Randomization vs. Nondeterminism[a]

- What are the differences between randomized algorithms and nondeterministic algorithms?
- Think of a randomized algorithm as a nondeterministic one but with a probability associated with every guess/branch.
- So each computation path of a randomized algorithm has a probability associated with it.

[a]Contributed by Mr. Olivier Valery (`D01922033`) and Mr. Hasan Alhasan (`D01922034`) on November 27, 2012.

Monte Carlo Algorithms[a]

- The randomized bipartite perfect matching algorithm is called a **Monte Carlo algorithm** in the sense that
  - If the algorithm finds that a matching exists, it is always correct (no **false positives**; no **type 1 errors**).
  - If the algorithm answers in the negative, then it may make an error (**false negatives**; **type 2 errors**).

[a]Metropolis & Ulam (1949).

Monte Carlo Algorithms (continued)

- The algorithm makes a false negative with probability ≤ 0.5.[a]
- Again, this probability refers to[b]

$$\text{prob}[\,\text{algorithm answers "no"} \mid G \text{ has a perfect matching}\,]$$

not

$$\text{prob}[\,G \text{ has a perfect matching} \mid \text{algorithm answers "no"}\,].$$

---

[a] Equivalently, among the coin flip sequences, at most half of them lead to the wrong answer.

[b] In general, prob[algorithm answers "no" | input is a yes instance].

Monte Carlo Algorithms (concluded)

- This probability 0.5 is *not* over the space of all graphs or determinants, but *over* the algorithm's own coin flips.
  - It holds for *any* bipartite graph.
- In contrast, to calculate

  $$\text{prob}[\, G \text{ has a perfect matching} \mid \text{algorithm answers "no"}\,],$$

  we will need the distribution of G.
- But it is an empirical statement that is very hard to verify.

The Markov Inequality[a]

Lemma 67 *Let x be a random variable taking nonnegative integer values. Then for any $k > 0$,*

$$\text{prob}[\, x \geq kE[\, x\,]\,] \leq 1/k.$$

- Let p_i denote the probability that $x = i$.

$$\begin{aligned} E[\, x\,] &= \sum_i i p_i = \sum_{i < kE[\, x\,]} i p_i + \sum_{i \geq kE[\, x\,]} i p_i \\ &\geq \sum_{i \geq kE[\, x\,]} i p_i \geq kE[\, x\,] \sum_{i \geq kE[\, x\,]} p_i \\ &\geq kE[\, x\,] \times \text{prob}[x \geq kE[\, x\,]]. \end{aligned}$$

[a]Andrei Andreyevich Markov (1856–1922).

Andrei Andreyevich Markov (1856–1922)

FSAT for k-SAT Formulas (p. 500)

- Let $\phi(x_1, x_2, \ldots, x_n)$ be a k-SAT formula.
- If ϕ is satisfiable, then return a satisfying truth assignment.
- Otherwise, return "no."
- We next propose a randomized algorithm for this problem.

A Random Walk Algorithm for ϕ in CNF Form

```
Start with an arbitrary truth assignment T;
for i = 1, 2, ..., r do
    if T ⊨ φ then
        return "φ is satisfiable with T";
    else
        Let c be an unsatisfied clause in φ under T; {All of
        its literals are false under T.}
        Pick any x of these literals at random;
        Modify T to make x true;
    end if
end for
return "φ is unsatisfiable";
```

3SAT vs. 2SAT Again

- Note that if ϕ is unsatisfiable, the algorithm will answer "unsatisfiable."
- The random walk algorithm needs expected exponential time for 3SAT.
  - In fact, it runs in expected $O((1.333 \cdots + \epsilon)^n)$ time with $r = 3n$,[a] much better than $O(2^n)$.[b]
- We will show immediately that it works well for 2SAT.
- The state of the art as of 2014 is expected $O(1.30704^n)$ time for 3SAT and expected $O(1.46899^n)$ time for 4SAT.[c]

---

[a]Use this setting per run of the algorithm.

[b]Schöning (1999). Makino, Tamaki, & Yamamoto (2011) improve the bound to deterministic $O(1.3303^n)$.

[c]Hertli (2014).

