Doubly Efficient Interactive Proofs

Ron Rothblum
Outsourcing Computation

Weak client outsources computation to the cloud.

\[ \begin{align*}
    x \\
    y &= f(x)
\end{align*} \]
Outsourcing Computation

We do not want to blindly trust the cloud.

\[ x \quad y = f(x) \]

Key security concern:

**Correctness**: why should we trust the server’s answer?
Interactive Proofs to the Rescue?

**Interactive Proof** [GMR85]: prover $P$ tries to interactively convince a polynomial-time verifier $V$ that $f(x) = y$.

- $f(x) = y \implies P$ convinces $V$.
- $f(x) \neq y \implies$ no $P^*$ can convince $V$ wp $\geq 1/2$.

**Key Problem:** in classical results complexity of proving is actually exponential:

**IP=PSPACE** [LFKN90,Shamir90]: Interactive Proofs for space $S$ computations with $2^\text{poly}(S)$ prover, $\text{poly}(n,S)$ verification, $\text{poly}(S)$ rounds.
Doubly Efficient Interactive Proof
[GKR08]

Interactive proof for $f(x) = y$ where the prover is efficient, and the verifier is super efficient.

Proportional to complexity of $f$

Much faster than complexity of $f$

Soundness holds against any (computationally unbounded) cheating prover.
Why Proof and not Arguments*?

1. Security against *unbounded* adversary.
   - Post-quantum secure, post post quantum secure...

2. No reliance on unproven crypto assumptions

3. Do not use any expensive crypto operations
   - Even if not currently practical, no clear bottleneck (e.g., [GKR08])...

* Disclaimer: arguments are GREAT! (e.g., [KRR14])
Doubly Efficient Interactive Proofs: The State of the Art

1) [GKR08]: Bounded Depth
   • Any bounded-depth circuit.
   • (Almost) linear time verifier, poly-time prover.
   • Number of rounds proportional to circuit depth.

2) [RRR16]: Bounded Space
   • Any bounded-space computation.
   • (Almost) linear time verifier, poly-time prover.
   • $O(1)$ rounds.
Constant-Round Doubly Efficient Interactive Proofs

**Theorem** [RRR16]: $\exists \delta > 0$ s.t. every language computable in $\text{poly}(n)$ time and $n^\delta$ space has an unconditionally sound interactive proof where:

1. Verifier is (almost) linear time.
2. Prover is polynomial-time.
3. Constant number of rounds.
Define $\text{IP}_{\text{DE}}$ as class of languages having doubly efficient interactive proofs.
Roadmap: A Taste of the Proof

Iterative construction:

1. Start with interactive proof for short computations.
2. Build interactive proof for slightly longer computations.
3. Repeat.
Iterative Construction

Suppose we have interactive proofs for time $T/k$ and space $S$ computations.

Consider a time $T$ and space $S$ computation.
Divide & Conquer

**Divide:** Prover sends Turing machine configuration in $k \ll T$ intermediate steps.

**Conquer?** recurse on all subcomputations.

**Problem:** verification blows up, no savings.
Divide & Conquer

**Divide:** Prover sends Turing machine configuration in $k \ll T$ intermediate steps.

**Conquer?** Choose a few at random and recurse.

**Problem:** huge soundness error.
Best of Both Worlds?

Can we **batch verify** $k$ instances much more efficiently than $k$ independent executions.

**Goal:**
- Suppose $x \in L$ can be verified in time $t$.
- Want to verify $x_1, ..., x_k \in L$ in $\ll k \cdot t$ time.
Concrete Example: Batch Verification of RSA moduli

**Def:** integer $N$ is an *RSA modulos* if it is the product of two $m$-bit primes $N = p \cdot q$.

The proof that $N$ is an RSA modulos is its factorization. Can we verify $k$ RSA moduli more efficiently?

\[
P(p_1, q_1, \ldots, p_k, q_k) \quad V(N_1, \ldots, N_k)
\]

\[
\ll k \cdot m \quad \text{communication}
\]
Warmup: Batch Verification for UP

**UP ⊆ NP** are all relations with unique accepting witnesses.

**Theorem [RRR16]:** Every $L \in \text{UP}$, has an interactive proof for verifying that $x_1, \ldots, x_k \in L$ with $m \cdot \text{polylog}(k) + \tilde{O}(k)$ communication.

For batch verification of interactive proofs we introduce interactive analogs of **UP** and **PCP**. 

$m = \text{witness length}$
Constant-Round Doubly Efficient Interactive Proofs

**Theorem [RRR16]:** \( \exists \delta > 0 \) s.t. every language computable in \( \text{poly}(n) \) time and \( n^\delta \) space has an unconditionally sound interactive proof where:

1. Verifier is (almost) linear time.
2. Prover is polynomial-time.
3. Constant number of rounds.
Motivation: statistical analysis of vast amounts of data.
Sublinear Time Verification

Can we verify without even reading the input?

Yes! If we allow for approximation.

Following Property Testing [GGR98]: only required to reject inputs that are far from the language.
Sublinear Time Verification

Revisiting classical notions of proof-systems:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NP</strong></td>
<td>Gur-R13, Fischer-Goldhirsh-Lachish13, Goldreich-Gur-R15</td>
</tr>
<tr>
<td><strong>Zero-Knowledge</strong></td>
<td>Berman-R-Vaikuntanathan17</td>
</tr>
<tr>
<td><strong>PCP/MIP</strong></td>
<td>Ergun-Kumar-Rubinfeld04, Dinur-Reingold06, BenSasson-Goldreich-Harsha-Sudan-Vadhan06, Gur-Ramnarayan-R17</td>
</tr>
</tbody>
</table>
Open Problems

• **Research directions:**
  – Bridge theory and practice.
  – **Sublinear** time verification.

• **Concrete questions:**
  – IP=PSPACE with “efficient” prover.
  – Batch verification for all of NP.
  – [GR17]: Simpler and more efficient protocols (even for smaller classes).
  – Improve [RRR16] round complexity: even exponentially.