**03684155: On the P vs. BPP problem.** 11/12/16 – Lecture 9

# An algebraic NW generator

*Amnon Ta-Shma and Dean Doron*

In previous lectures we saw that a uniform *Boolean* function hard for a *non-uniform* class gives rise to a pseudo-random generator against that class. We (presenting work of [1]) now adopt the "hardness implies pseudorandomness" approach to the arithmetic world, and show how a uniform polynomial that is hard against a non-uniform arithmetic class (for example, if the permanent is hard for polynomial-size arithmetic circuits) implies a HSG against low-degree, small-size arithmetic circuits, and hence a derandomization of PIT. Thus, arithmetic circuits take the place of boolean circuits, a hard polynomial (say, the permanent) takes the place of a hard function, and we try to fool arithmetic circuits (or PIT) instead of trying to fool BPP.

**Definition 1.** *Let $\Lambda$ be some subset of $m$-variate polynomials over $\mathbb{F}$. A set $S \subseteq \mathbb{F}^m$ is a HSG for $\Lambda$ if for all $f \in \Lambda$ there exists $s \in S$ such that $f(s) \neq 0$.*

In this lecture we will an arithmetic version of the NW generator, converting an explicit polynomial hard against small-size arithmetic circuits to a HSG against such circuits.

## 1 Finding roots of multivariate polynomials

Arithmetic circuits are the algebraic analogue of boolean circuits, and polynomials are the arithmetic analogue of boolean functions. In the boolean world there are $2^{2^n}$ boolean functions on $n$ bits, but only $s^{O(s)}$ boolean circuits of size $s$. Similarly, the vector space of all multi-variate polynomials in $n$ variables and total degree $d$ has dimension $\binom{n+d}{d}$, which we approximate by $n^d$, and the number of such polynomials over a finite field is exponential in that, namely, about $2^{n^d}$. Even if we just look at multi-linear polynomials over $\mathbb{F}_2$ we have $2^{2^n}$ such functions.

Thus, a multi-linear polynomial is not an efficient description of a function. In fact, we can choose to describe a polynomial by its coefficients or by its evaluations on points in general position, and both representations take the same number of bits, so it is correct to think of the coefficients representation as a truth-table representation. If you find it confusing, it is because we often work with uni-variate polynomials, and then the coefficients representation is efficient. The (few) polynomials that have a small arithmetic circuit are the "easy" polynomials, the same way the (few) boolean functions that have small circuits are easy and the rest are hard.

Now, let us consider factoring multi-variate polynomials. We already used the fact that we can factor uni-variate polynomials (for decoding RS codes) and bi-variate polynomials (for list-decoding RS codes) in BPP. Now, we would like to factor multi-variate polynomials. Because of the discussion before, we take the input to the problem to be a polynomial-size arithmetic circuit, representing an "easy" multi-variate polynomial. Our goal is to factorize it efficiently, i.e., we would like to output a small arithmetic circuit for each of its factors.

**Definition 2** (Root-finding problem)**.** *Given an arithmetic circuit $C(x_1, \ldots, x_n, y)$ computing a non-zero polynomial of total degree $d$ over a field $\mathbb{F}$, find a list of arithmetic circuits such that for every polynomial $p(x_1, \ldots, x_n)$ satisfying $C(x_1, \ldots, x_n, p(x_1, \ldots, x_n)) \equiv 0$, there is a circuit on the list that agrees with $p$ over $\mathbb{F}^n$.*

But now we need to pause and ask whether this is possible at all. Namely, can it happen that an easy multi-variate polynomial has a hard factor, or several hard factors? The remarkable answer is: No! An easy polynomial has easy factors and these can be found efficiently. This is captured in the remarkable:

**Theorem 3.** *The root-finding problem for a polynomial $g$ of total degree $d$ computable by an arithmetic circuit of size $s$ can be solved probabilistically in time* $\mathrm{poly}(s, d, \log|\mathbb{F}|)$ *when $\mathbb{F}$ is a finite field. For $\mathbb{F} = \mathbb{Q}$, the running time is* $\mathrm{poly}(s, d, a)$*, where $a$ is the maximum coefficient size of $g$.*

The result is a corollary of Kaltofen's randomized factorization algorithm [2], and will be central for the algebraic version of the NW generator.

## 2 The NW arithmetic generator

We apply the NW generator in the arithmetic setting, replacing the hard boolean function, with a hard arithmetic polynomial. We are given parameters $m, \ell, a \in \mathbb{N}$, such that:

- $p \in \mathbb{F}[x_1, \ldots, x_\ell]$,
- $S_1, \ldots, S_m \subseteq [t]$ is an $(\ell, a)$ design, where $t = O(\frac{\ell^2}{a})$.

