The Jacobi Symbol\textsuperscript{a}

- The Legendre symbol only works for odd \textit{prime} moduli.
- The \textbf{Jacobi symbol} \((a \mid m)\) extends it to cases where \(m\) is not prime.
  - \(a\) is sometimes called the \textit{numerator} and \(m\) the \textit{denominator}.
- Trivially, \((1 \mid m) = 1\).
- Define \((a \mid 1) = 1\).

\textsuperscript{a}Carl Jacobi (1804–1851).
The Jacobi Symbol (concluded)

- Let \( m = p_1 p_2 \cdots p_k \) be the prime factorization of \( m \).
- When \( m > 1 \) is odd and \( \gcd(a, m) = 1 \), then

\[
(a \mid m) \triangleq \prod_{i=1}^{k} (a \mid p_i).
\]

- Note that the Jacobi symbol equals \( \pm 1 \).
- It reduces to the Legendre symbol when \( m \) is a prime.
Properties of the Jacobi Symbol

The Jacobi symbol has the following properties when it is defined.

1. \((ab \mid m) = (a \mid m)(b \mid m)\).

2. \((a \mid m_1 m_2) = (a \mid m_1)(a \mid m_2)\).

3. If \(a \equiv b \text{ mod } m\), then \((a \mid m) = (b \mid m)\).

4. \((-1 \mid m) = (-1)^{(m-1)/2}\) (by Lemma 70 on p. 581).

5. \((2 \mid m) = (-1)^{(m^2-1)/8}\).

6. If \(a\) and \(m\) are both odd, then
   \[(a \mid m)(m \mid a) = (-1)^{(a-1)(m-1)/4}\].

\(^a\)By Lemma 70 (p. 581) and some parity arguments.
Properties of the Jacobi Symbol (concluded)

• Properties 3–6 allow us to calculate the Jacobi symbol \( \left( \frac{a}{m} \right) \) without factorization.
  
  – It will also yield the same result as Euler’s test\(^a\) when \( m \) is an odd prime.

• This situation is similar to the Euclidean algorithm.

• Note also that \( (a \mid m) = 1/(a \mid m) \) because \( (a \mid m) = \pm 1 \).\(^b\)

\(^a\)Recall p. 573.
\(^b\)Contributed by Mr. Huang, Kuan-Lin (B96902079, R00922018) on December 6, 2011.
Calculation of $(2200 \mid 999)$

$$(2200 \mid 999) = (202 \mid 999)$$

$$= (2 \mid 999)(101 \mid 999)$$

$$= (-1)^{(999^2 - 1)/8}(101 \mid 999)$$

$$= (-1)^{124750}(101 \mid 999) = (101 \mid 999)$$

$$= (-1)^{(100)(998)/4}(999 \mid 101) = (-1)^{24950}(999 \mid 101)$$

$$= (999 \mid 101) = (90 \mid 101) = (-1)^{(101^2 - 1)/8}(45 \mid 101)$$

$$= (-1)^{1275}(45 \mid 101) = -(45 \mid 101)$$

$$= -(-1)^{(44)(100)/4}(101 \mid 45) = -(101 \mid 45) = -(11 \mid 45)$$

$$= -(-1)^{(10)(44)/4}(45 \mid 11) = -(45 \mid 11)$$

$$= -(1 \mid 11) = -1.$$
A Result Generalizing Proposition 10.3 in the Textbook

**Theorem 72** The group of set $\Phi(n)$ under multiplication mod $n$ has a primitive root if and only if $n$ is either 1, 2, 4, $p^k$, or $2p^k$ for some nonnegative integer $k$ and an odd prime $p$.

This result is essential in the proof of the next lemma.
Lemma 73 If \((M \mid N) \equiv M^{(N-1)/2} \mod N\) for all \(M \in \Phi(N)\), then \(N\) is a prime. (Assume \(N\) is odd.)