Random Walk Works for 2SAT[a]

Theorem 68 *Suppose the random walk algorithm with $r = 2n^2$ is applied to any satisfiable 2SAT problem with n variables. Then a satisfying truth assignment will be discovered with probability at least 0.5.*

- Let $\hat{T}$ be a truth assignment such that $\hat{T} \models \phi$.
- Assume our starting T differs from $\hat{T}$ in i values.
  - Their Hamming distance is i.
  - Recall T is arbitrary.

---

[a] Papadimitriou (1991).

The Proof

- Let $t(i)$ denote the expected number of repetitions of the flipping step[a] until a satisfying truth assignment is found.
- It can be shown that $t(i)$ is finite.
- $t(0) = 0$ because it means that $T = \hat{T}$ and hence $T \models \phi$.
- If $T \neq \hat{T}$ or any other satisfying truth assignment, then we need to flip the coin at least once.
- We flip a coin to pick among the 2 literals of a clause not satisfied by the present T.
- At least one of the 2 literals is true under $\hat{T}$ because $\hat{T}$ satisfies all clauses.

[a] That is, Statement 7.

The Proof (continued)

- So we have at least a 50% chance of moving closer to $\hat{T}$.
- Thus
$$t(i) \leq \frac{t(i-1)+t(i+1)}{2}+1$$
for $0 < i < n$.
  - Inequality is used because, for example, T may differ from $\hat{T}$ in both literals.
- It must also hold that
$$t(n) \leq t(n-1)+1$$
because at $i = n$, we can only decrease i.

The Proof (continued)

- Now, put the necessary relations together:

$$t(0) = 0, \quad (10)$$

$$t(i) \leq \frac{t(i-1)+t(i+1)}{2} + 1, \quad 0 < i < n, \quad (11)$$

$$t(n) \leq t(n-1) + 1. \quad (12)$$

- Technically, this is a one-dimensional random walk with an absorbing barrier at $i = 0$ and a reflecting barrier at $i = n$ (if we replace "$\leq$" with "$=$").[a]

[a]The proof in the textbook does exactly that. But a student pointed out difficulties with this proof technique on December 8, 2004. So our proof here uses the original inequalities.

The Proof (continued)

- Add up the relations for $2t(1), 2t(2), 2t(3), \ldots, 2t(n-1), t(n)$ to obtain[a]

$$\begin{aligned} & 2t(1) + 2t(2) + \cdots + 2t(n-1) + t(n) \\ \leq \quad & t(0) + t(1) + 2t(2) + \cdots + 2t(n-2) + 2t(n-1) + t(n) \\ & +2(n-1) + 1. \end{aligned}$$

- Simplify it to yield

$$t(1) \leq 2n - 1. \tag{13}$$

[a]Adding up the relations for $t(1), t(2), t(3), \ldots, t(n-1)$ will also work, thanks to Mr. Yen-Wu Ti (D91922010).

The Proof (continued)

- Add up the relations for $2t(2), 2t(3), \ldots, 2t(n-1), t(n)$ to obtain

$$\begin{aligned} & 2t(2) + \cdots + 2t(n-1) + t(n) \\ \leq\ & t(1) + t(2) + 2t(3) + \cdots + 2t(n-2) + 2t(n-1) + t(n) \\ & +2(n-2) + 1. \end{aligned}$$

- Simplify it to yield

$$t(2) \leq t(1) + 2n - 3 \leq 2n - 1 + 2n - 3 = 4n - 4$$

by Eq. (13) on p. 544.

The Proof (continued)

- Continuing the process, we shall obtain

$$t(i) \leq 2in - i^2.$$

- The worst upper bound happens when $i = n$, in which case

$$t(n) \leq n^2.$$

- We conclude that

$$t(i) \leq t(n) \leq n^2$$

  for $0 \leq i \leq n$.

The Proof (concluded)

- So the expected number of steps is at most n^2.
- The algorithm picks $r = 2n^2$.
- Apply the Markov inequality (p. 535) with $k = 2$ to yield the desired probability of 0.5.
- The proof does *not* yield a polynomial bound for 3SAT.[a]

[a] Contributed by Mr. Cheng-Yu Lee (`R95922035`) on November 8, 2006.

Boosting the Performance

- We can pick $r = 2mn^2$ to have an error probability of

$$\leq \frac{1}{2m}$$

by Markov's inequality.

- Alternatively, with the same running time, we can run the "$r = 2n^2$" algorithm m times.

