CompSci 501 Lecture 38: BPP, BPL, Arthur-Merlin Games and Shamir’s Theorem

Despite Ladner’s Theorem, there are very few natural problems that are:

- Known to be in NP, and
- Not known to be NP-complete, and
- Not known to be in P

Examples: Factoring natural numbers, Graph Isomorphism, Model Checking the $\mu$-Calculus

$$\text{PRIME} = \{ m \in \mathbb{N} \mid m \text{ is prime} \}$$

**Proposition 38.1** \( \overline{\text{PRIME}} \in \text{NP} \)

**Proof:** \( m \in \overline{\text{PRIME}} \iff m < 2 \lor \exists xy \ (1 < x < m \land x \cdot y = m) \)

**Question:** Is \( \text{PRIME} \in \text{NP} \)?

**Fact 38.2 (Fermat’s Little Thm)** Let \( p \) be prime and \( 0 < a < p \), then, \( a^{p-1} \equiv 1 \mod p \).

$$\mathbb{Z}_n^* = \{ a \in \{1, 2, \ldots, n-1\} \mid \text{GCD}(a, n) = 1 \}$$

\( \mathbb{Z}_n^* \) is the multiplicative group of integers mod \( n \) that are relatively prime to \( n \).

**Euler’s phi function:** \( \varphi(n) \overset{\text{def}}{=} |\mathbb{Z}_n^*| \)
**Prop:** If \( n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k} \) is the prime factorization of \( n \), then
\[
\varphi(n) = n(p_1 - 1)(p_2 - 1) \cdots (p_k - 1)/(p_1 p_2 \cdots p_k)
\]

**Euler’s Thm:** For any \( n \) and any \( a \in \mathbb{Z}_n^* \), \( a^{\varphi(n)} \equiv 1 \pmod{n} \).

**Fact:** Let \( p > 2 \) be prime. Then \( \mathbb{Z}_p^* \) is a cyclic group of order \( p - 1 \). That is,
\[
\mathbb{Z}_p^* = \{a, a^2, a^3, \ldots, a^{p-1}\}
\]

\( m \in \text{PRIME} \iff \exists a \in \mathbb{Z}_m^* (\text{ord}(a) = m - 1) \)

**Pratt’s Thm:** \( \text{PRIME} \in \text{NP} \).

**Proof:** Given \( m \),

1. Guess \( a, 1 < a < m \)
2. Check \( a^{m-1} \equiv 1 \pmod{m} \) by repeated squaring.
3. Guess prime factorization: \( m - 1 = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k} \)
4. Check for \( 1 \leq i \leq k \), \( a^{m-1/p_i} \not\equiv 1 \pmod{m} \)
5. Recursively check that \( p_1, p_2, \ldots, p_k \) are prime.

**Divide and Conquer NP Algorithm:**
\[
T(n) = O(n^2) + T(n - 1)
\]
\[
T(n) = O(n^3) \quad \square
\]

**Cor:** \( \text{PRIME} \) and \( \text{FACTORIZATION} \) are in \( \text{NP} \cap \text{co-NP} \).

**Proof:** \( \text{PRIME} \): immediately from Pratt’s Thm.

\( \text{FACTORIZATION} \) is the problem of given \( N \), find it’s prime factorization: \( N = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k} \).

Think of this as a decision problem by putting the factorization in a standard form, e.g., \( p_1 < p_2 < \cdots < p_k \), and
asking if bit $i$ of the factorization is “1”.

This is in $\text{NP} \cap \text{co-NP}$ because an NP or co-NP machine can guess the unique prime factorization, check that it is correct, and then read bit $i$. \qed

### More Primality Testing

$a \in \mathbb{Z}_m^*$ is a **quadratic residue** mod $m$ iff, $\exists b \ (b^2 \equiv a \ (\text{mod } m))$

For $p$ prime let,

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } a \text{ is a quadratic residue mod } p \\ -1 & \text{otherwise} \end{cases}$$