- Assume \(N = mp\), where \(p\) is an odd prime, \(\gcd(m, p) = 1\), and \(m > 1\) (not necessarily prime).
- Let \(r \in \Phi(p)\) such that \((r \mid p) = -1\).
- The Chinese remainder theorem says that there is an \(M \in \Phi(N)\) such that

\[
\begin{align*}
M & \equiv r \mod p, \\
M & \equiv 1 \mod m.
\end{align*}
\]

---

\(^a\)Mr. Clement Hsiao (B4506061, R88526067) pointed out that the textbook’s proof for Lemma 11.8 is incorrect in January 1999 while he was a senior.
The Proof (continued)

• By the hypothesis,

\[ M^{(N-1)/2} = (M \mid N) = (M \mid p)(M \mid m) = -1 \mod N. \]

• Hence

\[ M^{(N-1)/2} = -1 \mod m. \]

• But because \( M = 1 \mod m, \)

\[ M^{(N-1)/2} = 1 \mod m, \]

a contradiction.
The Proof (continued)

• Second, assume that $N = p^a$, where $p$ is an odd prime and $a \geq 2$.

• By Theorem 72 (p. 596), there exists a primitive root $r$ modulo $p^a$.

• From the assumption,

$$M^{N-1} = \left[ M^{(N-1)/2} \right]^2 = (M|N)^2 = 1 \mod N$$

for all $M \in \Phi(N)$.
The Proof (continued)

• As $r \in \Phi(N)$ (prove it), we have

$$r^{N-1} = 1 \mod N.$$ 

• As $r$’s exponent modulo $N = p^a$ is $\phi(N) = p^{a-1}(p - 1)$,

$$p^{a-1}(p - 1) \mid (N - 1),$$

which implies that $p \mid (N - 1)$.

• But this is impossible given that $p \mid N$.

\(^a\text{For } p - 1 \text{ divides } N - 1 = p^a - 1.$$
• Third, assume that $N = mp^a$, where $p$ is an odd prime, $\gcd(m, p) = 1$, $m > 1$ (not necessarily prime), and $a$ is even.

• The proof mimics that of the second case.

• By Theorem 72 (p. 596), there exists a primitive root $r \mod p^a$.

• From the assumption,

$$M^{N-1} = \left[M^{(N-1)/2}\right]^2 = (M|N)^2 = 1 \mod N$$

for all $M \in \Phi(N)$. 
The Proof (continued)

• In particular,

\[ M^{N-1} = 1 \mod p^a \]  \hspace{1cm} (15)

for all \( M \in \Phi(N) \).

• The Chinese remainder theorem says that there is an \( M \in \Phi(N) \) such that

\[ M = r \mod p^a, \]
\[ M = 1 \mod m. \]

• Because \( M = r \mod p^a \) and Eq. (15),

\[ r^{N-1} = 1 \mod p^a. \]
The Proof (concluded)

• As $r$’s exponent modulo $N = p^a$ is $\phi(N) = p^{a-1}(p - 1)$,

$$p^{a-1}(p - 1) \mid (N - 1),$$

which implies that $p \mid (N - 1)$.

• But this is impossible given that $p \mid N$. 
The Number of Witnesses to Compositeness

**Theorem 74 (Solovay & Strassen, 1977)** If $N$ is an odd composite, then $(M \mid N) \equiv M^{(N-1)/2} \mod N$ for at most half of $M \in \Phi(N)$.

- By Lemma 73 (p. 597) there is at least one $a \in \Phi(N)$ such that $(a \mid N) \not\equiv a^{(N-1)/2} \mod N$.
- Let $B \triangleq \{ b_1, b_2, \ldots, b_k \} \subseteq \Phi(N)$ be the set of all distinct residues such that $(b_i \mid N) \equiv b_i^{(N-1)/2} \mod N$.
- Let $aB \triangleq \{ ab_i \mod N : i = 1, 2, \ldots, k \}$.
- Clearly, $aB \subseteq \Phi(N)$, too.
The Proof (concluded)

- $|aB| = k$.

  $ab_i \equiv ab_j \mod N$ implies $N | a(b_i - b_j)$, which is impossible because $\gcd(a, N) = 1$ and $N > |b_i - b_j|$.

- $aB \cap B = \emptyset$ because

  $\left(ab_i\right)^{(N-1)/2} \mod 2 = a^{(N-1)/2}b_i^{(N-1)/2} \mod 2$

  $\neq (a | N)(b_i | N) = (ab_i | N)$.