- The error probability is now reduced to

$$\leq 2^{-m}.$$

Primality Tests

- PRIMES asks if a number N is a prime.
- The classic algorithm tests if $k \mid N$ for $k = 2, 3, \ldots, \sqrt{N}$.
- But it runs in $\Omega(2^{(\log_2 N)/2})$ steps.

The Fermat Test for Primality

Fermat's "little" theorem (p. 486) suggests the following primality test for any given number N:

```
Pick a number a randomly from { 1, 2, ..., N − 1 };
if a^(N−1) ≢ 1 mod N then
    return "N is composite";
else
    return "N is (probably) a prime";
end if
```

The Fermat Test for Primality (concluded)

- **Carmichael numbers** are composite numbers that will pass the Fermat test for *all* $a \in \{ 1, 2, \ldots, N-1 \}$.[a]
  - The Fermat test will return "N is a prime" for all Carmichael numbers N.
- Unfortunately, there are infinitely many Carmichael numbers.[b]
- In fact, the number of Carmichael numbers less than N exceeds $N^{2/7}$ for N large enough.
- So the Fermat test is an incorrect algorithm for PRIMES.

---

[a] Carmichael (1910). Lo (1994) mentions an investment strategy based on such numbers!

[b] Alford, Granville, & Pomerance (1992).

Square Roots Modulo a Prime

- Equation $x^2 \equiv a \bmod p$ has at most two (distinct) roots by Lemma 64 (p. 491).
  - The roots are called **square roots**.
  - Numbers a with square roots *and* $\gcd(a, p) = 1$ are called **quadratic residues**.
    - They are

$$1^2 \bmod p, 2^2 \bmod p, \ldots, (p-1)^2 \bmod p.$$

- We shall show that a number either has two roots or has none, and testing which is the case is trivial.[a]

[a] But no efficient *deterministic* general-purpose square-root-extracting algorithms are known yet.

Euler's Test

Lemma 69 (Euler) *Let p be an odd prime and $a \neq 0 \bmod p$.*

1. *If*
$$a^{(p-1)/2} \equiv 1 \bmod p,$$
then $x^2 \equiv a \bmod p$ has two roots.

2. *If*
$$a^{(p-1)/2} \not\equiv 1 \bmod p,$$
then
$$a^{(p-1)/2} \equiv -1 \bmod p$$
and $x^2 \equiv a \bmod p$ has no roots.

The Proof (continued)

- Let r be a primitive root of p.
- Fermat's "little" theorem says $r^{p-1} \equiv 1 \bmod p$, so

$$r^{(p-1)/2}$$

  is a square root of 1.
- In particular,

$$r^{(p-1)/2} \equiv 1 \text{ or } -1 \bmod p.$$

- But as r is a primitive root, $r^{(p-1)/2} \not\equiv 1 \bmod p$.
- Hence $r^{(p-1)/2} \equiv -1 \bmod p$.

The Proof (continued)

- Let $a = r^k \bmod p$ for some k.
- Suppose $a^{(p-1)/2} \equiv 1 \bmod p$.
- Then

$$1 \equiv a^{(p-1)/2} \equiv r^{k(p-1)/2} \equiv \left[r^{(p-1)/2} \right]^k \equiv (-1)^k \bmod p.$$

- So k must be even.

The Proof (continued)

- Suppose $a = r^{2j} \bmod p$ for some $1 \leq j \leq (p-1)/2$.
- Then
$$a^{(p-1)/2} \equiv r^{j(p-1)} \equiv 1 \bmod p.$$
- The two *distinct* roots of a are
$$r^j, -r^j (\equiv r^{j+(p-1)/2} \bmod p).$$
  - If $r^j \equiv -r^j \bmod p$, then $2r^j \equiv 0 \bmod p$, which implies $r^j \equiv 0 \bmod p$, a contradiction as r is a primitive root.

The Proof (continued)

- As $1 \leq j \leq (p-1)/2$, there are $(p-1)/2$ such a's.
- Each such $a \equiv r^{2j} \bmod p$ has 2 distinct square roots.
- The square roots of all these a's are distinct.
  - The square roots of *different* a's must be different.
- Hence the set of *square roots* is $\{1, 2, \ldots, p-1\}$.
- As a result,

$$a = r^{2j} \bmod p, 1 \leq j \leq (p-1)/2,$$

exhaust all the quadratic residues.

