).

References

1. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. J. ACM **59**(2), 6 (2012)
2. Barak, B., Lindell, Y., Vadhan, S.P.: Lower bounds for non-black-box zero knowledge. J. Comput. Syst. Sci. **72**(2), 321–391 (2006)
3. Buhrman, H., Fortnow, L., Thierauf, T.: Nonrelativizing separations. In: IEEE Conference on Computational Complexity, pp. 8–12. IEEE Computer Society (1998)
4. Dwork, C., Naor, M.: Zaps and their applications. In: FOCS, pp. 283–293. IEEE Computer Society (2000)
5. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS, pp. 40–49. IEEE Computer Society (2013)
6. Goldreich, O., Kahan, A.: How to construct constant-round zero-knowledge proof systems for np. J. Cryptol. **9**(3), 167–190 (1996)
7. Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. J. Cryptol. **7**(1), 1–32 (1994)
8. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. **18**(1), 186–208 (1989)
9. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)

10. Impagliazzo, R., Wigderson, A.: P = BPP if e requires exponential circuits: derandomizing the xor lemma. In: Leighton, F.T., Shor, P.W. (eds.) STOC, pp. 220–229. ACM (1997)
11. Iwama, K., Morizumi, H.: An explicit lower bound of $5n$-$o(n)$ for boolean circuits. In: Diks, K., Rytter, W. (eds.) MFCS 2002. LNCS, vol. 2420, pp. 353–364. Springer, Heidelberg (2002)
12. Kannan, R.: Circuit-size lower bounds and non-reducibility to sparse sets. Inf. Control **55**(1–3), 40–56 (1982)
13. Karp, R.M., Lipton, R.J.: Some connections between nonuniform and uniform complexity classes. In: Miller, R.E., Ginsburg, S., Burkhard, W.A., Lipton, R.J. (eds.) STOC, pp. 302–309. ACM (1980)
14. Katz, J.: Which languages have 4-round zero-knowledge proofs? In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 73–88. Springer, Heidelberg (2008)
15. Koppula, V., Lewko, A.B., Waters, B.: Indistinguishability obfuscation for turing machines with unbounded memory. Cryptology ePrint Archive, Report 2014/925 (2014). http://eprint.iacr.org/
16. Lipton, R.J.: Some consequences of our failure to prove non-linear lower bounds on explicit functions. In: Structure in Complexity Theory Conference, pp. 79–87. IEEE Computer Society (1994)
17. Santhanam, R.: Circuit lower bounds for merlin-arthur classes. In: Johnson, D.S., Feige, U. (eds.) STOC, pp. 275–283. ACM (2007)
18. Shaltiel, R., Umans, C.: Simple extractors for all min-entropies and a new pseudo-random generator. In: FOCS, pp. 648–657. IEEE Computer Society (2001)
19. Umans, C.: Pseudo-random generators for all hardnesses. J. Comput. Syst. Sci. **67**(2), 419–440 (2003)
20. Vinodchandran, N.V.: A note on the circuit complexity.In: Electronic Colloquium on Computational Complexity (ECCC) (056) (2004)
21. Williams, R.: Improving exhaustive search implies superpolynomial lower bounds. In: Schulman, L.J. (ed.) STOC, pp. 231–240. ACM (2010)