- Combining the above two results, we know

  $$\frac{|B|}{\phi(N)} \leq \frac{|B|}{|B \cup aB|} = 0.5.$$
1: if $N$ is even but $N \neq 2$ then
2:    return “$N$ is composite”;
3: else if $N = 2$ then
4:    return “$N$ is a prime”;
5: end if
6: Pick $M \in \{2, 3, \ldots, N - 1\}$ randomly;
7: if gcd($M, N$) > 1 then
8:    return “$N$ is composite”;
9: else
10:   if $(M | N) \equiv M^{(N-1)/2} \mod N$ then
11:      return “$N$ is (probably) a prime”;
12:   else
13:      return “$N$ is composite”;
14: end if
15: end if
Analysis

- The algorithm certainly runs in polynomial time.
- There are no false positives (for COMPOSITENESS).
  - When the algorithm says the number is composite, it is always correct.
Analysis (concluded)

- The probability of a false negative (again, for COMPOSITENESS) is at most one half.
  - Suppose the input is composite.
  - By Theorem 74 (p. 604),
    \[
    \text{prob[algorithm answers “no” | } N \text{ is composite}] \leq 0.5.
    \]
  - Note that we are not referring to the probability that \( N \) is composite when the algorithm says “no.”
- So it is a Monte Carlo algorithm for COMPOSITENESS\(^a\) by the definition on p. 551.

\(^a\text{Not primes.}\)
The Improved Density Attack for \textit{COMPOSITENESS}

- All numbers \(< N\)
- Witnesses to compositeness of \(N\) via common factor
- Witnesses to compositeness of \(N\) via Jacobi
Randomized Complexity Classes; RP

- Let $N$ be a polynomial-time precise NTM that runs in time $p(n)$ and has 2 nondeterministic choices at each step.

- $N$ is a **polynomial Monte Carlo Turing machine** for a language $L$ if the following conditions hold:
  - If $x \in L$, then at least half of the $2^{p(n)}$ computation paths of $N$ on $x$ halt with “yes” where $n = |x|$.
  - If $x \notin L$, then all computation paths halt with “no.”

- The class of all languages with polynomial Monte Carlo TMs is denoted $\textbf{RP}$ (**randomized polynomial time**).\(^a\)

\(^a\)Adleman & Manders (1977).
Comments on RP

• In analogy to Proposition 41 (p. 344), a “yes” instance of an RP problem has many certificates (witnesses).

• There are no false positives.

• If we associate nondeterministic steps with flipping fair coins, then we can phrase RP in the language of probability.
  - If $x \in L$, then $N(x)$ halts with “yes” with probability at least 0.5.
  - If $x \notin L$, then $N(x)$ halts with “no.”
Comments on RP (concluded)

• The probability of false negatives is $\leq 0.5$.

• But any constant $\epsilon$ between 0 and 1 can replace 0.5.
  - Repeat the algorithm
    \[
    k \triangleq \left\lceil -\frac{1}{\log_2 \epsilon} \right\rceil
    \]
    times.
  - Answer “no” only if all the runs answer “no.”
  - The probability of false negatives becomes $\epsilon^k \leq 0.5$. 
Where RP Fits

• $P \subseteq \text{RP} \subseteq \text{NP}$.
  
  - A deterministic TM is like a Monte Carlo TM except that all the coin flips are ignored.
  
  - A Monte Carlo TM is an NTM with more demands on the number of accepting paths.

• $\text{COMPOSITENESS} \in \text{RP};^{a} \text{PRIMES} \in \text{coRP};$
  
  - In fact, $\text{PRIMES} \in P$.\(^{b}\)

• $\text{RP} \cup \text{coRP}$ is an alternative “plausible” notion of efficient computation.

\(^{a}\)Rabin (1976); Solovay & Strassen (1977).
\(^{b}\)Adleman & Huang (1987).
\(^{c}\)Agrawal, Kayal, & Saxena (2002).
ZPP\textsuperscript{a} (Zero Probabilistic Polynomial)

- The class ZPP is defined as RP \cap coRP.
- A language in ZPP has \textit{two} Monte Carlo algorithms, one with no false positives (RP) and the other with no false negatives (coRP).
- If we repeatedly run both Monte Carlo algorithms, \textit{eventually} one definite answer will come (unlike RP).
  - A \textit{positive} answer from the one without false positives.
  - A \textit{negative} answer from the one without false negatives.