The Proof (concluded)

- Suppose $a = r^{2j+1} \bmod p$ now.
- Then it has no square roots because all the square roots have been taken.
- Finally,

$$a^{(p-1)/2} \equiv \left[r^{(p-1)/2} \right]^{2j+1} \equiv (-1)^{2j+1} \equiv -1 \bmod p.$$

The Legendre Symbol[a] and Quadratic Residuacity Test

- By Lemma 69 (p. 553),

$$a^{(p-1)/2} \bmod p = \pm 1$$

for $a \not\equiv 0 \bmod p$.

- For odd prime p, define the **Legendre symbol** $(a \,|\, p)$ as

$$(a \,|\, p) \triangleq \begin{cases} 0, & \text{if } p \,|\, a, \\ 1, & \text{if } a \text{ is a quadratic residue modulo } p, \\ -1, & \text{if } a \text{ is a } \textbf{quadratic nonresidue} \text{ modulo } p. \end{cases}$$

- It is sometimes pronounced "a over p."

[a] Andrien-Marie Legendre (1752–1833).

The Legendre Symbol and Quadratic Residuacity Test (concluded)

- Euler's test (p. 553) implies

$$a^{(p-1)/2} \equiv (a \,|\, p) \bmod p$$

for any odd prime p and any integer a.

- Note that $(ab \,|\, p) = (a \,|\, p)(b \,|\, p)$.

Gauss's Lemma

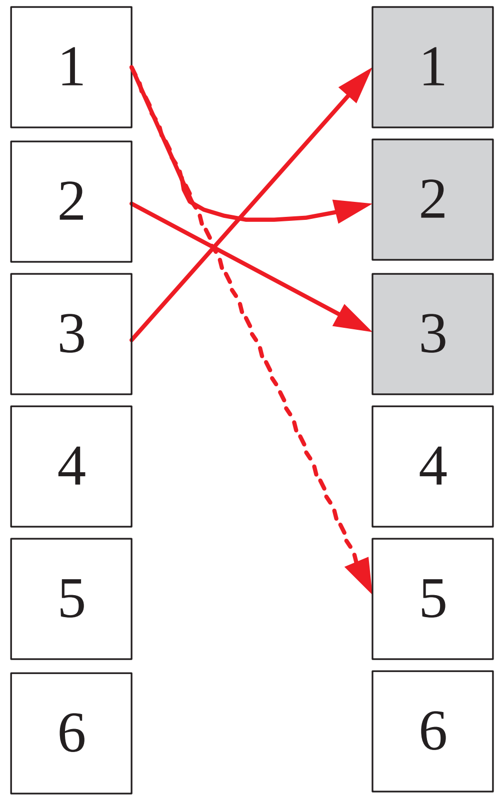
Lemma 70 (Gauss) *Let p and q be two distinct odd primes. Then $(q \,|\, p) = (-1)^m$, where m is the number of residues in $R \stackrel{\Delta}{=} \{\, iq \bmod p : 1 \leq i \leq (p-1)/2 \,\}$ that are greater than $(p-1)/2$.*

- All residues in R are distinct.
  - If $iq = jq \bmod p$, then $p \,|\, (j-i)$ or $p \,|\, q$.
  - But neither is possible.
- No two elements of R add up to p.
  - If $iq + jq \equiv 0 \bmod p$, then $p \,|\, (i+j)$ or $p \,|\, q$.
  - But neither is possible.

The Proof (continued)

- Replace each of the m elements $a \in R$ such that $a > (p-1)/2$ by $p - a$.
  - This is equivalent to performing $-a \bmod p$.
- Call the resulting set of residues R'.
- All numbers in R' are at most $(p-1)/2$.
- In fact, $R' = \{ 1, 2, \ldots, (p-1)/2 \}$ (see illustration next page).
  - Otherwise, two elements of R would add up to p,[a] which has been shown to be impossible.

[a] Because then $iq \equiv -jq \bmod p$ for some $i \neq j$.



$p = 7$ and $q = 5$.