The generator $G^p : \mathbb{F}^t \to \mathbb{F}^m$ is given by:

$$G^p(y) \;=\; p(y|_{S_1}), \ldots, p(y|_{S_m}).$$

We want to claim that if $p$ is hard (i.e., requires a large arithmetic circuit) then $IMAGE(G^p)$ is a *hitting set generator* against all low-degree polynomials.

**Theorem 4.** *Let $\mathbb{F}$ be a finite field, $B \subseteq \mathbb{F}$. Let $p \in \mathbb{F}[y_1, \ldots, y_\ell]$ be a degree $d_p$ polynomial with* $\mathsf{SIZE}(p) \geq s_p$*. Let $d$ be an integer and suppose $|B| > d \cdot d_p$. Then $NW^p : B^t \to \mathbb{F}^m$ is a HSG for $SizeDegree(s, d)$ polynomials, for $s = s_p^\alpha - O(m \cdot d \cdot d_p)$ for some constant $\alpha > 0$.*

*Proof.* Suppose not. Then exists $f \in \mathbb{F}[x_1, \ldots, x_m]$ of degree $d$ and size $s$ such that $f \neq 0$ but $f(G(y)) = 0$ for all $y \in \mathbb{F}^t$. However, $|B| > dd_p \geq \deg(f \circ G)$ hence by Schwartz-Zippel $f \circ G \neq 0$ in $\mathbb{F}[y_1, \ldots, y_t]$.

We define a hybrid of polynomials bridging between $f$ and $f \circ G$ as follows. Let $z_i(y_1, \ldots, z_t) = p(y|_{S_i})$. Then

$$g_i(y_1, \ldots, y_t, x_1, \ldots, x_m) = f(z_1, \ldots, z_i, x_{i+1}, \ldots, x_m).$$

That is $g_0 = f$ and $g_m = f \circ G$.

Clearly,

1. $\deg(g_i) \leq deg(f \circ G) \leq d_f d_p$.
2. $g_0 \neq 0$ but $g_m = 0$ in $\mathbb{F}[y_1, \ldots, y_t, x_1, \ldots, x_m]$, so there must exist a smallest $i$ such that $g_i \neq 0$ but $g_{i+1} = 0$.

3. Since $g_i(y_1, \ldots, y_t, x_1, \ldots, x_m) \neq 0$, by the Schwartz-Zippel lemma there is a way fix all variables with constants from $B$ and get a non-zero value. In particular there is a way to fix the variables $y_j$ for $j \notin S_i$ to some field elements so that the restricted polynomial $\tilde{g}_i(y_{j_1}, \ldots, y_{j_\ell}, x_{i+1})$ remains non-zero. Denote this new polynomial by $g(y_{j_1}, \ldots, y_{j_\ell}, x)$.

We know that $g$ is nonzero, however $g(y_{j_1}, \ldots, y_{j_\ell}, p(y_{j_1}, \ldots, y_{j_\ell})) = 0$. Therefore, by Theorem 3,

$$\mathsf{SIZE}(p) \leq \text{poly}(\mathsf{SIZE}(g), \deg(g) = dd_p, \log |B|).$$

A circuit for $g$ can be obtained by plugging values into a circuit computing $f$, where the values are either $p(y|_{S_j})$ or $x_{j'}$, taking the right constant where we have a variable that was fixed. Thus, there required size at most $\mathsf{SIZE}(f)$ plus $m$ times what is needed for a computation of $p$ on a restriction $y|_{S_j}$ after setting all variables outside $S_i$. Each such restriction has is a polynomial of degree at most $d_p$ on at most $a$ variables. Every such polynomial has at most $M = (d_p + 1)^a$ monomials, and so can be computed by a circuit of size poly$(M)$. Thus, $\mathsf{SIZE}(g) \leq \mathsf{SIZE}(f) + m\text{poly}(M)$. Altogether,$\mathsf{SIZE}(p) \leq (s + m(d_p + 1)d + \log B)^c$ for some constant $c$. Finally, take $b$ just larger than $dd_p$. □

# 3 Algebraic hardness-randomness tradeoffs

We first define $AIT$, the algebraic analogue of PIT.

