Interactive Arguments with Preprocessing

Bar-Ilan Winter School on Verifiable Computation
Class 5
January 4, 2016

Michael Walfish
Dept. of Computer Science, Courant Institute, NYU
Next two sessions: foundations for (implemented) arguments

- Key ideas: linear PCPs, QAPs

This session: focus on interactive arguments

- Conceptually illuminating
- Based on standard assumptions
- Plan: linear PCPs, interactivity, the role of QAPs
Recall: by construction, $C$ is satisfiable iff $f$ executes correctly.

More formally: $C$ is degree-2 constraints over $\mathbb{F}$ and variables $(X, Y, Z)$ s.t.

$$\forall x, y: \exists w \text{ s.t. } y = f(x, w) \iff C(X=x, Y=y) \text{ is satisfiable}$$

Thus, as usual, $P$ must convince $V$ that constraints $C'$ are satisfiable. ($C'$ is notation for $C(X=x, Y=y)$.)
Goal: P convinces V that a set of constraints is satisfiable while:

- V’s running time (possibly amortized) is less than executing f
- P’s running time is quasilinear in the time to execute f
- mechanics are simple, concrete costs aren’t crazy

(Today, assume that P has an assignment, z. Tomorrow, we will describe how P derives z.)

Attempt 0: P sends z to V; V checks that z satisfies all constraints
- Doesn’t meet the goal; z is same “size” as running time of f
Attempt 1: Use PCPs that are asymptotically short
[ALMSS92, AS92]  [BGHSV05, BGHSV06, Dinur07, BS08, Meir12, BCGT13]

This also doesn’t meet the goal (because $|\text{PCP}| >$ running time of $f$).

For more detail on this picture, see the notes after the end of the slides.
Attempt 2: Use arguments or CS proofs
[Kilian92, Micali94, BG02]

But the constants seem too high …

For more detail on this picture, see the notes after the end of the slides.
Attempt 3: Use long PCPs interactively

Claim (informal): V can check the satisfiability of constraints C’ by inspecting a long proof in a constant number of entries.

A more formal version of this claim is a “lite” PCP theorem: QuadConstraint_{\mathbb{F}} \subseteq \text{PCP}(\text{poly}(n), \log |\mathbb{F}|), with soundness error 7/9.

Claim (informal): P can play the role of the long proof without materializing the proof explicitly.

More formally: Using additively homomorphic encryption, the PCP above implies an argument (with similar soundess) in which the V→P communication complexity is the length of the PCP queries, and the P→V communication is a constant number of field elements.

This requires preprocessing, or setup work, for V.

Claim (soft): The approach is simple, with good constants.
Claim: \( \forall \) can check the satisfiability of degree-2 constraints \( C' \) (over a field \( \mathbb{F} \)) by inspecting a long proof in a constant number of entries.

Consider a polynomial \( Q(Z) \triangleq \sum_{j=1}^{m} \gamma_j \cdot Q_j(Z) \):

\( Q(Z) \) is a “bellwether”:

- If \( Z = z \) satisfies \( C' \), then . . .
- If \( Z = z \) does not satisfy \( C' \), then . . .
$Q(z)$ can be evaluated with two queries to a correct proof oracle, $\pi$

\[ Q(z) = \]

\[ \pi_1 = \]

\[ \pi_2 = \]
But what if $\pi$ is adversarially constructed?

- $\pi_1, \pi_2$ might not be Hadamard codewords
- $\pi_1, \pi_2$ might be close to Hadamard codewords but inconsistent
- $\pi_1, \pi_2$ might be close to Hadamard codewords and consistent but encode a non-satisfying assignment.

Hadamard codeword $\longleftrightarrow$ linear function. So test for linearity [BLR90].

Defn. of $\delta$-close, $\delta$-far: functions $g, h$ with the same domain are $\delta$-close ($\delta$-far) if they disagree on less (more) then a fraction $\delta$ of their domain.
But what if \( \pi \) is adversarially constructed?

- \( \pi_1, \pi_2 \) might not be (close to) Hadamard code words
- \( \pi_1, \pi_2 \) might be close to Hadamard codewords but inconsistent

Quad correction test:
But what if $\pi$ is adversarially constructed?