The Proof (concluded)

- Alternatively, $R' = \{ \pm iq \bmod p : 1 \leq i \leq (p-1)/2 \}$, where exactly m of the elements have the minus sign.
- Take the product of all elements in the two representations of R'.
- So

$$[(p-1)/2]! \equiv (-1)^m q^{(p-1)/2} [(p-1)/2]! \bmod p.$$

- Because $\gcd([(p-1)/2]!, p) = 1$, the above implies

$$1 = (-1)^m q^{(p-1)/2} \bmod p.$$

Legendre's Law of Quadratic Reciprocity[a]

- Let p and q be two distinct odd primes.
- The next result says $(p \,|\, q)$ and $(q \,|\, p)$ are distinct if and only if both p and q are 3 mod 4.

Lemma 71 (Legendre, 1785; Gauss)

$$(p \,|\, q)(q \,|\, p) = (-1)^{\frac{p-1}{2} \frac{q-1}{2}}.$$

---

[a]First stated by Euler in 1751. Legendre (1785) did not give a correct proof. Gauss proved the theorem when he was 19. He gave at least 8 different proofs during his life. The 152nd proof appeared in 1963. A computer-generated formal proof was given in Russinoff (1990). As of 2008, there had been 4 such proofs. Wiedijk (2008), "the Law of Quadratic Reciprocity is the first nontrivial theorem that a student encounters in the mathematics curriculum."

The Proof (continued)

- Sum the elements of R' in the previous proof in mod2.
- On one hand, this is just $\sum_{i=1}^{(p-1)/2} i \bmod 2$.
- On the other hand, the sum equals

$$
\begin{aligned}
& mp + \sum_{i=1}^{(p-1)/2} \left(iq - p \left\lfloor \frac{iq}{p} \right\rfloor \right) \bmod 2 \\
= \quad & mp + \left(q \sum_{i=1}^{(p-1)/2} i - p \sum_{i=1}^{(p-1)/2} \left\lfloor \frac{iq}{p} \right\rfloor \right) \bmod 2.
\end{aligned}
$$

  - m of the $iq \bmod p$ are replaced by $p - iq \bmod p$.
  - But signs are irrelevant under mod2.
  - m is as in Lemma 70 (p. 561).

The Proof (continued)

- Ignore odd multipliers to make the sum equal

$$m + \left(\sum_{i=1}^{(p-1)/2} i - \sum_{i=1}^{(p-1)/2} \left\lfloor \frac{iq}{p} \right\rfloor \right) \bmod 2.$$

- Equate the above with $\sum_{i=1}^{(p-1)/2} i$ modulo 2.
- Now simplify to obtain

$$m \equiv \sum_{i=1}^{(p-1)/2} \left\lfloor \frac{iq}{p} \right\rfloor \bmod 2.$$

The Proof (continued)

- $\sum_{i=1}^{(p-1)/2} \lfloor \frac{iq}{p} \rfloor$ is the number of integral points *below* the line
$$y = (q/p)\, x$$
for $1 \leq x \leq (p-1)/2$.
- Gauss's lemma (p. 561) says $(q \,|\, p) = (-1)^m$.
- Repeat the proof with p and q reversed.
- Then $(p \,|\, q) = (-1)^{m'}$, where m' is the number of integral points *above* the line $y = (q/p)\, x$ for $1 \leq y \leq (q-1)/2$.

The Proof (concluded)