Generalize to $(\frac{a}{m})$ when $m$ is not prime,

$$(\frac{a}{mn}) = (\frac{a}{m})(\frac{a}{n})$$

$$(\frac{a}{m}) = (\frac{a \mod m}{m})$$

**Quadratic Reciprocity Thm:** [Gauss] For odd $a, m$,

$$\left(\frac{a}{m}\right) = \begin{cases} \left(\frac{m}{a}\right) & \text{if } a \equiv 1 \ (\text{mod } 4) \ or \ m \equiv 1 \ (\text{mod } 4) \\ -\left(\frac{m}{a}\right) & \text{if } a \equiv 3 \ (\text{mod } 4) \ and \ m \equiv 3 \ (\text{mod } 4) \end{cases}$$

$$\left(\frac{2}{m}\right) = \begin{cases} 1 & \text{if } m \equiv 1 \ (\text{mod } 8) \ or \ m \equiv 7 \ (\text{mod } 8) \\ -1 & \text{if } m \equiv 3 \ (\text{mod } 8) \ or \ m \equiv 5 \ (\text{mod } 8) \end{cases}$$

Thus, we can calculate $(\frac{a}{m})$ efficiently. For example,
\[
\begin{align*}
\left( \frac{107}{351} \right) &= -\left( \frac{351}{107} \right) = -\left( \frac{30}{107} \right) \\
&= -\left( \frac{2}{107} \right) \left( \frac{15}{107} \right) = -\left( \frac{107}{15} \right) \\
&= -\left( \frac{2}{15} \right) = -1
\end{align*}
\]

\[107 \equiv 351 \equiv 15 \equiv 3 \pmod{4}\]

\[107 \equiv 3 \pmod{8}; \quad 15 \equiv 7 \pmod{8}\]
Fact: [Gauss] For $p$ prime, $a \in \mathbb{Z}_p^*$, \( \left( \frac{a}{p} \right) \equiv a^{(p-1)/2} \pmod{p} \).

Fact: If $m$ not prime then,

\[ \left| \left\{ a \in \mathbb{Z}_m^* \mid \left( \frac{a}{m} \right) \equiv a^{(m-1)/2} \pmod{m} \right\} \right| < \frac{m-1}{2} \]

Solovay-Strassen Primality Algorithm:

1. Input is odd number $m$
2. For $i := 1$ to $k$ do {
3. choose $a < m$ at random
4. if $\text{GCD}(a, m) \neq 1$ return (“not prime”)
5. if \( \left( \frac{a}{m} \right) \not\equiv a^{(m-1)/2} \pmod{m} \) return (“not prime”)
6. }
7. return (“probably prime”)

Thm:

• If $m$ is prime then Solovay-Strassen($m$) returns “probably prime”.
• If $m$ is not prime, then the probability that Solovay-Strassen($m$) returns “probably prime” is less than $1/2^k$.

Cor: PRIME $\in$ “Truly Feasible”

Fact: [Agrawal, Kayal, and Saxena, 2002] PRIME $\in$ P

Def: A decision problem $S$ is in BPP (Bounded Probabilistic Polynomial Time) iff there is a probabilistic, polynomial-time algorithm $A$ such that for all inputs $w$,

\[
\text{if } (w \in S) \text{ then } \Pr(A(w) = 1) \geq \frac{2}{3} \\
\text{if } (w \notin S) \text{ then } \Pr(A(w) = 1) \leq \frac{1}{3}
\]
**Prop:** If \( S \in \text{BPP} \) then there is a probabilistic, polynomial-time algorithm \( A' \) such that for all \( n \) and all inputs \( w \) of length \( n \),

\[
\begin{align*}
\text{if } (w \in S) \text{ then } \text{Prob}(A'(w) = 1) & \geq 1 - \frac{1}{2^n} \\
\text{if } (w \notin S) \text{ then } \text{Prob}(A'(w) = 1) & \leq \frac{1}{2^n}
\end{align*}
\]

**Proof:** Iterate \( A \) polynomially many times and answer with the majority. Probability the mean is off by \( \frac{1}{3} \) decreases exponentially with \( n \) — Chernoff bounds.

Is BPP equal to P???

Probably, because pseudo-random number generators are good.

Is randomness ever useful?


Colonel Kelly:

Which base to inspect?

If we randomize, then our opponent cannot know what we will do.
UREACH = \{ G, \text{undirected} \mid s \xrightarrow{G} t \}\)

**Fact 38.3** Consider a random walk in a connected undirected graph $G$. Let $T(i)$ be the expected number of steps until we have reached all vertices, assuming we start at vertex $i$. Then, $T(i) \leq 2m(n - 1)$, where $n = |V|$, $m = |E|$.

**Corollary 38.4** $\text{UREACH} \in \text{BPL}$.

**Definition 38.5** A *universal traversal sequence* for graphs on $n$ nodes, is a sequence of instructions, $q = a_1a_2a_3 \cdots a_t \in \{1, \ldots, n - 1\}^*$, such that for any *undirected* graph on $n$ nodes, if we start at $s$ in $G$ and follow $q$, then we will visit every vertex in the connected component of $s$. □
Fact 38.6  Undirected graphs with \( n \) vertices have universal traversal sequences of length \( O(n^3) \).