\textsuperscript{a}Gill (1977).
The ZPP Algorithm (Las Vegas)

1: \{Suppose \( L \in \text{ZPP}. \}\}
2: \{\( N_1 \) has no false positives, and \( N_2 \) has no false negatives.\}
3: \textbf{while} true \textbf{do}
4: \quad \textbf{if} \( N_1(x) = \text{“yes”} \) \textbf{then}
5: \quad \quad \textbf{return} \text{“yes”};
6: \quad \textbf{end if}
7: \quad \textbf{if} \( N_2(x) = \text{“no”} \) \textbf{then}
8: \quad \quad \textbf{return} \text{“no”};
9: \quad \textbf{end if}
10: \textbf{end while}
ZPP (concluded)

- The *expected* running time for the correct answer to emerge is polynomial.
  - The probability that a run of the 2 algorithms does not generate a definite answer is 0.5 (why?).
  - Let $p(n)$ be the running time of each run of the while-loop.
  - The expected running time for a definite answer is
    $$\sum_{i=1}^{\infty} 0.5^i p(n) = 2p(n).$$

- Essentially, ZPP is the class of problems that can be solved, without errors, in expected polynomial time.
Large Deviations

• Suppose you have a *biased* coin.

• One side has probability $0.5 + \epsilon$ to appear and the other $0.5 - \epsilon$, for some $0 < \epsilon < 0.5$.

• But you do not know which is which.

• How to decide which side is the more likely side—with high confidence?

• Answer: Flip the coin many times and pick the side that appeared the most times.

• Question: Can you quantify your confidence?
The (Improved) Chernoff Bound\(^a\)

**Theorem 75 (Chernoff, 1952)** Suppose \(x_1, x_2, \ldots, x_n\) are independent random variables taking the values 1 and 0 with probabilities \(p\) and \(1 - p\), respectively. Let \(X = \sum_{i=1}^{n} x_i\).

Then for any constant \(0 \leq \theta \leq 1\),

\[
\text{prob}[X \geq (1 + \theta)pn] \leq e^{-\theta^2 pn/3}.
\]

- The probability that the deviate of a **binomial random variable** from its expected value 
  \(E[X] = E[\sum_{i=1}^{n} x_i] = pn\) decreases exponentially with the deviation.

\(^a\)Herman Chernoff (1923–). This bound is asymptotically optimal. The original bound is \(e^{-2\theta^2 p^2 n}\) (McDiarmid, 1998).
The Proof

• Let $t$ be any positive real number.

• Then

$$\text{prob}[X \geq (1 + \theta) pn] = \text{prob}[e^{tX} \geq e^{t(1+\theta)pn}].$$

• Markov’s inequality (p. 554) generalized to real-valued random variables says that

$$\text{prob}\left[e^{tX} \geq kE[e^{tX}]\right] \leq 1/k.$$  

• With $k = e^{t(1+\theta)pn}/E[e^{tX}]$, we have\(^a\)

$$\text{prob}[X \geq (1 + \theta) pn] \leq e^{-t(1+\theta)pn} E[e^{tX}].$$

\(^a\)Note that $X$ does not appear in $k$. Contributed by Mr. Ao Sun (R05922147) on December 20, 2016.
The Proof (continued)

• Because \( X = \sum_{i=1}^{n} x_i \) and \( x_i \)'s are independent,

\[
E[e^{tX}] = (E[e^{tx_1}])^n = [1 + p(e^t - 1)]^n.
\]

• Substituting, we obtain

\[
\text{prob}[X \geq (1 + \theta) pn] \leq e^{-t(1+\theta) pn} [1 + p(e^t - 1)]^n \\
\leq e^{-t(1+\theta) pn} e^{pn(e^t - 1)}
\]

as \((1 + a)^n \leq e^{an}\) for all \(a > 0\).
The Proof (concluded)

• With the choice of $t = \ln(1 + \theta)$, the above becomes

$$\text{prob}[X \geq (1 + \theta)pn] \leq e^{pn[\theta - (1 + \theta)\ln(1 + \theta)]}.$$ 

• The exponent expands to\(^a\)

$$-\frac{\theta^2}{2} + \frac{\theta^3}{6} - \frac{\theta^4}{12} + \cdots$$

for $0 \leq \theta \leq 1$.

• But it is less than

$$-\frac{\theta^2}{2} + \frac{\theta^3}{6} \leq \theta^2 \left( -\frac{1}{2} + \frac{\theta}{6} \right) \leq \theta^2 \left( -\frac{1}{2} + \frac{1}{6} \right) = -\frac{\theta^2}{3}.$$ 

\(^a\)Or McDiarmid (1998): $x - (1 + x)\ln(1 + x) \leq -3x^2/(6 + 2x)$ for all $x \geq 0$. 
How Good Is the Bound?