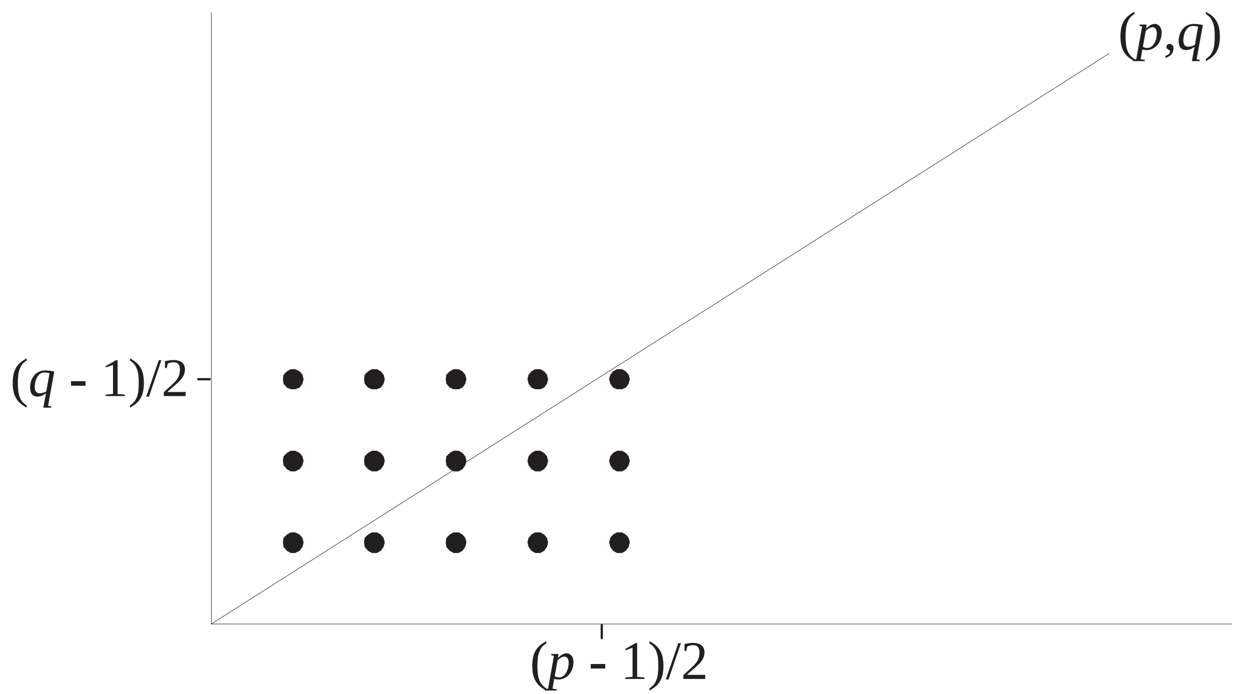
- As a result,

$$(p\,|\,q)(q\,|\,p) = (-1)^{m+m'}.$$

- But $m + m'$ is the total number of integral points in the $[1, \frac{p-1}{2}] \times [1, \frac{q-1}{2}]$ rectangle, which is

$$\frac{p-1}{2}\,\frac{q-1}{2}.$$

Eisenstein's Rectangle



Above, $p = 11$, $q = 7$, $m = 7$, $m' = 8$.

The Jacobi Symbol[a]

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

[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

$$(a \,|\, m) \stackrel{\Delta}{=} \prod_{i=1}^{k} (a \,|\, p_i).$$

  - Note that the Jacobi symbol equals ± 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. $(ab \,|\, m) = (a \,|\, m)(b \,|\, m)$.
2. $(a \,|\, m_1 m_2) = (a \,|\, m_1)(a \,|\, m_2)$.
3. If $a \equiv b \bmod m$, then $(a \,|\, m) = (b \,|\, m)$.
4. $(-1 \,|\, m) = (-1)^{(m-1)/2}$ (by Lemma 70 on p. 561).
5. $(2 \,|\, m) = (-1)^{(m^2-1)/8}$.[a]
6. If a and m are both odd, then $(a \,|\, m)(m \,|\, a) = (-1)^{(a-1)(m-1)/4}$.

---

[a] By Lemma 70 (p. 561) and some parity arguments.

Properties of the Jacobi Symbol (concluded)

- Properties 3–6 allow us to calculate the Jacobi symbol *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. 553.

[b] Contributed by Mr. Huang, Kuan-Lin (B96902079, R00922018) on December 6, 2011.

Calculation of $(2200 \mid 999)$

$$
\begin{aligned}
(2200 \mid 999) &= (202 \mid 999) \\
&= (2 \mid 999)(101 \mid 999) \\
&= (-1)^{(999^2-1)/8}(101 \mid 999) \\
&= (-1)^{124750}(101 \mid 999) = (101 \mid 999) \\
&= (-1)^{(100)(998)/4}(999 \mid 101) = (-1)^{24950}(999 \mid 101) \\
&= (999 \mid 101) = (90 \mid 101) = (-1)^{(101^2-1)/8}(45 \mid 101) \\
&= (-1)^{1275}(45 \mid 101) = -(45 \mid 101) \\
&= -(-1)^{(44)(100)/4}(101 \mid 45) = -(101 \mid 45) = -(11 \mid 45) \\
&= -(-1)^{(10)(44)/4}(45 \mid 11) = -(45 \mid 11) \\
&= -(1 \mid 11) = -1.
\end{aligned}
$$

A Result Generalizing Proposition 10.3 in the Textbook

Theorem 72 *The group of set* $\Phi(n)$ *under multiplication* $\bmod n$ *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.

