

Lecture notes: Algorithms for integers, polynomials (Thorsten Theobald)

1. Euclid's Algorithm

Euclid's Algorithm for computing the greatest common divisor belongs to the oldest known computing procedures. It was already known to Eudoxus (375 BC), and it is described in the Euclid's "Elements" (300 BC). It is of fundamental importance and occurs in many computational tasks. We write $a|b$ for integers a, b , if a is a divisor of b , i.e., if there exists a $k \in \mathbb{Z}$ with $b = k \cdot a$.

DEFINITION 1.1. For two nonnegative integers a, b let

$$\gcd(a, b) = \max\{f : f|a \text{ and } f|b\}.$$

By convention, $\gcd(0, 0) = 0$. If $\gcd(a_1, \dots, a_n) = 1$ then a_1, \dots, a_n are called *relatively prime* or *coprime*.

Measure of complexity: $\log n$ for an input n (bit-length of n)

Euclid's algorithm: Euclid's algorithm provides an efficient (polynomial time) computation of the gcd. It is based on a fundamental property of integers: division with remainder: For integers $a, b \neq 0$ there exist $m, r \in \mathbb{Z}$ with

$$a = mb + r \text{ and } 0 \leq r < |b|.$$

Euclid's algorithm:

Input: $a, b \in \mathbb{N}$.

Output: $r_{j-1} = \gcd(a, b)$.

Method: Set $r_{-1} = a$, $r_0 = b$. Compute by division with remainder successively r_1, \dots, r_{j-1} with $|b| > r_1 > \dots > r_{j-1} > r_j = 0$, until no remainder occurs. I.e., r_{i+1} is the remainder in the division of r_{i-1} by r_i :

$$\begin{aligned} r_{-1} &= m_1 r_0 + r_1, \\ r_0 &= m_2 r_1 + r_2, \\ &\vdots \\ r_{i-1} &= m_{i+1} r_i + r_{i+1}, \\ &\vdots \\ r_{j-3} &= m_{j-1} r_{j-2} + r_{j-1}, \\ r_{j-2} &= m_j r_{j-1}, \end{aligned}$$

with $m_1, \dots, m_j \in \mathbb{Z}$.

Termination: Since the remainders r_i are falling monotonically, the algorithm terminates after finitely many steps.

Correctness: r_{j-1} divides successively $r_{j-2}, r_{j-3}, \dots, r_0 = b$ and $r_{-1} = a$ (just consider successively the equations $r_{i-1} = m_{i+1}r_i + r_{i+1}$). If, conversely, z divides both a and b , then z successively also divides r_1, \dots, r_{j-1} (this results from the equations $r_{i+1} = r_{i-1} - m_{i+1}r_i$).

EXAMPLE. For $a = 9876$, $b = 3456$ we obtain

$$\begin{aligned} 9876 &= 2 \cdot 3456 + 2964, \\ 3456 &= 1 \cdot 2964 + 492, \\ 2964 &= 6 \cdot 492 + 12, \\ 492 &= 41 \cdot 12 (+0). \end{aligned}$$

I.e., $\gcd(9876, 3456) = 12$.

By substituting r_{j-1}, r_j, \dots, r_0 into the expression for r_j we also obtain a linear combination of the gcd in terms of the inputs a and b . (In the example this yields $\gcd(9876, 3456) = 12 = 7 \cdot 9876 - 20 \cdot 3456$.)

THEOREM 1.2. *For all integers $a, b > 0$ there exist integers x and y such that*

$$\gcd(a, b) = ax + by.$$

Moreover, the gcd and x and y can be computed in polynomial time.

PROOF. The worst-case running time of the algorithm can be bounded by considering a and b for which the maximal numbers of divisions occur. W.l.o.g. assume $a > b > 0$. (In case $b > a$, a and b are swapped in the first step.) The smallest pair (a, b) (in the sense $(a_1, b_1) < (a_2, b_2) \iff (b_1 < b_2 \text{ or } (b_1 = b_2 \text{ and } a_1 < a_2))$), for which j divisions are required, is obtained by choosing r_{j-1} and the m_i as small as possible: $m_1 = \dots = m_{j-1} = 1$, $m_j = 2$ and $r_{j-1} = 1$ ($m_j = 1$ is excluded, since $r_j = 0$ would then imply $r_{j-1} = r_{j-2}$). The equations

$$r_{j-1} = 1, \quad r_{j-2} = 2, \quad r_{i-1} = r_i + r_{i+1} \text{ for } i = j-2, \dots, 0$$

then determine $r_{-1} = a$ and $r_0 = b$ uniquely. This yields a connection to the Fibonacci sequence $0, 1, 1, 2, 3, 5, 8, 13, \dots$: For inputs $a > b > 0$, Euclid's algorithm requires at most $c \ln(b\sqrt{5})$ divisions, with $c = (\ln \frac{1+\sqrt{5}}{2})^{-1} \approx 2.08$. \square

2. \mathbb{Z}_m : Integers modulo m

When computing with integers it is often advantageous not to operate with the numbers themselves, but instead with the remainders which occur in the division by a fixed modulus $m \geq 2$.