Chernoff bound

true probability

$n$
Other Variations of the Chernoff Bound

The following can be proved similarly (prove it).

**Theorem 76** Given the same terms as Theorem 75 (p. 618),

\[ \Pr[X \leq (1 - \theta) pn] \leq e^{-\theta^2 pn/2}. \]

The following slightly looser inequalities achieve symmetry.

**Theorem 77 (Karp, Luby, & Madras, 1989)** Given the same terms as Theorem 75 (p. 618) except with \( 0 \leq \theta \leq 2 \),

\[ \Pr[X \geq (1 + \theta) pn] \leq e^{-\theta^2 pn/4}, \]
\[ \Pr[X \leq (1 - \theta) pn] \leq e^{-\theta^2 pn/4}. \]
Power of the Majority Rule

The next result follows from Theorem 76 (p. 623).

**Corollary 78** If \( p = (1/2) + \epsilon \) for some \( 0 \leq \epsilon \leq 1/2 \), then

\[
\text{prob} \left[ \sum_{i=1}^{n} x_i \leq n/2 \right] \leq e^{-\epsilon^2 n/2}.
\]

- The textbook’s corollary to Lemma 11.9 seems too loose, at \( e^{-\epsilon^2 n/6} \).

- Our original problem (p. 617) hence demands, e.g.,

\[ n \approx 1.4k/\epsilon^2 \]

independent coin flips to guarantee making an error with probability \( \leq 2^{-k} \) with the majority rule.

---

\(^{a}\)See Dubhashi & Panconesi (2012) for many Chernoff-type bounds.
BPP\(^a\) (Bounded Probabilistic Polynomial)

- The class \textbf{BPP} contains all languages \(L\) for which there is a precise polynomial-time NTM \(N\) such that:
  - If \(x \in L\), then at least \(3/4\) of the computation paths of \(N\) on \(x\) lead to “yes.”
  - If \(x \not\in L\), then at least \(3/4\) of the computation paths of \(N\) on \(x\) lead to “no.”

- So \(N\) accepts or rejects by a \emph{clear} majority.

\(^a\)Gill (1977).
Magic 3/4?

- The number 3/4 bounds the probability (ratio) of a right answer away from 1/2.

- Any constant \textit{strictly} between 1/2 and 1 can be used without affecting the class BPP.

- In fact, as with RP,

\[
\frac{1}{2} + \frac{1}{q(n)}
\]

for any polynomial \( q(n) \) can replace 3/4.

- The next algorithm shows why.
The Majority Vote Algorithm

Suppose $L$ is decided by $N$ by majority $(1/2) + \epsilon$.

1: \textbf{for} $i = 1, 2, \ldots, 2k + 1$ \textbf{do}
2: \hspace{1em} Run $N$ on input $x$;
3: \textbf{end for}
4: \textbf{if} “yes” is the majority answer \textbf{then}
5: \hspace{1em} “yes”;
6: \textbf{else}
7: \hspace{1em} “no”;
8: \textbf{end if}
Analysis

• By Corollary 78 (p. 624), the probability of a false answer is at most $e^{-\epsilon^2 k}$.

• By taking $k = \lceil 2/\epsilon^2 \rceil$, the error probability is at most $e^{-2} < 1/4$.

• Even if $\epsilon$ is any inverse polynomial, $k$ remains a polynomial in $n$.

• The running time remains polynomial: $2k + 1$ times $N$’s running time.
Aspects of BPP

- BPP is the most comprehensive yet plausible notion of efficient computation.
  - If a problem is in BPP, we take it to mean that the problem can be solved efficiently.
  - In this aspect, BPP has effectively replaced P.
- \((\text{RP} \cup \text{coRP}) \subseteq (\text{NP} \cup \text{coNP})\).
- \((\text{RP} \cup \text{coRP}) \subseteq \text{BPP}\).
- Whether \(\text{BPP} \subseteq (\text{NP} \cup \text{coNP})\) is unknown.
- But it is unlikely that \(\text{NP} \subseteq \text{BPP}\).\(^a\)

\(^a\)See p. 641.
coBPP

- The definition of BPP is symmetric: acceptance by clear majority and rejection by clear majority.
- An algorithm for $L \in \text{BPP}$ becomes one for $\overline{L}$ by reversing the answer.
- So $\overline{L} \in \text{BPP}$ and $\text{BPP} \subseteq \text{coBPP}$.
- Similarly $\text{coBPP} \subseteq \text{BPP}$.
- Hence $\text{BPP} = \text{coBPP}$.
- This approach does not work for RP.\(^a\)

\(^a\)It did not work for NP either.
BPP and coBPP

"yes"  "no"

"no"  "yes"
“The Good, the Bad, and the Ugly”

Diagram:

- BPP
- P
- RP
- ZPP
- coRP
- coNP
- NP
- ZPP
- P
- RP
- BPP
Circuit Complexity

• Circuit complexity is based on boolean circuits instead of Turing machines.

