Algebraic Linear IOPs

An interactive oracle proof (IOP) TCC:BenChiSpo16,STOC:ReiRotRot16 is a multi-round interactive PCP: in each round of an IOP the verifier sends a message to the prover and the prover responds with a polynomial length proof, which the verifier can query via random access. A t-round ℓ-query IOP has t rounds of interaction in which the verifier makes exactly ℓ queries in each round. Linear IOPs C:BBCGI19 are defined analogously except that in each round the prover sends a linear PCP CC:IKO07, in which the prover sends a single proof vector $\in^m$ and the verifier makes $linear queries$ to π. Specifically, the PCP gives the verifier access to an oracle that receives queries of the form $q \in^m$ and returns the inner product $\langle , q\rangle$.

Bitansky et al. TCC:BCIOP13 defined a linear PCP to be of degree (d_Q, d_V) if there is an explicit circuit of degree d_Q that derives the query vector from the verifier's random coins, and an explicit circuit of degree d_V that computes the verifier's decision from the query responses. In a multi-query PCP, d_Q refers to the maximum degree over all the independent circuits computing each query. Bitansky et al. called the linear PCP algebraic for a security parameter λ if it has degree $(,)$. The popular linear PCP based on Quadratic Arithmetic Programs (QAPs) implicit in the GGPR protocol EC:GGPR13 and follow-up works is an algebraic linear PCP with $d_Q \in O(m)$ and $d_V = 2$, where m is the size of the witness.

For the purposes of the present work, we are only interested in the algebraic nature of the query circuit and not the verifier's decision circuit. Of particular interest are linear PCPs where each query-and-response interaction corresponds to the evaluation of a fixed μ-variate degree d polynomial at a query point in $^\mu$. This description is equivalent to saying that the PCP is a vector of length $m = \binom{d+\mu}{\mu}$ and the query circuit is the vector of all μ-variate monomials of degree at most d (in some canonical order) evaluated at a point in $^\mu$. We call this a (μ, d) Polynomial PCP and define Polynomial IOPs analogously. As we will explain, we are interested in Polynomial PCPs where $\mu \ll m$ because we can cryptographically compile them into succinct arguments using polynomial commitments, in the same way that Merkle trees are used to compile classical (point) IOPs.

In general, evaluating the query circuit for a linear PCP requires $\Omega(m)$ work. However, a general "bootstrapping" technique can reduce the work for the verifier: the prover expands the verifier's random coins into a full query vector, and then provides the verifier with a second PCP demonstrating that this expansion was computed correctly. It may also help to allow the verifier to perform $O(m)$ work in a one-time preprocessing stage (for instance, to check the correctness of a PCP oracle), enabling it to perform sublinear "online" work when verifying arbitrary PCPs later. We call this a preprocessing IOP. In fact, we will see that any t-round (μ, d) algebraic linear IOP can be transformed into a $(t+1)$-round Polynomial IOP in which the verifier preprocesses (μ, d) Polynomial PCPs, at most one for each distinct query.

We recall the formal definition of public-coin linear IOPs as well an algebraic linear IOPs. Since we are not interested in the algebraic nature of the decision algorithm, we omit specifying the decision polynomial. From here onwards we use algebraic linear IOP as shorthand for algebraic query linear IOP.

definition [Public-coin linear IOP] def:linearIOP Let R be a binary relation and a finite field. A t-round ℓ-query public-coin linear IOP for R over with soundness error ϵ and knowledge error δ and query length $m = (m_1, ..., m_t)$ consists of two stateful PPT algorithms, the prover , and the verifier $= (,)$, where the verifier consists in turn of a public deterministic query generator and a decision algorithm , that satisfy the following requirements:

$\underline{Protocol syntax}.For each ith round there is a prover state$st^i and a verifier state st^i. For any common input x and R witness w, at round 0 the states are $st^{0=(x,w)}$ and $st^{0=x}$. In the ith round (starting at $i = 1$) the prover outputs a singleThe prover may also output more than one proof oracle per round, however this doesn't add any power since two proof oracles of the same size may be viewed as a single (concatenated) oracle of twice the length.roof oracle $(st^{i-1)\to_i\in^{m_i}}$. The verifier samples public random coins $coins_i\{0,1\}^*$ and the query generator computes a query matrix from the verifier state and these coins: $(st^{i-1,coins_i)\to Q_i\in F^{m_i\times\ell}}$. The verifier obtains the linear oracle response vector $_i^\top Q_i = a_i \in F^{1\times\ell}$. The updated prover state is $st^{i\leftarrow(st^{i-1},Q_i)}$ and verifier state is $st^{i\leftarrow(st^{i-1},coins_i,a_i)}$ Finally, $(st^{t)}$ returns 1 or 0.

(Querying prior round oracles: The syntax can be naturally extended so that in the ith round the verifier may query any oracle, whether sent in the ith round or earlier).

of twice the length. p

$\underline{Argumentof Knowledge.}Asaproofsystem,(,)satisfiesperfectcompleteness,soundnesswithrespecttotherelation\mathrm{R}andw$
and witness-extended emulation with respect R with knowledge error δ.

Furthermore, a linear IOP is stateless if for each $i \in [t]$, $(st^{i-1,coins_i)=(i,coins_i)}$. It has algebraic queries if, additionally, for each $i \in [t]$, the map $coins_i Q(i,\cdot)Q_i \in F^{m_i \times \ell}$ decomposes into two maps, $coins_{i0}(i,\cdot)\Sigma_{i1}(i,\cdot)Q_i$, where $\Sigma_i \in F^{\mu_i \times \ell}$ is a matrix of $\mu_i < m_i$ rows and ℓ and ${}_1(i,\cdot)$ is described by ℓ μ_i-variate polynomial functions of degree at most $d =: \vec{p}_1, \dots, \vec{p}_\ell : F^{\mu_i} \to F^{m_i}$ such that for all $k \in [\ell]$, $\vec{p}_k(\sigma_{i,k}) = q_{i,k}$, where $\sigma_{i,k}$ and $q_{i,k}$ denote the kth column of Σ_i and Q_i, respectively.