- $\pi_1, \pi_2$ might not be (close to) Hadamard code words
- $\pi_1, \pi_2$ might be close to Hadamard codewords but inconsistent
- $\pi_1, \pi_2$ might be close to Hadamard codewords and consistent but encode a non-satisfying assignment.

Circuit test:
Putting together the guarantees of the tests:

- If $C'$ is satisfiable, there are $\pi_1, \pi_2$ s.t. $\Pr\{\text{all tests pass}\} = 1$.
- If $C'$ is not satisfiable, then for all $\pi_1, \pi_2$, $\Pr\{\text{all tests pass}\} \leq \frac{7}{9}$.

There were 14 queries; each of length $n$ or $n^2$ field elements. The mechanics were simple.

So our desired claim is established:

\[ \forall \text{ can check the satisfiability of degree-2 constraints } C' \text{ (over } \mathbb{F}) \text{ by inspecting a long proof in a constant number of entries.} \]

More formally, \( \text{QuadConstraint}_\mathbb{F} \subset \text{PCP}(\text{poly}(n), \log |\mathbb{F}|) \).

This kind of PCP is known as a Hadamard PCP, or linear PCP [ALMSS92, IKO07].
Attempt 3: Use long PCPs interactively

Claim (informal): $V$ can check the satisfiability of constraints $C'$ by inspecting a long proof in a constant number of entries.

A more formal version of this claim is a “lite” PCP theorem:
QuadConstraint$_\mathbb{F} \subset \text{PCP}(\text{poly}(n), \log |\mathbb{F}|)$, with soundness error $7/9$.

Claim (informal): $P$ can play the role of the long proof without materializing the proof explicitly.

More formally: Using additively homomorphic encryption, the PCP above implies an argument (with similar soundess) in which the $V \rightarrow P$ communication complexity is the length of the PCP queries, and the $P \rightarrow V$ communication is a constant number of field elements.

This requires preprocessing, or setup work, for $V$.

Claim (soft): The approach is simple, with good constants.
How can $\mathcal{P}$ play the role of an exponentially-sized linear PCP?

- Idea: represent linear PCP implicitly, as a **linear function**
- Idea: $\mathcal{V}$ uses additively homomorphic encryption to make $\mathcal{P}$ commit to a function of this form.
Informal guarantee of linear commitment primitive: $\mathcal{P}$ is bound to a fixed function.

This leads to our desired claim:

$\mathcal{P}$ can play the role of the linear PCP without materializing that PCP explicitly. More formally:

Using additively homomorphic encryption, a linear PCP can be transformed into an argument (with negligibly more soundness error) in which the $\mathcal{V}$-to-$\mathcal{P}$ communication complexity equals the PCP query length and the $\mathcal{P}$-to-$\mathcal{V}$ communication is a constant number of field elements.
\[ t = r + \alpha_1 q_1 + \ldots + \alpha_u q_u, \quad \alpha_i \overset{R}{\leftarrow} \mathbb{F} \]

consistency test: \[ \pi(t) = \pi(r) + \alpha_1 \pi(q_1) + \ldots + \alpha_u \pi(q_u) \]
Attempt 3: Use long PCPs interactively (summary)  
[IKO07, SMBW12]

Achieves simplicity, with good constants …

… but pre-processing is required (because $|q_i| = |v|$)

… and prover’s work is quadratic; address that shortly
Attempt 4: Use long PCPs non-interactively \[\text{[BCIOP13]}\]

Query process now happens “in the exponent”

... **pre-processing** still required (again because \( |q_i| = |v| \))

... prover’s work still quadratic; addressing that soon
## Recap

(Thanks to Rafael Pass.)

<table>
<thead>
<tr>
<th>who</th>
<th>efficient (short) PCPs</th>
<th>arguments, CS proofs</th>
<th>arguments w/ preprocessing</th>
<th>SNARGs w/ preprocessing</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALMSS92, AS92, BGSHV, Dinur, ...</td>
<td>Kilian92, Micali94</td>
<td></td>
<td>IKO07, SMBW12, SVPBBW12</td>
<td>Groth10, GGPR12, BCIOP13, ...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>what</th>
<th>classical PCP</th>
<th>commit to PCP by hashing</th>
<th>commit to long PCP using linearity</th>
<th>encrypt queries to a long PCP</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>security</th>
<th>unconditional</th>
<th>CRHFs</th>
<th>linearly HE</th>
<th>knowledge-of-exponent</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>why/why not</th>
<th>not efficient for V</th>
<th>constants are unfavorable</th>
<th>simple</th>
<th>simple, non-interactive</th>
</tr>
</thead>
</table>
“As noted above, state-of-the-art PCP constructions leave little to be desired in terms of asymptotic efficiency. However, the practical motivation of verifying computations may call for a more refined efficiency analysis. From this point of view, we believe that our approach has potential to yield better efficiency, at least in some circumstances. … our approach does not inherently require the prover to compute a redundant encoding of its input. This suggests the possibility of designing PCPs that are optimized to make better use of the ‘implicit encoding’ feature of our approach.”