Fact 38.7 (Reingold, 2004)  UREACH \( \in \) \( \mathbb{L} \)

Proof idea: derandomization of universal traversal sequences using expander graphs.

Corollary 38.8  Symmetric-\( \mathbb{L} \) = \( \mathbb{L} \)
Recall From Last Time
Factoring natural numbers $\in \text{NP} \cap \text{co-NP}$

Definition of BPP and BPL

**Thm:** PRIME $\in$ BPP

**Thm:** UREACH $\in$ BPL

**Newish Results:**

- PRIME $\in$ P  [Agrawal, Kayal, and Saxena, 2002]
- UREACH $\in$ L  [Reingold, 2004]
One-Time Pad: \( p \in \{0, 1\}^n; \quad m \in \{0, 1\}^n \)

\[
E(p, x) = p \oplus x
\]

\[
D(p, x) = p \oplus x
\]

\[
D(p, E(p, m)) = p \oplus (p \oplus m) = m
\]
One-Time Pad, Continued

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( m )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( E(p, m) )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( D(p, E(p, m)) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Thm:** If \( p \) is chosen at random and known only to \( A \) and \( B \) Then \( E(p, m) \) provides no information to \( E \) about \( m \) except perhaps its length.

**Better not use \( p \) more than once!**
Public-Key Cryptography

**Idea:** [Diffie, Hellman, 1976] Using computational complexity, I may be able to publish a key for sending secret messages to me, that are intractable to decode. Example: Diffie-Hellman key exchange.

**Realization:** [Rivest, Shamir, Adleman, 1976] This is the Public-Key Algorithm that is used today in the SSL algorithm that lets your browser generate a key to send an order to Amazon.com without, *we believe*, divulging any useful information about your credit card number, or what you bought.
RSA

B chooses \( p, q \) \( n \)-bit primes, \( e \), s.t. \( \text{GCD}(e, \varphi(pq)) = 1 \);

B publishes: \( pq, e \); keeps \( p, q \) secret.

Using Euclid’s algorithm, B computes \( d, k \), s.t. \( ed + k\varphi(pq) = 1 \)

[Break message into pieces shorter than \( 2n \) bits]

\[
\begin{align*}
E_B(x) & \equiv x^e \pmod{pq} \\
D_B(x) & \equiv x^d \pmod{pq} \\
D_B(E_B(m)) & \equiv (m^e)^d \pmod{pq} \\
& \equiv m^{1-k\varphi(pq)} \pmod{pq} \\
& \equiv m \cdot (m^{\varphi(pq)})^{-k} \pmod{pq} \\
& \equiv m \pmod{pq} \\
& \equiv E_B(D_B(m)) \pmod{pq}
\end{align*}
\]
For **sufficiently large** $n$,   \[ n \geq 300 \text{ bits is fine in 2005}, \]

**It is widely believed that:** $E_B(m)$ divulges no useful information about $m$ to anyone not knowing $p, q$, or $d$.

**Message signing:**

Let $m = "B$ promises to give $A$ $10 by 5/17/05."$

Let $m' = m \circ r$ where $r$ is nonce or current date and time

**It is widely believed that:** $D_B(m')$ could be produced only by $B$. Thus it can be used as a contract signed by $B$.

**Useful for proving authenticity**
Interactive Proofs

[Goldwasser, Micali, Rackoff], [Babai]

Decision problem: $D$; input string: $x$

Two players:

**Prover — Merlin** is computationally all-powerful. Wants to convince **Verifier** that $x \in D$.

**Verifier — Arthur**: probabilistic polynomial-time TM. Wants to know the truth about whether $x \in D$. 
Input = $x$; \; \; n = |x|; \; \; t = n^{O(1)}

0. \textbf{Arthur} has $x$ \quad \textbf{Merlin} has $x$

1. flip $\sigma_1$, compute $m_1 \rightarrow$

2. \quad $\leftarrow m_2$

3. flip $\sigma_3$, compute $m_3 \rightarrow$

4. \quad $\leftarrow m_4$

: \quad : \quad : \quad :

2$t$.

