COSC 544 Probabilistic Proof Systems 11/21/17

A PCP of Quasilinear Size for Arithmetic Circuit-SAT

Lecturer: Justin Thaler

1 The Goal of This Lecture

We have seen previously (Lecture 15, Section 1) that known MIPs can fairly directly yield a PCP of polynomial size for simulating a (non-deterministic) Random Access Machine (RAM), in which the verifier runs in linear time. But the polynomial proof length and prover runtime may be quite large. This section describes how to use techniques tailored specifically to the PCP model to reduce the PCP length to $T \cdot \text{polylog}(T)$ for simulating a RAM that runs in time T, while maintaining a verifier runtime of $n \cdot \text{polylog}(T)$.

The PCP described here originates in work of Ben-Sasson and Sudan [BS08]. Their work gave a PCP of size $\tilde{O}(T)$ in which the verifier runs in time $\text{poly}(n)$ and makes only makes a polylogarithmic number of queries to the proof oracle. Subsequent work by Ben-Sasson et al. [BGH$^+$05] reduced the verifier's time to $n \cdot \text{polylog}(T)$. Finally, Ben-Sasson et al. [BSCGT13] showed how to actually generate the PCP in $T \cdot \text{polylog}(T)$ time using FFT techniques, and the factors hidden by the $\tilde{O}$ notation. This PCP system is fairly involved, so we elide some details in this survey.

1.1 Step 1: Reduce to checking that a polynomial vanishes on designated a subspace.

In Ben-Sasson and Sudan's PCP, the prover (or more precisely, the proof string π) claims to be holding a low-degree extension Z of a correct transcript W for the computation being simulated, just like in the MIP of Lecture 14. And just as in the MIP, the first step of Ben-Sasson and Sudan's PCP is to construct a polynomial $g_{x,y,Z}$ such that Z extends a correct transcript for $\{\mathcal{C}, x, y\} \iff g_{x,y,Z}(\mathbf{a}) = 0$ for all a in a certain set H.

The process of constructing the polynomial $g_{x,y,Z}$ used in the PCP bears much resemblance to the "RAMs $\implies$ Circuit-SAT instance $\{\mathcal{C}, x, y\} \implies$ multivariate polynomial $g_{x,y,Z}$" sequence of transformations used in the MIP of Lecture 14. For example, De Bruijn graphs again play a central role in the construction used in the PCP, for the same reasons as in the MIP: they can be used for sorting, while maintaining an "algebraically regular" structure. This structure allows for keeping the degree of $g_{x,y,Z}$ low while still keeping $g_{x,y,Z}$ "efficiently evaluate-able" in the sense that the verifier can compute $g_{x,y,Z}(r)$ at any point r, if given Z's values at a handful of points derived from r.

The details, however, are different and somewhat more involved than the construction in the MIP. We elide several of these details here, and focus on highlighting the primary differences between the constructions in the PCP and the MIP.

Most importantly, in the PCP, $g_{x,y,Z}$ ***is a univariate polynomial.*** The PCP views a correct transcript as a univariate function $W : [S] \to \mathbb{F}$ rather than as a v-variate function mapping $\{0,1\}^v$ to $\mathbb{F}$ for $v = \log S$ as in the MIP. Hence, any extension Z of W is a univariate polynomial, and $g_{x,y,Z}$ is defined to be a univariate polynomial too. (The reason for using univariate polynomials is that it allows the PCP to utilize low-degree testing techniques in Steps 2 and 3 below that are tailored to univariate rather than multivariate polynomials. It is not currently known how to obtain PCPs of quasilinear length based on multivariate techniques.). Note that even the lowest-degree extension Z of W may have degree $|S| - 1$, which is much larger than the degrees of the multivariate polynomials that we've used in previous lectures, and $g_{x,y,Z}$ will inherit this degree.