—Ishai, Kushilevitz, and Ostrovsky,
Efficient Arguments without Short PCPs, 2007
Final attempt: apply linear query structure to GGPR’s QAPs
[Groth10, Lipmaa12, GGPR12]

Addresses the issue of quadratic costs.

PCP structure implicit in GGPR. Made explicit in [BCIOP13, SBVBBW13].
Summary of published argument implementations

“Zaatar” [SBVBBW13]

interactive argument [IKO07]

“Pinocchio,” “libsnark” [PGHR13, BCTV14b]

SNARG, zk-SNARK with preprocessing [Groth10, BCCT12, GGPR12]

preprocessing lowered to (high) constant [BCCT13, BCTV14a, CTV15]

---

linear PCP via QAPs [GGPR13]

plaintext queries

queries in exponent

---

- standard assumptions
- amortize over batch
- interactive

- non-falsifiable assumptions
- amortize indefinitely
- non-interactive, ZK, …

All recent implementations are based on GGPR
SBVBPW13, PGHR13, BFRSBW13, BCGTV13, BCGGMTV14, BCTV14a, BCTV14b, FL14, KPPSST14, WSRBW15, CFHKKNPZ15, BBFR15, CTV15
Summary of key concepts

1. **Linear (Hadamard) PCPs**, to prove satisfiability
   - Exponentially long but mechanically simple, with good constants

2. Linear PCPs can be transformed into argument protocols with preprocessing
   - Interactive version: only standard assumptions
   - Non-interactive version: better amortization and properties

3. **QAPs** lower quadratic costs to quasilinear, and fit into the linear PCP framework.
NOTES
Notes for Attempt 1 (Use asymptotically short PCPs)

(This note relies heavily on [AB09, Ch. 11].)

Let us define the complexity class $\text{PCP}$. A language $L \in \text{PCP}(r(n), q(n))$ if:

- **Efficiency.** There is a PPT (probabilistic polynomial-time algorithm) $V_L$ that, on input $a \in \{0, 1\}^n$, uses $O(r(n))$ random coins and inspects $O(q(n))$ locations in a string $\pi$, after which $V_L$ outputs “accept” or “reject”.

- **Completeness.** If $a \in L$, there exists a $\pi$ s.t. $\Pr\{V_L^\pi(a) \text{ accepts}\} = 1$.

- **Soundness.** If $a \notin L$, then for all $\tilde{\pi}$, $\Pr\{V_L^{\tilde{\pi}}(a) \text{ accepts}\} < 1/2$.

The notation $V_L^{\pi}(\cdot)$ denotes $V_L$ with random access to $\pi$. The probabilities are taken over the random coins of $V_L$.

The PCP theorem [ALMSS92, AS92] says: $\text{NP} = \text{PCP}(\log n, 1)$.

What happens if we apply this theory to QuadConstraint$_\mathbb{F}$?

Correct proofs are “short”: asymptotically, they are bounded by $2^{O(\log n)} \cdot O(1)$ (because this is the space “swept out” by the verifier’s coin flips).

But this length is still longer than $n$ (which in our context, is the length of the execution trace of the computation), so transmitting such a proof to $V$ would conflict with our efficiency goals.
Informal and rough definition of an argument: an interactive proof (that is, a probabilistic verifier \( \mathcal{V} \) and a prover \( \mathcal{P} \)) plus an assumption that any would-be prover is computationally limited [BCC86, GMR85, Kilian92, Micali94, BG02]. In particular, a dishonest \( \mathcal{P} \) with unbounded computational ability can “win” (spuriously convince \( \mathcal{V} \) that instances are in a language). In addition, in the current context, we want the protocol to require not much more of an honest \( \mathcal{P} \) than \( T \): the time required to decide the given instance’s membership (in our context, this corresponds to running the computation \( f \)). We also want \( \mathcal{V} \)’s work to be substantially less than \( T \). These properties were beautifully articulated by [Micali94] in the context of CS proofs.