\quad $\leftarrow m_{2t}$

$2t + 1$. \; \text{flip $\sigma_{2t+1}$, accept or reject}
**Def:** $D \in \text{IP}$ iff there is a PTIME interactive protocol

1. If $x \in D$, then there exists a strategy for Merlin

   $$\text{Prob}\{\text{Arthur accepts } x\} > \frac{2}{3}$$

2. If $x \notin D$, then for all strategies for Merlin

   $$\text{Prob}\{\text{Arthur accepts } x\} < \frac{1}{3}$$

**Observation:** As for BPP, by iterating we can make probability of error exponentially small.
**Def:** MA is the set of decision problems admitting two step proofs where Merlin moves first.

AM is the set of decision problems admitting two step proofs where Arthur moves first. For $k \geq 2$,

$$\text{AM}[k] = \underbrace{\text{ArthurMerlinArthur} \cdots}_{k}$$

□

**Fact:** [Babai] For all $k \geq 2$, $\text{AM}[k] = \text{AM}$. 
PSPACE \equiv \text{P} \quad \text{BPP} \quad \text{NP} \quad \text{MA} \quad \text{AM} \quad \text{AM[\text{poly}]} \quad \text{PSPACE}

\text{MA} \equiv \text{AM}

\text{BP(NP)}
Fact: [Goldwasser & Sipser] The power of interactive proofs is unchanged if Merlin knows Arthur’s coin tosses. For all $k$,

- $\text{IP}[k] = \text{AM}[k]$

- $\text{IP} = \text{AM}[n^{O(1)}]$
Graph Isomorphism $\in$ NP; Is it in co-NP?

Input $= G_0, G_1, \quad n = \|G_0\| = \|G_1\|

0. Arthur has $G_0, G_1$\quad Merlin has $G_0, G_1$

1. flip $\kappa : \{1, \ldots, r\} \rightarrow \{0, 1\}$
   flip $\pi_1, \ldots, \pi_r \in S_n$
   $\pi_1(G_{\kappa(1)}), \ldots, \pi_r(G_{\kappa(r)})$ $\rightarrow$

2. $\leftarrow m_2 \in \{0, 1\}^r$

3. accept iff $\kappa = m_2$

**Prop**: Graph Isomorphism $\in$ co-AM

**proof**: If $G_0 \not\cong G_1$, then Arthur will accept with probability 1.

If $G_0 \cong G_1$, then Arthur will accept with probability $\leq 2^{-r}$. Q.E.D.
Shamir’s Thm: \( \text{IP} = \text{PSPACE} \)

**proof that** \( \text{IP} \subseteq \text{PSPACE} \): Evaluate the game tree.

For *Merlin*’s moves choose the maximum value.

For *Arthur*’s moves choose the average value.
Show QSAT ∈ IP

\[ \varphi \equiv \forall x \exists y (x \lor y) \land \forall z ((x \land z) \lor (y \land \overline{z})) \lor \exists w (z \lor (y \land \overline{w})) \]

Formula \( \varphi \) is simple iff no occurrence of a variable is separated by more than one universal quantifier from its point of quantification.

**Lemma 38.9** *Any quantified boolean formula can be transformed in logspace to an equivalent, simple formula.*

**proof:** Suppose \( x \) is quantified before \( \forall y \) and used after \( \forall y \)

\[ \varphi = \cdots Qx \cdots \forall y \psi(x) \]

Right after the \( \forall y \), rename \( x \),

\[ \varphi' = \cdots Qx \cdots \forall y \exists x'((x \land x') \lor (\overline{x} \land \overline{x'})) \land \psi(x') \]

This needs to be done fewer than \( |\varphi|^2 \) times. \( \square \)

From now on we may assume that \( \varphi \) is simple and all \( \neg \)'s are pushed all the way inside.
Arithmetization of formulas

Define $f : \text{boolean formulas} \rightarrow \text{polynomials}$.

$x = 1$ means $x$ is true; $x = 0$ means $x$ is false.

\[
\begin{align*}
    f(\overline{x}) &= 1 - x \\
    f(\alpha \land \beta) &= f(\alpha) \cdot f(\beta) \\
    f(\alpha \lor \beta) &= f(\alpha) + f(\beta) \\
    f(\forall x(\alpha(x))) &= \prod_{i=0}^{1} f(\alpha(i)) \\
    f(\exists x(\alpha(x))) &= \sum_{i=0}^{1} f(\alpha(i))
\end{align*}
\]