Recall the ring \mathbb{Z}_m of integers modulo m .

DEFINITION 2.1. Let $m \in \mathbb{N}$ and $a, b \in \mathbb{Z}$. a and b are *congruent modulo m* if $m \mid (b - a)$, i.e., if $a \bmod m = b \bmod m$. Notation: $a \equiv b \pmod{m}$.

Let $\mathbb{Z}_m = \{0, 1, \dots, m-1\}$. In \mathbb{Z}_m we can define the addition and multiplication modulo m . $(\mathbb{Z}_m, +)$ forms a group under addition modulo m . Let $\mathbb{Z}_m^* = \{x \in \mathbb{Z}_m : \gcd(x, m) = 1\}$. Then \mathbb{Z}_m^* forms a group under multiplication modulo m . (Notice that $0 \notin \mathbb{Z}_m^*$.)

Q: How to compute the (multiplicative) inverse of an element $a \in \mathbb{Z}_m^*$ in polynomial time?

As an application of efficiently computing the inverse we present the following constructive version of the Chinese Remainder Theorem, which allows to decompose computational problems into smaller problems. In order to achieve this, an integer a is replaced by a tuple (a_1, \dots, a_k) of integers, which is obtained by considering a modulo k given, pairwise relatively prime moduli m_1, \dots, m_k . Since computations on a carry over in a canonical way to the a_i , we can work with the remainders modulo m_i .

This strategy requires that at the end of a computation one can reconstruct the number back from the remainders. An answer to this question of reconstructability is given by the Chinese Remainder Theorem, which in a special case was already known to Sun Tsu 300 AD.

THEOREM 2.2. Let $m_1, \dots, m_k \in \mathbb{N}$ pairwise relatively prime and $m := m_1 \cdot \dots \cdot m_k$. For any sequence of residues $a_1 \in \mathbb{Z}_{m_1}, \dots, a_k \in \mathbb{Z}_{m_k}$, there exists a unique $x \in \mathbb{Z}_m$ such that

$$x \equiv a_1 \pmod{m_1}, \quad x \equiv a_2 \pmod{m_2}, \quad \dots, \quad x \equiv a_k \pmod{m_k},$$

Moreover, a can be computed in polynomial time.

PROOF. *Existence and algorithm:* Let e_i , $1 \leq i \leq k$, be solutions of the system of equations for the special case $a_i = 1$, $a_j = 0$ for $j \neq i$ (“basis solutions”). Set

$$m'_i := \prod_{j \neq i} m_j = \frac{m}{m_i}.$$

Since the moduli m_1, \dots, m_k are pairwise coprime, we have $\gcd(m_i, m'_i) = 1$. Hence, there exists an inverse element t_i of m'_i modulo m_i , and it can be computed in polynomial time.

We set

$$e_i := m'_i t_i.$$

Then, by definition of m'_i , as desired

$$e_i \equiv \begin{cases} 1 & (\text{mod } m_i), \\ 0 & (\text{mod } m_j) \quad \text{for } j \neq i. \end{cases}$$

The basis solutions can be put together to a solution

$$x = \sum_{i=1}^k a_i e_i$$

of the original system.

Uniqueness: Uniqueness of the choice of r follows from a simple counting argument. The number of distinct choices of each r_i is m_i , and so there are exactly m distinct sequences (r_i) . By the existence proof each such sequence has at least one associated $r \in \mathbb{Z}_m$, which defines an injective mapping from $\mathbb{Z}_{m_1} \times \cdots \times \mathbb{Z}_{m_k} \rightarrow \mathbb{Z}_m$. However, since the cardinalities of the two sets are equal, the mapping must be bijective. \square

EXAMPLE. We ask for a solution to the system

$$x \equiv 2 \pmod{3}, \quad x \equiv 3 \pmod{5}, \quad x \equiv 4 \pmod{7}.$$

We have $m'_1 = 5 \cdot 7 = 35$, $m'_2 = 3 \cdot 7 = 21$, $m'_3 = 3 \cdot 5 = 15$. From

$$\begin{aligned} 1 &= \gcd(3, 35) = 12 \cdot 3 - 1 \cdot 35, \\ 1 &= \gcd(5, 21) = -4 \cdot 5 + 1 \cdot 21, \\ 1 &= \gcd(7, 15) = -2 \cdot 7 + 1 \cdot 15 \end{aligned}$$

we obtain the basis solutions $e_1 = -35$, $e_2 = 21$, $e_3 = 15$. Consequently,

$$2 \cdot (-35) + 3 \cdot 21 + 4 \cdot 15 = 53$$

is the solution to the congruence.