Theorem (informal): Assuming CRHFs (collision-resistant hash functions), arguments (and CS proofs) exist.

The construction is “PCP + tree hash” [Kilian92, Micali94, BG02]. Specifically, \( \mathcal{P} \) materializes a PCP, and commits to it using a hash tree [Merkle87]. Then, \( \mathcal{V} \) asks \( \mathcal{P} \) what values the PCP contains at particular locations; \( \mathcal{P} \) is forced (by the CRHF assumption) to respond in a way that is consistent with the original commitment [Merkle87, BEGKN91]. Thus (except with negligible probability), \( \mathcal{P} \) “acts like” the fixed proof string of the PCP model.

This results in a 4-message (2-round) scheme. Micali makes this approach non-interactive, in the Random Oracle model. Barak and Goldreich strengthen the analysis so that the construction can work with a CRHF that resists a stronger prover.
What happens if we apply this theory to QuadConstraint\(_F\)?

This is a great idea in principle. In fact, it remains asymptotically the best approach, if we are limited to standard assumptions (in which case we use the interactive version) or we are comfortable with the Fiat-Shamir Heuristic and the Random Oracle Model (in which case we get a non-interactive version).

But in practice, it’s rather costly because of the high constants and intricate constructions in asymptotically short PCP constructions. Indeed, despite intense interest, no experimental results from this approach have been reported.
Notes for Attempt 3 (Use long PCPs interactively)

This approach turns to a long (or linear) PCP (a concept that we are fleshing out in this class). This approach is worse in theory than the prior approach; yet, it’s a simpler approach, and yields good constants. Perhaps for this reason, the ideas that underly it have been at the heart of published implementations of arguments.

The first claim that we are establishing is certainly not a strong PCP theorem. But the construction that we present highlights some important techniques. We are going through the construction in class, so here are just a few brief notes.

The constraint set $C$ is over variables in the set $X$ (corresponding to the inputs), the set $Y$ (corresponding to the outputs), and the set $Z$ (corresponding to “intermediate” variables and variables set non-deterministically). We label the constraints in $C$ as $Q_1, \ldots, Q_m$. Notice: each $Q_j$ is a degree-2 function of $(X, Y, Z)$ and that, at a particular $(X=x, Y=y)$, the $Q_j$ are functions only of $Z_1, \ldots, Z_n$. Denote $C(X=x, Y=y)$ with $C'$.

Let’s notate $Q(z)$ as $Q^{(\gamma)}(z)$ to make clearer that it is a random variable. The reason that $Q^{(\gamma)}(z)$ is a bellwether is as follows:

- If $Z = z$ satisfies $C'$, then $\Pr_{\gamma} \{ Q^{(\gamma)}(z) = 0 \} = 1$.
- If $Z = z$ does not satisfy $C'$, then $\Pr_{\gamma} \{ Q^{(\gamma)}(z) = 0 \} \leq 1/|\mathbb{F}|$.

The second one holds because at least one constraint $Q_{j'}(z)$ is not equal to 0. So the whole sum is equal to 0 only if: $\gamma_{j'} = ( - \sum_{j \neq j'} \gamma_j \cdot Q_j(z)) \cdot (Q_{j'}(z))^{-1}$, which has probability $1/|\mathbb{F}|$ (since $\gamma_{j'}$ is conceptually chosen after $z$).
There is a proof oracle such that $Q^{(\gamma)}(z)$ can be evaluated with two queries to that oracle (if it’s correctly constructed). To see this, write

$$Q^{(\gamma)}(z) = \langle \lambda_2, z \otimes z \rangle + \langle \lambda_1, z \rangle + \lambda_0,$$

where $v \otimes w$ is the outer product of two vectors, meaning all pairs $v_iw_j$ (how do we know that $Q^{(\gamma)}(z)$ has this form?). Notice that $\lambda_0, \lambda_1, \lambda_2$ depend on $\gamma$ and the structure of the $Q_j(\cdot)$. In addition, $\lambda_0$ depends on $x, y$ (but we can arrange for $\lambda_1$ and $\lambda_2$ not to have such a dependence).