Communication	Queries	$\mathcal{V}$ time	$\mathcal{P}$ time
polylog(S) bits	polylog(S)	$O(n \cdot \text{polylog}(S))$	$O(S \cdot \text{polylog}(S))$

Table 1: Costs of PCP from Section 1 when run on a non-deterministic circuit $\mathcal{C}$ of size S. The PCP is due to Ben-Sasson and Sudan [BS08], as refined by Ben-Sasson et al. [BGH+05] and [BSCGT13]. The stated bound on $\mathcal{P}$'s time assumes $\mathcal{P}$ knows a witness w for $\mathcal{C}$.

The univariate nature of $g_{x,y,Z}$ forces several additional differences in its construction, compared to the multivariate polynomial used in the MIP. In particular, in the univariate setting, $g_{x,y,Z}$ is specifically defined over a field GF[2^ℓ] of characteristic 2. The structure of fields of characteristic 2 are exploited multiple times in the construction of $g_{x,y,Z}$ and in the PCP as a whole. For example:

- The construction of $g_{x,y,Z}$ exploits the fact that there is a way to assign labels from $\mathbb{F} = \text{GF}[2^\ell]$ to nodes in a De Bruijn[1] graph such that, for each node v, the labels of the neighbors of v are *affine* (i.e., degree 1) functions of the label of v. (Just as in Lecture 11, the reason this holds boils down to the fact that the neighbors of a node with label v are simple bit-shifts of v. When v is an element of GF[2^ℓ], a bit-shift of v is an affine function of v.).

  This is crucial for ensuring that the degree of $g_{x,y,Z}$ is not much larger than the degree of Z itself. In particular, the $g_{x,y,Z} \colon \mathbb{F} \to \mathbb{F}$ used in the PCP has the form

  $$g_{x,y,Z}(z) = A(z, Z(N_1(z)), \ldots, Z(N_k(z))), \tag{1}$$

  where $(N_1(z), \ldots, N_k(z))$ denotes the neighbors of node z in the De Bruijn graph, and A is a certain "constraint polynomial" of polylogarithmic degree. Since $N_1, \ldots, N_k$ are affine over GF[2^ℓ], $\deg(g_{x,y,Z})$ is at most a polylogarithmic factor larger than the degree of Z itself.

- The set H on which $g_{x,y,Z}$ should vanish if Z extends a correct transcript is chosen to ensure that the polynomial $h_H(z) = \prod_{\alpha \in H}(z - \alpha)$ is *sparse* (having $O(\text{polylog}(S))$ non-zero coefficients). The sparsity of h_H ensures that it can be evaluated an any point in polylogarithmic time, even though H is a very large set (of size $\Omega(S)$). This will be crucial to allowing the verifier to run in polylogarithmic time in Step 2 of the PCP, discussed below. It turns out that if $\mathbb{F}$ has characteristic $O(1)$ and H is a linear subspace of $\mathbb{F}$, then $h_H(z)$ has sparsity $O(\log S)$ as desired.

The final difference worth highlighting is that the field GF[2^ℓ] over which $g_{x,y,Z}$ is defined must be small in the PCP. In particular, $\mathbb{F}$ must be of size $O(S \cdot \text{polylog}(S))$, since the proof length will be at least as large as $|\mathbb{F}|$. This is in contrast to the MIP setting, where the field size was not critical (so long as it was sufficiently large to ensure negligible soundness error).

1.1.1 Step 2: Reducing to Checking that a Related Polynomial is Low-Degree

We have already seen the reduction described in this section, in Lemma 1.1 of Lecture 18. We recall the reduction for the reader's convenience.

Lemma 1.1 ([BS08]). *A degree d univariate polynomial $g_{x,y,Z}(z)$ vanishes on H if and only if the polynomial $h_H(z) := \prod_{\alpha \in H}(z - \alpha)$ divides $g_{x,y,Z}(z)$, i.e., iff there exists a polynomial h^* with $\deg(h^*) \leq d - |H|$ such that $g_{x,y,Z}(z) = h_H(z) \cdot h^*(z)$.*

[1] Recall that we saw De Brujin graphs in the Lecture 11 notes, where they used to "resort" the trace of a random access machine's execution from time order into memory order.