• A boolean circuit with \( n \) inputs computes a boolean function of \( n \) variables.

• Now, identify true/1 with “yes” and false/0 with “no.”

• Then a boolean circuit with \( n \) inputs accepts certain strings in \( \{0, 1\}^n \).

• To relate circuits with an arbitrary language, we need one circuit for each possible input length \( n \).
Formal Definitions

• The size of a circuit is the number of gates in it.

• A family of circuits is an infinite sequence
  \( C = (C_0, C_1, \ldots) \) of boolean circuits, where \( C_n \) has \( n \)
  boolean inputs.

• For input \( x \in \{0, 1\}^* \), \( C_{|x|} \) outputs 1 if and only if
  \( x \in L \).

• In other words,

  \[
  C_n \text{ accepts } L \cap \{0, 1\}^n.
  \]
Formal Definitions (concluded)

- \( L \subseteq \{0, 1\}^* \) has **polynomial circuits** if there is a family of circuits \( C \) such that:
  
  - The size of \( C_n \) is at most \( p(n) \) for some fixed polynomial \( p \).
  
  - \( C_n \) accepts \( L \cap \{0, 1\}^n \).
Exponential Circuits Suffice for All Languages

- Theorem 16 (p. 219) implies that there are languages that cannot be solved by circuits of size $2^n/(2n)$.

- But surprisingly, circuits of size $2^{n+2}$ can solve all problems, decidable or otherwise!
Proposition 79  All decision problems (decidable or otherwise) can be solved by a circuit of size $2^{n+2}$ and depth $2n$.

- We will show that for any language $L \subseteq \{0, 1\}^*$, $L \cap \{0, 1\}^n$ can be decided by a circuit of size $2^{n+2}$.

- Define boolean function $f : \{0, 1\}^n \to \{0, 1\}$, where

$$ f(x_1x_2\cdots x_n) = \begin{cases} 1, & x_1x_2\cdots x_n \in L, \\ 0, & x_1x_2\cdots x_n \notin L. \end{cases} $$
The Proof (concluded)

• Clearly, any circuit that implements \( f \) decides \( L \cap \{0, 1\}^n \).

• Now,

\[
  f(x_1x_2\cdots x_n) = (x_1 \land f(1x_2\cdots x_n)) \lor (\lnot x_1 \land f(0x_2\cdots x_n)).
\]

• The circuit size \( s(n) \) for \( f(x_1x_2\cdots x_n) \) hence satisfies

\[
  s(n) = 4 + 2s(n-1)
\]

with \( s(1) = 1 \).

• Solve it to obtain \( s(n) = 5 \times 2^{n-1} - 4 \leq 2^{n+2} \).
The Circuit Complexity of P

**Proposition 80** All languages in $P$ have polynomial circuits.

- Let $L \in P$ be decided by a TM in time $p(n)$.
- By Corollary 35 (p. 328), there is a circuit with $O(p(n)^2)$ gates that accepts $L \cap \{0, 1\}^n$.
- The size of that circuit depends only on $L$ and the length of the input.
- The size of that circuit is polynomial in $n$. 
Polynomial Circuits vs. P

- Is the converse of Proposition 80 true?
  - Do polynomial circuits accept only languages in P?
- No.

- Polynomial circuits can accept *undecidable* languages!\(^a\)

\(^a\)See p. 268 of the textbook.
BPP’s Circuit Complexity: Adleman’s Theorem

Theorem 81 (Adleman, 1978) All languages in BPP have polynomial circuits.

- Our proof will be nonconstructive in that only the existence of the desired circuits is shown.
  - Recall our proof of Theorem 16 (p. 219).
  - Something exists if its probability of existence is nonzero.

- It is not known how to efficiently generate circuit $C_n$.
  - If the construction of $C_n$ can be made efficient, then $P = BPP$, an unlikely result.
The Proof

• Let $L \in BPP$ be decided by a precise polynomial-time NTM $N$ by clear majority.