The proof oracle is then, for some $z$, two long tables:

1. $\pi_1 = \langle u_1, z \rangle$ for all $u_1 \in \mathbb{F}^n$
2. $\pi_2 = \langle u_2, z \otimes z \rangle$ for all $u_2 \in \mathbb{F}^{n^2}$

Notice that $\pi_1, \pi_2$ can be thought of as linear functions.

**Exercise:** Convince yourself that if $\pi_1, \pi_2$ are constructed this way, $Q^{(\gamma)}(z)$ can be evaluated with only one query each to $\pi_1$ and $\pi_2$. 

But of course $\pi$ might be constructed adversarially. To address this issue, $\mathcal{V}$ will perform three tests:

- **Linearity tests of $\pi_1$ and $\pi_2$.** This consists of choosing two elements $a, b$ at random from the domain of a (purportedly) linear function $\pi$ and checking whether $\pi(a) + \pi(b) = \pi(a + b)$. (In a linear function, this holds for all $a, b$ in the domain.)

- **Quadratic test (self-corrected).** This consists of choosing two elements $q_5, q_6$ at random from the domain of $\pi_1$ and an element $q_7$ from the domain of $\pi_2$ and checking whether $\pi_1(q_5) \cdot \pi_1(q_6) = \pi_2(q_5 \otimes q_6 + q_7) - \pi_2(q_7)$.

- **Circuit test (self-corrected).** This is a modified form of our idealized check on the previous page. Choose random $q_8$ from the domain of $\pi_1$ and $q_9$ from the domain of $\pi_2$ and check whether
  
  $$\pi_2(\lambda_2 + q_9) - \pi_2(q_9) + \pi_1(\lambda_1 + q_8) - \pi_1(q_8) + \lambda_0 = 0.$$
The remaining step is to prove the completeness and soundness of this long PCP by analyzing the tests. Take the following as a given:

**Lemma ([BLR90, BGLR93, BCHKS96]):** If no linear function is \((1/10)\)-close to \(g\), then \(\Pr_{a,b}\{\text{linearity test passes}\} \leq 7/9\).

**Exercise:** Prove the following lemma: If \(\pi_1\) and \(\pi_2\) are \((1/10)\)-close to some linear functions \(\tilde{\pi}_1\) and \(\tilde{\pi}_2\) (respectively) and \(\Pr\{\pi_1, \pi_2\ \text{pass the quadratic test}\} > 4 \cdot (1/10) + \frac{2}{|F|}\) (the probability is taken over the random choices made in the quadratic test), then there exists a vector \(w\) for which \(\tilde{\pi}_1(\cdot) = \langle \cdot, w \rangle\) and \(\tilde{\pi}_2(\cdot) = \langle \cdot, w \otimes w \rangle\).

**Exercise:** Prove the following lemma: If \(\pi_1\) and \(\pi_2\) are \((1/10)\)-close to some linear functions \(\tilde{\pi}_1\) and \(\tilde{\pi}_2\) (respectively), if \(\tilde{\pi}_1\) and \(\tilde{\pi}_2\) are consistent (in the sense of encoding the same vector, \(z\)), and \(\Pr\{\pi_1, \pi_2\ \text{pass the circuit test}\} > 4 \cdot (1/10) + \frac{1}{|F|}\) (the probability is taken over the random choices made in the circuit test), then \(z\) is a satisfying assignment.

**Exercise:** Prove completeness and soundness:

- If \(C'\) is satisfiable, there are \(\pi_1, \pi_2\) s.t. \(\Pr\{\text{all tests pass}\} = 1\).
- If \(C'\) is not satisfiable, then for all \(\pi_1, \pi_2\), \(\Pr\{\text{all tests pass}\} \leq 7/9\).
Notes on linear commitment primitive

In the security proof for this primitive, we imagine running the “commitment” phase once and the “decommit” phase twice. If $\mathcal{P}$ could, with non-negligible probability, produce different answers for the same query in the two different decommit phases, then $\mathcal{P}$ could break the semantic security of the additively homomorphic encryption scheme.

This in turn means that the prover is bound to a single function after the commitment phase. The details are given in [IKO07], and the compiler given there is improved in [SMBW12].
References