The Jacobi Symbol and Primality Test[a]

Lemma 73 *If* $(M \mid N) \equiv M^{(N-1)/2} \bmod N$ *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

$$\begin{aligned} M &= r \bmod p, \\ M &= 1 \bmod m. \end{aligned}$$

[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,

$$M^{(N-1)/2} = (M \mid N) = (M \mid p)(M \mid m) = -1 \bmod N.$$

- Hence

$$M^{(N-1)/2} = -1 \bmod m.$$

- But because $M = 1 \bmod m$,

$$M^{(N-1)/2} = 1 \bmod m,$$

  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. 576), there exists a primitive root r modulo p^a.
- From the assumption,

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

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

The Proof (continued)

- As $r \in \Phi(N)$ (prove it), we have
$$r^{N-1} = 1 \bmod N.$$
- As r's exponent modulo $N = p^a$ is $\phi(N) = p^{a-1}(p-1)$,
$$p^{a-1}(p-1) \,|\, (N-1),$$
which implies that $p \,|\, (N-1)$.
- But this is impossible given that $p \,|\, N$.

The Proof (continued)

- 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. 576), there exists a primitive root r modulo p^a.
- From the assumption,

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

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

The Proof (continued)

- In particular,

$$M^{N-1} = 1 \bmod p^a \tag{14}$$

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

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

$$\begin{aligned} M &= r \bmod p^a, \\ M &= 1 \bmod m. \end{aligned}$$

- Because $M = r \bmod p^a$ and Eq. (14),

$$r^{N-1} = 1 \bmod p^a.$$

The Proof (concluded)

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

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

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

- But this is impossible given that $p \,|\, N$.

The Number of Witnesses to Compositeness

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

- By Lemma 73 (p. 577) there is at least one $a \in \Phi(N)$ such that $(a \mid N) \not\equiv a^{(N-1)/2} \bmod N$.
- Let $B \stackrel{\Delta}{=} \{ b_1, b_2, \ldots, b_k \} \subseteq \Phi(N)$ be the set of *all* distinct residues such that $(b_i \mid N) \equiv b_i^{(N-1)/2} \bmod N$.
- Let $aB \stackrel{\Delta}{=} \{ ab_i \bmod N : i = 1, 2, \ldots, k \}$.
- Clearly, $aB \subseteq \Phi(N)$, too.

The Proof (concluded)

- $|\, aB \,| = k$.
  - $ab_i \equiv ab_j \bmod N$ implies $N \mid a(b_i - b_j)$, which is impossible because $\gcd(a, N) = 1$ and $N > |\, b_i - b_j \,|$.
- $aB \cap B = \emptyset$ because

$$(ab_i)^{(N-1)/2} \equiv a^{(N-1)/2} b_i^{(N-1)/2} \not\equiv (a \mid N)(b_i \mid N) \equiv (ab_i \mid N).$$

- Combining the above two results, we know

$$\frac{|\, B \,|}{\phi(N)} \leq \frac{|\, B \,|}{|\, B \cup aB \,|} = 0.5.$$

if N is even but $N \neq 2$ **then**
    return "N is composite";
else if $N = 2$ **then**
    return "N is a prime";
end if
Pick $M \in \{ 2, 3, \ldots, N - 1 \}$ randomly;
if $\gcd(M, N) > 1$ **then**
    return "N is composite";
else
    if $(M \mid N) \equiv M^{(N-1)/2} \bmod N$ **then**
        return "N is (probably) a prime";
    else
        return "N is composite";
    end if
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)