• We shall prove that $L$ has polynomial circuits $C_0, C_1, \ldots$.
  – These deterministic circuits do not err.

• Suppose $N$ runs in time $p(n)$, where $p(n)$ is a polynomial.

• Let $A_n = \{ a_1, a_2, \ldots, a_m \}$, where $a_i \in \{ 0, 1 \}^{p(n)}$.

• Each $a_i \in A_n$ represents a sequence of nondeterministic choices (i.e., a computation path) for $N$.

• Pick $m = 12(n + 1)$. 
The Proof (continued)

• Let $x$ be an input with $|x| = n$.

• Circuit $C_n$ simulates $N$ on $x$ with all sequences of choices in $A_n$ and then takes the majority of the $m$ outcomes.$^a$
  
  – Note that each $A_n$ yields a circuit.

• As $N$ with $a_i$ is a polynomial-time deterministic TM, it can be simulated by polynomial circuits of size $O(p(n)^2)$.
  
  – See the proof of Proposition 80 (p. 639).

---

$^a$As $m$ is even, there may be no clear majority. Still, the probability of that happening is very small and does not materially affect our general conclusion. Thanks to a lively class discussion on December 14, 2010.
The Circuit

Majority logic

a_1, a_2, a_3, ..., a_m
The Proof (continued)

• The size of $C_n$ is therefore $O(mp(n)^2) = O(np(n)^2)$.
  – This is a polynomial.

• We now confirm the existence of an $A_n$ making $C_n$ correct on all $n$-bit inputs.

• Call $a_i$ bad if it leads $N$ to an error (a false positive or a false negative) for $x$.

• Select $A_n$ uniformly randomly.
The Proof (continued)

• For each $x \in \{0, 1\}^n$, $1/4$ of the computations of $N$ are erroneous.

• Because the sequences in $A_n$ are chosen randomly and independently, the expected number of bad $a_i$’s is $m/4$.\(^a\)

• Also note after fixing the input $x$, the circuit is a function of the random bits.

\(^a\)So the proof will not work for NP. Contributed by Mr. Ching-Hua Yu (D00921025) on December 11, 2012.
The Proof (continued)

• By the Chernoff bound (p. 618), the probability that the number of bad $a_i$’s is $m/2$ or more is at most

$$e^{-m/12} = 2^{-(n+1)}.$$ 

• The error probability of using the majority rule is thus

$$\leq 2^{-(n+1)}$$

for each $x \in \{0, 1\}^n$. 

The Proof (continued)

- The probability that there is an $x$ such that $A_n$ results in an incorrect answer is

\[
\leq 2^n 2^{-(n+1)} = 2^{-1}
\]

by the union bound (Boole’s inequality).\(^a\)

- We just showed that at least half of the random $A_n$ are correct.

- So with probability $\geq 0.5$, a random $A_n$ produces a correct $C_n$ for all inputs of length $n$.

  - Of course, verifying this fact may take a long time.

\(^a\)That is, $\text{prob}[A \cup B \cup \cdots] \leq \text{prob}[A] + \text{prob}[B] + \cdots$. 
The Proof (concluded)

• Because this probability exceeds 0, an $A_n$ that makes majority vote work for all inputs of length $n$ exists.

• Hence a correct $C_n$ exists.$^a$

• We have used the probabilistic method$^b$ popularized by Erdős (1947).$^c$

• This result answers the question on p. 549 with a “yes.”

$^a$Quine (1948), “To be is to be the value of a bound variable.”
$^b$A counting argument in the probabilistic language.
$^c$Szele (1943) and Turán (1934) were earlier.
Leonard Adleman\textsuperscript{a} (1945–)

\textsuperscript{a}Turing Award (2002).
Paul Erdős (1913–1996)
Cryptography
Whoever wishes to keep a secret must hide the fact that he possesses one.
— Johann Wolfgang von Goethe (1749–1832)
Cryptography

- **Alice** (A) wants to send a message to **Bob** (B) over a channel monitored by **Eve** (eavesdropper).

- The protocol should be such that the message is known only to Alice and Bob.

- The art and science of keeping messages secure is **cryptography**.