Proof. See Lemma 1.1 of Lecture 18. □

So to convince $\mathcal{V}$ that $g_{x,y,Z}$ vanishes on H, the proof merely needs to convince $\mathcal{V}$ that $g_{x,y,Z}(z) = h_H(z) \cdot h^*(z)$ for some polynomial h^* of degree $d - |H|$. To be convinced of this, $\mathcal{V}$ can pick a random point $r \in \mathbb{F}$ and check that

$$g_{x,y,Z}(r) = h_H(r) \cdot h^*(r). \tag{2}$$

Indeed, if $g_{x,y,Z} \neq h_H \cdot h^*$, then this equality will fail with probability $\frac{999}{1000}$ as long as $|\mathbb{F}|$ is at least 1000 times larger than the degrees of $g_{x,y,Z}$ and $h_H \cdot h^*$.

A PCP convincing $\mathcal{V}$ that Equation (2) holds consists of four parts. The first part contains the evaluations of $Z(z)$ for all $z \in \mathbb{F}$. The second part contains a proof π_Z that Z has degree at most $|H| - 1$, and hence that $g_{x,y,Z}$ has degree at most $d = |H| \cdot \text{polylog}(S)$. The third part contains the evaluation of $h^*(z)$ for all $z \in \mathbb{F}$. The fourth part purportedly contains a proof π_{h^*} that $h^*(z)$ has degree at most $d - |H|$, and hence that $h_H \cdot h^*$ has degree at most d.

Let us assume that the verifier can efficiently check π_Z and π_{h^*} to confirm that Z and $h^*(z)$ have the claimed degrees (this will be the purpose of Step 3 below). $\mathcal{V}$ can evaluate $g_{x,y,Z}(r)$ in quasilinear time after making a constant number of queries to the first part of the proof specifying Z. $\mathcal{V}$ can compute $h^*(r)$ with a single query to the third part of the proof. Finally, $\mathcal{V}$ can evaluate $h_H(r)$ without help in polylogarithmic time as described in Step 1. The verifier can then check that $g_{x,y,W}(r) = h^*(r) \cdot h_H(r)$.

In actuality, Step 3 will not be able to guarantee that π_Z and π_{h^*} are *exactly* equal to low-degree polynomials, but will be able to guarantee that, if the verifier's checks all pass, then they are each close to some low-degree polynomial Y and h' respectively. One can then argue that $g_{x,y,Y}$ vanishes on H, analogously to the proof of Theorem 2.2 in the context of the MIP from Lecture 14.

1.1.2 Step 3: A PCP for Checking that a Univariate Polynomial Has Low-Degree

Overview. The meat of the PCP construction is in this third step. The construction is recursive. The basic idea is to reduce the problem of checking that a *univariate* polynomial G_1 has degree at most d to the problem of checking that a related *bivariate* polynomial Q over $\mathbb{F}$ has degree at most $\sqrt{d}$ in each variable. It is known (cf. Lemma 1.3 below) how the latter problem can in turn be reduced back to a univariate problem, that is, to checking that a related univariate polynomial G_2 over $\mathbb{F}$ as degree at most $\sqrt{d}$. Recursing $\ell = O(\log\log n)$ times results in checking that a polynomial G_ℓ has *constant* degree, which can be done with constantly many queries to the proof. We fill in some of the details of this outline below.

The precise soundness and completeness guarantees of this step are as follows. If G_1 indeed has degree at most d, then there is a proof π that is always accepted. Meanwhile, the soundness guarantee is that there is some universal constant k satisfying the following property: if a proof π is accepted with probability $1 - \varepsilon$, then there is a polynomial G of degree at most d such that G_1 agrees with G at a $1 - \varepsilon \cdot \log^k(S)$ fraction of points in $\mathbb{F}$ (we say that G and G_1 are δ-*far*, for $\delta = \varepsilon \cdot \log^k(S)$.)

The claimed polylogarithmic query complexity of the PCP as a whole comes by repeating the base protocol, say, $m = \log^{2k}(S)$ times and rejecting if any run of the protocol ever rejects. If a proof π is accepted by the m-fold repetition with probability $1 - \varepsilon$, then it is accepted by the base protocol with probability at least $1 - \varepsilon/\log^k m$, implying that G is ε-far from a degree d polynomial G_1.