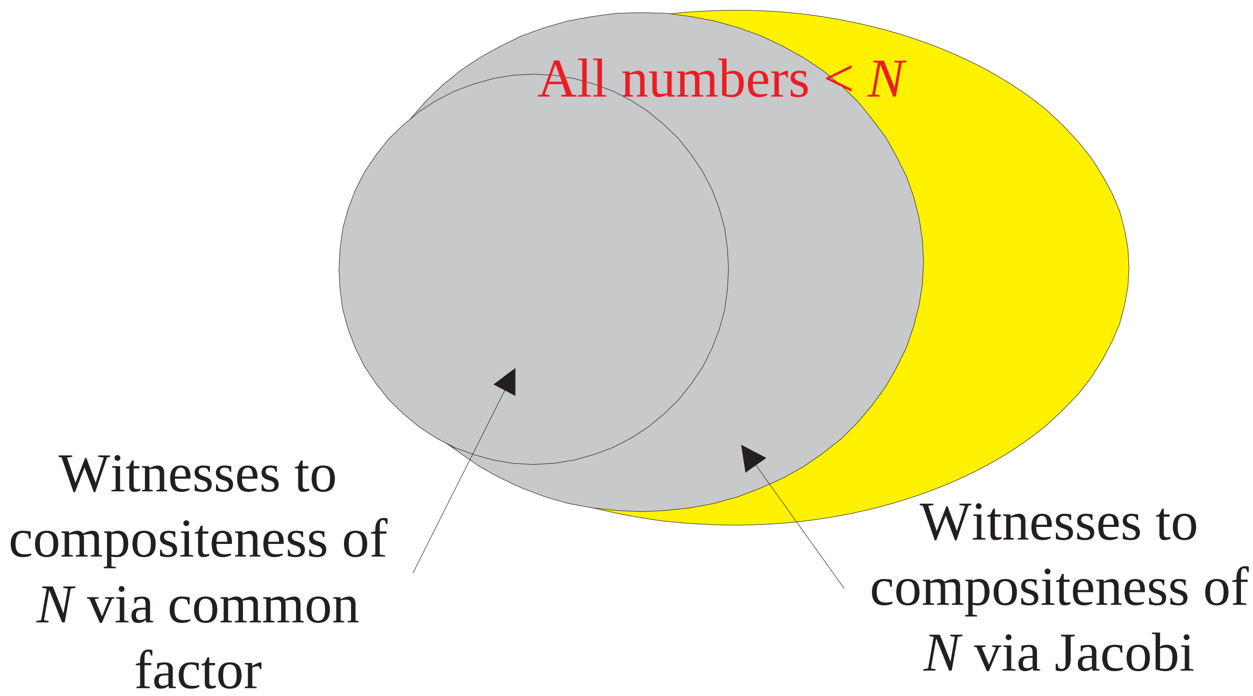
- The probability of a false negative (again, for COMPOSITENESS) is at most one half.
  - Suppose the input is composite.
  - By Theorem 74 (p. 584),

$$\text{prob}[\,\text{algorithm answers "no"} \mid N \text{ is composite}\,] \leq 0.5.$$

  - 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]

[a] Not PRIMES.

The Improved Density Attack for COMPOSITENESS



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 **RP** (**randomized polynomial time**).[a]

[a] Adleman & Manders (1977).

Comments on RP

- In analogy to Proposition 41 (p. 331), 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 ≤ 0.5.
- But *any* constant ϵ between 0 and 1 can replace 0.5.
  - Repeat the algorithm $k \stackrel{\Delta}{=} \lceil -\frac{1}{\log_2 \epsilon} \rceil$ times and answer "no" only if all the runs answer "no."
  - The probability of false negatives becomes $\epsilon^k \leq 0.5$.

Where RP Fits

- $P \subseteq RP \subseteq NP$.
  - 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.
- COMPOSITENESS $\in$ RP;[a] PRIMES $\in$ coRP; PRIMES $\in$ RP.[b]
  - In fact, PRIMES $\in$ P.[c]
- $RP \cup coRP$ is an alternative "plausible" notion of efficient computation.

---

[a] Rabin (1976); Solovay & Strassen (1977).
[b] Adleman & Huang (1987).
[c] Agrawal, Kayal, & Saxena (2002).

ZPP[a] (Zero Probabilistic Polynomial)

- The class **ZPP** is defined as RP $\cap$ coRP.
- A language in ZPP has *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, *eventually* one definite answer will come (unlike RP).
  - A *positive* answer from the one without false positives.
  - A *negative* answer from the one without false negatives.

[a]Gill (1977).

The ZPP Algorithm (**Las Vegas**)

{Suppose $L \in \text{ZPP}$.}
{N_1 has no false positives, and N_2 has no false negatives.}
while `true` **do**
    if $N_1(x) =$ "yes" **then**
        return "yes";
    end if
    if $N_2(x) =$ "no" **then**
        return "no";
    end if
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

$$\sum_{i=1}^{\infty} 0.5^i i p(n) = 2p(n).$$

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