Chapter 7
Randomization

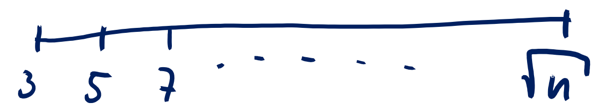
Algorithm Theory
WS 2019/20

Philipp Bamberger

Primality Testing

Problem: Given a natural number $n \geq 2$, is n a prime number?

Simple primality test:



if n is even **then**
    return $(n = 2)$
for $i := 1$ **to** $\lfloor \sqrt{n}/2 \rfloor$ **do**
    if $2i + 1$ divides n **then**
        return false
return true

n is not prime $\Leftrightarrow$ $\underbrace{n = a \cdot b}_{\Rightarrow\, a \leq \sqrt{n} \text{ or } b \leq \sqrt{n}}$, $a, b < n$

- **Running time:** $O(\sqrt{n})$

size of input: $O(\log n)$

A Better Algorithm?

- How can we test primality efficiently?
- We need a little bit of basic number theory…

Square Roots of Unity: If p is a prime, the only solutions of the equation $x^2 \equiv 1 \pmod{p}$ are $x \equiv \pm 1 \pmod{p}$

$x^2 \equiv 1 \pmod{p}$

$\Leftrightarrow x^2 - 1 \equiv 0 \pmod{p}$

$\Leftrightarrow (x+1)(x-1) \equiv 0 \pmod{p}$

$\Leftrightarrow (x+1)(x-1) = c \cdot p$

$\Rightarrow p$ is a prime factor of $x+1$ or $x-1$

$\Rightarrow x+1 \equiv 0 \pmod{p}$ or $x-1 \equiv 0 \pmod{p}$

$\Rightarrow x \equiv -1 \pmod{p}$ or $x \equiv 1 \pmod{p}$

not true if p is not prime

$p = 15$

$x = 4 \qquad x^2 \equiv 1 \pmod{15}$

$x \in \{1, \dots, n-1\} \qquad -1 \equiv n-1 \pmod{n}$

- If we find an $x \not\equiv \pm 1 \pmod{n}$ such that $x^2 \equiv 1 \pmod{n}$, we can conclude that n is not a prime. and vice versa

Algorithm Idea

$p-1 = \underbrace{2 \cdot \ldots \cdot 2}_{2^s} \cdot \underbrace{3 \cdot 5 \cdot \ldots}_{d}$

Claim: Let $p > 2$ be a prime number such that $p - 1 = 2^s d$ for an integer $s \geq 1$ and some odd integer d. Then for all $a \in \{1, \ldots, p-1\}$

$$a^d \equiv 1 \pmod{p} \;\; \textbf{or} \;\; a^{2^r d} \equiv -1 \pmod{p} \;\; \text{for some } 0 \leq r < s.$$

Proof:

- **Fermat's Little Theorem:** Given a prime number p,

$$\forall a \in \{1, \ldots, p-1\}: \quad a^{p-1} \equiv 1 \pmod{p}$$

Let $a \in \{1, \ldots, p-1\} \overset{\text{Fermat}}{\Longrightarrow} \left(a^{\frac{p-1}{2}}\right)^2 \equiv 1 \pmod{p} \Rightarrow a^{\frac{p-1}{2}} \equiv \pm 1 \pmod{p}$

$$a^{\frac{p-1}{2}} \equiv \begin{cases} -1 \pmod{p} \Rightarrow a^{2^{s-1}d} \equiv -1 \pmod{p} \; \checkmark \\ 1 \pmod{p} \Rightarrow \begin{cases} \text{if } s = 1, \text{ then } a^{\frac{p-1}{2}} = a^d \equiv 1 \pmod{p} \; \checkmark \\ \text{if } s \geq 2: \frac{p-1}{2} \text{ is even} \Rightarrow a^{\frac{p-1}{2}} = \left(a^{\frac{p-1}{4}}\right)^2 \equiv 1 \pmod{p} \end{cases} \end{cases}$$

$$\Rightarrow a^{\frac{p-1}{4}} \equiv \begin{cases} -1 \pmod{p} \; \checkmark \\ 1 \pmod{p} \end{cases} \quad \begin{cases} s = 2 \\ s > 2 \to a^{\frac{p-1}{8}} \ldots \end{cases}$$

Primality Test

We have: If n is an odd prime and $n - 1 = 2^s d$ for an integer $s \geq 1$ and an odd integer d. Then for all $a \in \{1, \dots, n-1\}$,

$$\varphi(a) = \quad a^d \equiv 1 \pmod{n} \;\; \textbf{or} \;\; a^{2^r d} \equiv -1 \pmod{n} \;\; \text{for some } 0 \leq r < s.$$

Idea: If we find an $a \in \{1, \dots, n-1\}$ such that

$$\neg\varphi(a) = \quad a^d \not\equiv 1 \pmod{n} \;\; \textbf{and} \;\; a^{2^r d} \not\equiv -1 \pmod{n} \;\; \text{for all } 0 \leq r < s,$$

we can conclude that n is not a prime.