Reducing Bivariate Low-Degree Testing on Product Sets to Univariate Testing. The bivariate low-degree testing technique described here is due to Spielman and Polishchuk [PS94]. Assume that Q is a

bivariate polynomial defined on a product set $A \times B \subseteq \mathbb{F} \times \mathbb{F}$, claimed to have degree d in each variable. (In all recursive calls of the protocol, A and B will in fact both be subspaces of $\mathbb{F}$). The goal is to reduce this claim to checking that a related univariate polynomial G_2 over $\mathbb{F}$ has degree at most d.

Definition 1.2. For a set $U \subseteq \mathbb{F} \times \mathbb{F}$, partial bivariate function $Q\colon U \to \mathbb{F}$, and nonnegative integers d_1, d_2, define $\delta^{d_1,d_2}(Q)$ to be the fractional distance of Q from a polynomial of degree d_1 in its first variable and d_2 in its second variable. Formally,

$$\delta^{d_1,d_2}(Q) := \min_{f\colon U \to \mathbb{F}, \deg_x(f) \le d_1, \deg_y(f) \le d_2} \delta(Q,f).$$

Let $\delta^{d_1,*}(Q)$ and $\delta^{*,d_2}(Q)$ denote the fractional distances when the degree in one of the variables is unrestricted.

Lemma 1.3. *(Bivariate test on a product set [PS94]). There exists a universal constant $c_0 \ge 1$ such that the following holds. For every $A, B \subseteq \mathbb{F}$ and integers $d_1 \le |A|/4$, $d_2 \le |B|/8$ and function $Q\colon A \times B \to \mathbb{F}$, it is the case that $\delta^{d_1,d_2}(Q) \le c_0 \cdot \left(\delta^{d_1,*}(Q) + \delta^{*,d_2}(Q)\right)$.*

The proof of Lemma 1.3 is not long, but we omit it from the survey for brevity.

Lemma 1.3 implies that, to test if a bivariate polynomial Q defined on a product set has degree at most d in each variable, it is sufficient to pick a variable $i \in \{1,2\}$, then pick a random value $r \in \mathbb{F}$ and test whether the univariate polynomial $Q(r,\cdot)$ or $Q(\cdot,r)$ obtained by restricting the ith coordinate of Q to r has degree at most d.

To be precise, if the above test passes with probability $1-\varepsilon$, then $\left(\delta^{d,*}(Q) + \delta^{*,d}(f)\right)/2 = \varepsilon$, and Lemma 1.3 implies that $\delta^{d,d}(Q) \le 2 \cdot c_0 \cdot \varepsilon$. Note that $Q(r,\cdot)$ and $Q(\cdot,r)$ are typically called a "random row" or "random column" of Q, respectively, and the above procedure is referred to as a "random row or column test".

Note that $\delta^{d,d}(f)$ may be larger than the acceptance probability ε by only a constant factor $c_1 = 2c_0$. Ultimately, the PCP will will recursively apply the "Reducing Bivariate Low-Degree Testing to Univariate Testing" technique $O(\log\log n)$ times, and each step may cause $\delta^{d_1,d_2}(Q)$ to blow up, relative to the rejection probability ε, by a factor of c_1. This is why the final soundness guarantee states that, if the recursive test as a whole accepts a proof with probability $1-\varepsilon$, then the input polynomial G_1 is δ-close to a degree d polynomial, where $\delta = \varepsilon \cdot c_1^{O(\log\log S)} = \varepsilon \cdot \mathrm{polylog}(S)$.

Reducing Univariate Low-Degree Testing to Bivariate Testing on a Lower Degree Polynomial. Let G_1 be a univariate polynomial defined on a linear subspace L of $\mathbb{F}$ (In all recursive calls of the protocol, the domain of G_1 will indeed be a subspace L of $\mathbb{F}$). Our goal in this step is to reduce testing that G_1 has degree at most d to testing that a related bivariate polynomial Q has degree at most $\sqrt{d}$ in each variable. It is okay to assume that the number of vectors in L is at most a constant factor larger than d, as this will be the case every time this step is applied.