![Diagram]

Alice → Eve → Bob
Encryption and Decryption

• Alice and Bob agree on two algorithms $E$ and $D$—the encryption and the decryption algorithms.
• Both $E$ and $D$ are known to the public in the analysis.
• Alice runs $E$ and wants to send a message $x$ to Bob.
• Bob operates $D$. 
Encryption and Decryption (concluded)

- Privacy is assured in terms of two numbers $e, d$, the encryption and decryption keys.
- Alice sends $y = E(e, x)$ to Bob, who then performs $D(d, y) = x$ to recover $x$.
- $x$ is called plaintext, and $y$ is called ciphertext.$^a$

\[^a\text{Both “zero” and “cipher” come from the same Arab word.}\]
Some Requirements

• $D$ should be an inverse of $E$ given $e$ and $d$.

• $D$ and $E$ must both run in (probabilistic) polynomial time.

• Eve should not be able to recover $x$ from $y$ without knowing $d$.

  – As $D$ is public, $d$ must be kept secret.
  – $e$ may or may not be a secret.
Degree of Security

- **Perfect secrecy**: After a ciphertext is intercepted by the enemy, the a posteriori probabilities of the plaintext that this ciphertext represents are identical to the a priori probabilities of the same plaintext before the interception.
  - The probability that plaintext $P$ occurs is independent of the ciphertext $C$ being observed.
  - So knowing $C$ yields no advantage in recovering $P$. 
Degree of Security (concluded)

- Such systems are said to be *informationally secure*.
- A system is *computationally secure* if breaking it is theoretically possible but computationally infeasible.
Consider a cryptosystem where:
- The space of ciphertext is as large as that of keys.
- Every plaintext has a nonzero probability of being used.

It is perfectly secure if and only if the following hold.
- A key is chosen with uniform distribution.
- For each plaintext $x$ and ciphertext $y$, there exists a unique key $e$ such that $E(e, x) = y$.

---

Shannon (1949).
The One-Time Pad\textsuperscript{a}

1: Alice generates a random string \( r \) as long as \( x \);
2: Alice sends \( r \) to Bob over a secret channel;
3: Alice sends \( x \oplus r \) to Bob over a public channel;
4: Bob receives \( y \);
5: Bob recovers \( x := y \oplus r \);

\textsuperscript{a}Mauborgne & Vernam (1917); Shannon (1949). It was allegedly used for the hotline between Russia and the U.S.
Analysis

• The one-time pad uses $e = d = r$.

• This is said to be a **private-key cryptosystem**.

• Knowing $x$ and knowing $r$ are equivalent.

• Because $r$ is random and private, the one-time pad achieves perfect secrecy.$^a$

• The random bit string must be new for each round of communication.

• But the assumption of a private channel is problematic.

---

$^a$See p. 660.
Chosen-Plaintext Attack

- Suppose Eve can obtain the ciphertexts for any plaintexts of her choice.
- She can ask the encryption algorithm to encrypt an arbitrary plaintext $x$ to obtain ciphertext $y$.
- Then she analyze those pairs to attack the cryptosystem.
- This is called the **chosen-plaintext attack**.
- For the one-time pad, she only has to perform $y \oplus x$ with a single pair $(x, y)$ to recover the private key $r$!
Public-Key Cryptography\textsuperscript{a}

- Suppose only $d$ is private to Bob, whereas $e$ is public knowledge.

- Bob generates the $(e, d)$ pair and publishes $e$.

- Anybody like Alice can send $E(e, x)$ to Bob.

- Knowing $d$, Bob can recover $x$ via

\[ D(d, E(e, x)) = x. \]

\textsuperscript{a}Diffie & Hellman (1976).
Public-Key Cryptography (concluded)

- The assumptions are complexity-theoretic.
  - It is computationally difficult to compute $d$ from $e$.
  - It is computationally difficult to compute $x$ from $y$ without knowing $d$. 
Whitfield Diffie\textsuperscript{a} (1944–)

\textsuperscript{a}Turing Award (2016).
Martin Hellman\textsuperscript{a} (1945–)

\textsuperscript{a}Turing Award (2016).
Complexity Issues

• Given \( y \) and \( x \), it is easy to verify whether \( E(e, x) = y \).

• Hence one can always guess an \( x \) and verify.

• Cracking a public-key cryptosystem is thus in NP.

• A necessary condition for the existence of secure public-key cryptosystems is \( P \neq NP \).

• But more is needed than \( P \neq NP \).

• For instance, it is not sufficient that \( D \) is hard to compute in the worst case.

• It should be hard in “most” or “average” cases.