- For every odd composite $n > 2$, at least $^3/_4$ of all possible a satisfy $\underbrace{\text{the above condition}}_{\varphi(a)}$ $\{1, \dots, n-1\}$
- How can we find such a *witness* a efficiently?

Miller-Rabin Primality Test

- Given a natural number $n \geq 2$, is n a prime number?

$n-1 \equiv -1 \pmod{n}$

Miller-Rabin Test:

1. **if** n is even **then return** $(n = 2)$
2. compute s, d such that $n - 1 = 2^s d$;
3. choose $a \in \{2, \dots, n-2\}$ uniformly at random;
4. $x \coloneqq a^d \bmod n$;
5. **if** $x = 1$ **or** $x = n - 1$ **then return probably prime;**
   $P(a)$ holds
6. **for** $r \coloneqq 1$ **to** $s - 1$ **do**
7. $\quad x \coloneqq x^2 \bmod n$;
8. $\quad$ **if** $x = n - 1$ **then return probably prime;**
   "true"
9. **return composite;**
   $\neg P(a)$ "false"

check if $P(a)$ is true

Analysis

Theorem:

- If n is prime, the Miller-Rabin test always returns **true**.
- If n is composite, the Miller-Rabin test returns **false** with probability at least $^3/_4$.

Corollary: If the Miller-Rabin test is repeated k times, it fails to detect a composite number n with probability at most 4^{-k}.

Running Time

Cost of Modular Arithmetic:

- Representation of a number $x \in \mathbb{Z}_n$: $O(\log n)$ bits
- Cost of adding two numbers $x + y \bmod n$: $O(\log n)$
- Cost of multiplying two numbers $x \cdot y \bmod n$: naively $O(\log^2 n)$
  - It's like multiplying degree $O(\log n)$ polynomials
    → use FFT to compute $z = x \cdot y$

$$O(\log n \cdot \log\log n \cdot \log\log\log n) = \tilde{O}(\log n)$$

$$\tilde{O}(f(n)) = O(f(n) \cdot \log^c(f(n)))$$

Running Time

$x, d \in \{1, \dots, n-1\}$

Cost of exponentiation $x^d \bmod n$: naively: $O(d)$ multiplications

- Can be done using $O(\log d)$ multiplications

  $\Rightarrow$ on total $O(\log^2 n \cdot \log\log n \cdot \log\log\log n)$

- Base-2 representation of d: $d = \sum_{i=0}^{\lfloor \log d \rfloor} d_i 2^i$ $= O(\log^2 n)$

- **Fast exponentiation:**
  1. $y := 1$;
  2. **for** $i := \lfloor \log d \rfloor$ **to** 0 **do**
  3. $\quad y := y^2 \bmod n$;
  4. $\quad$**if** $d_i = 1$ **then** $y := y \cdot x \bmod n$;
  5. **return** y;

- **Example:** $d = 22 = 10110_2$

$$x^{22} = (x^{11})^2 = (x^{10} \cdot x)^2 = \left((x^5)^2 \cdot x\right)^2 = \left(\left((x^2)^2 \cdot x\right)^2 \cdot x\right)^2$$

$x^5 = (x^2)^2 \cdot x$

Running Time

Theorem: One iteration of the Miller-Rabin test can be implemented with running time $O(\log^2 n \cdot \log\log n \cdot \log\log\log n)$. $= \tilde{O}(\log^2 n)$

1. **if** n is even **then return** $(n = 2)$
2. compute s, d such that $n - 1 = 2^s d$;
3. choose $a \in \{2, \dots, n-2\}$ uniformly at random;
4. $x \coloneqq a^d \bmod n$;
5. **if** $x = 1$ **or** $x = n - 1$ **then return probably prime;**
6. **for** $r \coloneqq 1$ **to** $s - 1$ **do**
7. $\quad x \coloneqq x^2 \bmod n$;
8. $\quad$ **if** $x = n - 1$ **then return probably prime;**
9. **return composite;**

$O(\log n)$

$s = O(\log n)$ multipl.

Corollary: There is an algorithm with runtime $\tilde{O}(\log^3 n)$ that correctly tests whether n is prime w.h.p.

$1 - \frac{1}{n^c}$ for any $c > 1$

Failure prob. $\leq \frac{1}{n^c}$

Proof: Let $c > 1$. Run Miller-Rabin $\frac{c}{2} \log n$ times.

Return false iff M.-R. returns false in at least one iteration.

If n is prime, M.-R. outputs true in each iteration.

If n is composite, M.-R. fails (i.e., outputs true) with probability

$$4^{-\frac{c}{2}\log n} = \frac{1}{n^c}.$$

Deterministic Primality Test

- If a conjecture called the generalized Riemann hypothesis (GRH) is true, the Miller-Rabin test can be turned into a polynomial-time, deterministic algorithm

  → It is then sufficient to try all $a \in \{1, \dots, O(\log^2 n)\}$

- It has long not been proven whether a deterministic, polynomial-time algorithm exists

- In 2002, Agrawal, Kayal, and Saxena gave an $\tilde{O}(\log^{12} n)$-time deterministic algorithm
  - Has been improved to $\tilde{O}(\log^6 n)$

- In practice, the randomized Miller-Rabin test is still the fastest algorithm