**Definition 5** (Arithmetic circuit identity testing)**.** *Given an arithmetic circuit $C$ computing a polynomial $p(x_1, \ldots, x_n)$, decide whether $p \equiv 0$.*

**Theorem 6.** *We assume a uniform family of hard arithmetic functions. Specifically, let $p = \{p_\ell\}$ be a family such hat:*

- *$p_\ell$ is a multilinear non-zero $\ell$-variate polynomials over $\mathbb{Z}$.*
- *(uniform family) $\{p_\ell(x)\}$ can be computed in exponential time in the input length, and the maximum coefficient size of $p_\ell$ over $\mathbb{Z}$ is at most* poly$(\ell)$.
- *(hard function) The arithmetic circuit complexity of $p$ over $\mathbb{Q}$ is $s_p(\ell)$ for some function $s_p$.*

*Let $C$ be an arithmetic circuit over $\mathbb{Z}[x_1, \ldots, x_m]$ with:*

- poly$(m)$*-sized, and maximum coefficient size at most* poly$(m)$.
- *$d_f(m) =$* poly$(m)$ *degree.*

*Then, for all large enough $m$, testing wether $C \equiv 0$ can be done deterministically in time:*

1. *$2^{m^\varepsilon}$ for any constant $\varepsilon > 0$, if $s_p(\ell) = \ell^{\omega(1)}$.*
2. *$2^{\text{poly}(\log m)}$ if $s_p(\ell) = 2^{\ell^{\Omega(1)}}$.*

*Proof.* We will only prove (1), and the proof of (2) is similar. We set $\ell = m^{\varepsilon}$, so $t \leq m^{2\varepsilon}$. We let $B \subseteq \mathbb{Z}$ of size at most $d_p d_f < m^{\varepsilon} m = \text{poly}(m)$ (of, say, the smallest integers). To solve arithmetic circuit identity testing, enumerate all elements $y \in B^t$, compute the output $r = G^p_{\ell,n}(y)$ and then evaluate $C(r)$. We output "$C$ is nonzero" iff $C(r) \neq 0$ for some $r$.

For correctness, by Theorem 4, the test succeeds since otherwise $s_p(\ell) = \text{poly}(\ell)$, in contradiction to the fact that $s_p(\ell) = \ell^{\omega(1)} = m^{\omega(1)}$.

For the running time, the size of $B^t$ is at most $m^{O(t)} < 2^{m^{3\varepsilon}}$. $p$ is computable in exponential time, an altogether the running time of $G^p_{t,m}$ on $B^t$ takes time $|S^t| 2^{\ell^c} < 2^{m^{4c\varepsilon}}$ for some fixed constant $c$. Choosing $\varepsilon$ small enough the derandomization runs in $\mathsf{SUBEXP}$. (We also need the assumption on the degree and coefficient size of $p_\ell$, to be able to simulate the arithmetic circuit in the Boolean world, but we leave out the details). □

The above result can be extended to the case of finite fields. The only caveat is that $\mathbb{F}$ may be too small to choose a poly$(m)$-sized set $B$. However, since we test wether $C_m$ is the identically zero polynomial, we can work with sufficiently large extension fields. We only need $O(\log m)$-degree extensions of $\mathbb{F}$, and such an extension can be found by brute-force in time poly$(n)$ (in fact, there are efficient non-brute force ways to do that, but we do not need it).

Note that unlike the boolean case, the assumption of $2^{\ell^{\Omega(1)}}$ circuit complexity does not imply a *polynomial*-time derandomization procedure, but only a quasi-polynomial derandomization, as we still need to enumerate all $O(\log m)$-tuples of field elements of bit-size $O(\log m)$ each.

We conclude with stating a *weak* converse of our theorem from the previous lecture, that if PERM $\in$ $\mathsf{AP/poly}$ and $\mathsf{NEXP} \subseteq \mathsf{P/poly}$ then PIT $\notin \mathsf{P}$.

**Theorem 7.** *If either* $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$ *or* PERM $\notin \mathsf{AP/poly}$*, then arithmetic circuit identity testing for size $n$ circuits computing polynomials of degree and maximum coefficient size at most* poly$(n)$ *is in* $\mathsf{io\text{-}NTIME}(2^{n^{\varepsilon}})/n^{\varepsilon}$ *for every constant $\varepsilon > 0$.*

The case of PERM $\notin \mathsf{AP/poly}$ follows from 6, even without the advice. The case of $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$ follows from the fact that if indeed $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$ then *co*$\mathsf{RP} \subseteq \mathsf{io\text{-}NTIME}(2^{n^{\varepsilon}})/n^{\varepsilon}$ for every constant $\varepsilon > 0$ – a fact we will not prove.

## References


[1] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

[2] Erich Kaltofen. Factorization of polynomials given by straight-line programs. *Randomness and Computation*, 5:375–412, 1989.