Lemma 38.10 Let $\varphi$ be a closed, quantified boolean formula with all “$\neg$”s pushed to variables. Then,

$$\varphi \in \text{QSAT} \iff f(\varphi) > 0$$
M must prove to A that \( f(\varphi) > 0 \)

**Lemma 38.11** Let \( n = |\varphi| \). If \( f(\varphi) \neq 0 \), then there is a prime \( p \), \( 2^n < p < 2^{3n} \) s.t.

\[
f(\varphi) \not\equiv 0 \pmod{p}
\]

M must prove to A that \( f(\varphi) \not\equiv 0 \pmod{p} \)

At step 1, M sends \( p \) to A and says,

“I will now prove to you that \( f(\varphi) \not\equiv 0 \pmod{p} \)!”
\[ \varphi \equiv \forall x \exists y (x \lor y) \land \forall z ((x \land z) \lor (y \land \bar{z})) \lor \exists w (z \lor (y \land \bar{w})) \]

\[
f(\varphi) = \prod_x \sum_y ((x + y) \cdot \prod_z ((x \cdot z) + (y \cdot (1 - z))) + \sum_w (z + (y \cdot (1 - w)))
\]

\[
f_1(x) = \sum_y ((x + y) \cdot \prod_z ((x \cdot z) + (y \cdot (1 - z))) + \sum_w (z + (y \cdot (1 - w)))
\]

\[
= 2x^2 + 8x + 6
\]

Note, \( f_1 \in \mathbb{Z}[x] \) has degree \( \leq 2n \) because \( \varphi \) is simple. There is at most one “\( \prod \)” affecting \( x \).

\[
f(\varphi) = f_1(0) \cdot f_1(1)
\]

\[
96 = 6 \cdot 16
\]
\[ \varphi = (\forall x)(\exists y)\psi \]

\[ f(\varphi) = \prod_{x=0}^{1} f_1(x) \]

1. M sends to A:
   - \( p \)
   - a proof that \( p \) is prime
   - \( v_0 \) where \( v_0 \equiv f(\varphi) \pmod{p} \)
   - coefficients of \( g_1 \), where \( g_1 \equiv f_1 \pmod{p} \)

2. A
   - checks that \( g_1(0) \cdot g_1(1) \equiv v_0 \pmod{p} \)
   - chooses random \( r_1 \in \mathbb{Z}_p \)
   - computes \( v_1 \equiv g_1(r_1) \pmod{p} \)
   - sends \( r_1 \) to M

   **M must prove to A that** \( f_1(r_1) \equiv v_1 \pmod{p} \)
M must prove to A that $f_1(r_1) \equiv v_1 \pmod{p}$

**proof:** If $g_1 \not\equiv f_1 \pmod{p}$, then

$$\Pr[g_1(r_1) \equiv f_1(r_1) \pmod{p}] \leq \frac{2n}{p} < \frac{2n}{2^n}$$

Proof: Since $g_1$ and $f_1$ each have degree $2n$, so does $g_1 - f_1$.

But a degree $d$ polynomial has at most $d$ zeros.

Thus, with $r$ chosen at random,

$$\Pr[(g_1 - f_1)(r) \equiv 0 \pmod{p}] \leq \frac{2n}{p} \quad \Box$$

Thus, in one double round, we have removed one quantifier from $\varphi$.

**Key idea:** replace the universal boolean quantifier:

$$\forall x(f_1(x) = g_1(x))$$

with a random quantifier

$$(\text{for most } r)(f_1(r) = g_1(r))$$
M must prove to A that $f_1(r_1) \equiv v_1 \pmod{p}$
\[ \varphi = (\forall x)(\exists y)\psi \]
\[ f(\varphi) = \prod_{x=0}^{1} f_1(x) \]
\[ f_1(r_1) = \sum_{y=0}^{1} f_2(y) \]

3. M sends to A:
   - coefficients of \( g_2 \), where \( g_2 \equiv f_2 \pmod{p} \)

4. A
   - checks that \( g_2(0) + g_2(1) \equiv v_1 \pmod{p} \)
   - chooses random \( r_2 \in \mathbb{Z}_p \)
   - computes \( v_2 \equiv g_2(r_2) \pmod{p} \)
   - sends \( r_2 \) to M

**M must prove to A that** \( f_2(r_2) \equiv v_2 \pmod{p} \)
After $n$ steps, all the variables are eliminated and $A$ should accept iff $f_n(r_n) = v_n$.

The probability of $M$ getting away with a lie is at most $n \left( \frac{2n}{2^n} \right)$.

Shamir’s Theorem is proved.