Lemma 1.4. *[BS08] Given any pair of polynomials $G_1(z)$, $q(z)$, there exists a unique bivariate polynomial $Q(x,y)$ with $\deg_x(Q) < \deg(G_1)$ and $\deg_y(Q) = \lfloor \deg(G_1)/\deg(q) \rfloor$ such that $G_1(z) = Q(z,q(z))$.*

Sketch. Divide $G_1(z)$ by $(y - q(z))$ to represent $G_1(z)$ as:

$$G_1(z) = Q_0(z,y) \cdot (y - q(z)) + Q(z,y). \tag{3}$$

By the basic properties of division in this ring, $\deg_y(Q) = \lfloor \deg(G_1)/\deg(q) \rfloor$, and $\deg_z(Q) < \deg(q)$. To complete the proof, set $y = q(z)$ and notice that the first summand on the right-hand side of Equation (3) vanishes. □

By Lemma 1.4, to establish that G_1 has degree at most d, it suffices for a proof to establish that $P = Q(z, q(z))$, where the degree of Q in each variable is at most $\sqrt{d}$. Thus, as a first (dumb) attempt, the proof could specify Q's value on all points in $L \times \mathbb{F}$. Then $\mathcal{V}$ can check that $G_1(z) = Q(z, q(z))$, by picking a random $r \in L$ and checking that $G_1(r) = Q(r, q(r))$. If this check passes, it is safe for $\mathcal{V}$ to believe that $G_1(z) = Q(z, q(z))$, as long as Q is indeed low-degree in each variable, and we have indeed reduced testing that G_1 has degree $\approx d$ to testing that Q has degree at most $\sqrt{d}$ in each variable.

The problem with the dumb attempt is that the proof has length $|L| \cdot |\mathbb{F}|$, which is far too large; we need a proof of size $\tilde{O}(|L|)$. A second attempt might be to have the proof specify Q's value on all points in the set $\mathcal{T} := \{(z, q(z)) : \ : z \in L\}$. This would allow $\mathcal{V}$ to check that $G_1(z) = Q(z, q(z))$ by picking a random $r \in L$ and checking that $G_1(r) = Q(r, q(r))$. While this shortens the proof to an appropriate size, the problem is that $\mathcal{T}$ is not a product set, so Lemma 1.3 cannot be applied to check that Q has low-degree in each variable.

To get around this issue, Ben-Sasson and Sudan ingeniously choose the polynomial $q(z)$ in such a way that there is a set $\mathcal{B}$ of points, of size $O(|L|)$, at which it suffices to specify Q's values. Specifically, they choose $q(z) = \prod_{\alpha \in L_0}(z - \alpha)$, where L_0 is a linear subspace of L containing $\sqrt{d}$ vectors. Then $q(z)$ is not just a polynomial of degree $\sqrt{d}$, it is also a *linear map* on L, with kernel equal to L_0. This has the effect of ensuring that $q(z)$ takes on just $|L|/|L_0|$ distinct values, as z ranges over L.

Ben-Sasson and Sudan use this property to show that, although $\mathcal{T}$ is not a product set, it is possible to add $O(L)$ additional points $\mathcal{S}$ to $\mathcal{T}$ to ensure that $\mathcal{B} := \mathcal{S} \cup \mathcal{T}$ contains within it a large subset that is product. So $\mathcal{P}$ need only provide Q's evaluation on the points in $\mathcal{B}$: since $\mathcal{T} \subseteq \mathcal{B}$, the verifier can check that $G_1(z) = Q(z, q(z))$ by picking a random $r \in L$ and checking that $G_1(r) = Q(r, q(r))$, and since there is a large product set within $\mathcal{S} \cup \mathcal{T}$, Lemma 1.3 can be applied.

References

[BGH+05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Short pcps verifiable in polylogarithmic time. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 120–134. IEEE Computer Society, 2005.

[BS08] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput.*, 38(2):551–607, 2008.

[BSCGT13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *STOC*, pages 585–594, 2013.

[PS94] Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 194–203. ACM, 1994.