3. Euler's totient function and Fermat's Little Theorem

Computing in \mathbb{Z}_n^* is of fundamental importance for algorithm design and in modern cryptography (which is essentially based on prime number concepts).

DEFINITION 3.1. *Euler's totient function* $\varphi(n)$ is defined to be the number of elements of \mathbb{Z}_n that are coprime to n , i.e., $\varphi(n) = |\mathbb{Z}_n^*|$.

EXAMPLE. $\varphi(6) = 2$, $\varphi(8) = 4$, $\varphi(12) = |\{1, 5, 7, 11\}| = 4$.

If the prime factorization of a number n is known, then $\varphi(n)$ can be computed in polynomial time.

THEOREM 3.2. *Let n have the prime factorization $p_1^{e_1} \cdot p_2^{e_2} \cdots p_r^{e_r}$ with pairwise distinct primes p_i and exponents $e_i > 0$. Then*

$$\varphi(n) = \prod_{i=1}^r p_i^{e_i-1} (p_i - 1).$$

PROOF. The proof follows from the following three statements.

i) For prime p we have $\varphi(p) = p - 1$.

ii) For prime p and $e > 0$, $\varphi(p^e) = p^{e-1}(p - 1)$.

This follows from the observation that among all numbers in \mathbb{Z}_n exactly the numbers $0, p, 2p, \dots, (p^{e-1} - 1)p$ do not belong to $\mathbb{Z}_{p^e}^*$, i.e.,

$$\varphi(p^e) = p^e - p^{e-1} = p^e \left(1 - \frac{1}{p}\right).$$

iii) For m and n with $\gcd(m, n) = 1$, $\varphi(mn) = \varphi(m)\varphi(n)$.

By the Chinese Remainder Theorem, the mapping

$$\mathbb{Z}_{m \cdot n} \rightarrow \mathbb{Z}_m \times \mathbb{Z}_n, \quad a \bmod (m \cdot n) \mapsto (a \bmod m, a \bmod n)$$

is bijective. Since $\gcd(a, mn) = 1$ if and only if $\gcd(a, m) = 1$ and $\gcd(a, n) = 1$, the induced mapping

$$\mathbb{Z}_m^* \rightarrow \mathbb{Z}_m^* \times \mathbb{Z}_n^*, \quad a \bmod (m \cdot n) \mapsto (a \bmod m, a \bmod n)$$

is bijective as well. □

If the prime factorization is not known, then computing $\varphi(n)$ is essentially as hard as factoring. Euler's totient function is used, e.g., in the RSA coding scheme.

The following Theorem of Euler is of fundamental importance. As usual, we write $a^n := \underbrace{a \cdot \dots \cdot a}_{n \text{ times}}$.

THEOREM 3.3. (*Euler's Theorem.*) *For all $n \in \mathbb{N}$ and $a \in \mathbb{Z}_n^*$ we have $a^{\varphi(n)} \equiv 1 \pmod{n}$.*

PROOF. Let $\mathbb{Z}_n^* =: \{a_1, \dots, a_{\varphi(n)}\}$. Further let $a \in \mathbb{Z}_n^*$ arbitrary. Bijectivity of the left translation $L_a : \mathbb{Z}_n^* \rightarrow \mathbb{Z}_n^*, x \mapsto ax$ implies that then $\{aa_1, \dots, aa_{\varphi(n)}\} = \mathbb{Z}_n^*$ as well. Multiplication of all elements in the two sets yields

$$a_1 \cdot \dots \cdot a_{\varphi(n)} = aa_1 \cdot \dots \cdot aa_{\varphi(n)}$$

in \mathbb{Z}_n^* and after cancelling down $a^{\varphi(n)} = 1$. □

COROLLARY 3.4. (*Fermat's Little Theorem.*) *For prime p and $x \in \mathbb{Z}_p$,*

$$a^p \equiv a \pmod{p}.$$

PROOF. For $a \equiv 0 \pmod{p}$ the statement is obvious, and for $a \neq 0$ the property $\varphi(p) = p - 1$ implies $a^{p-1} \equiv 1 \pmod{p}$. \square

4. Quadratic residues

The exponentiation problem – to compute $y = x^a \pmod{n}$ given x , a and n – can be done in polynomial time (using iterative squaring). The inverse problem of root finding – given a , y and n , find an x with $y = x^a \pmod{n}$ – is in general much harder. Here, we describe an algorithm for finding square roots when n is prime.

DEFINITION 4.1. A number $a \in \mathbb{Z}_n^*$ is said to be a *quadratic residue* if there exists some $x \in \mathbb{Z}_n^*$ such that

$$a = x^2 \pmod{n}.$$

Otherwise a is called a *quadratic non-residue*.

We consider a prime modulus p . From elementary number theory, it is known that the multiplicative group \mathbb{Z}_p^* is cyclic. E.g. \mathbb{Z}_7^* is generated by the element 3, since $\{3^0 \pmod{7}, 3^1 \pmod{7}, 3^2 \pmod{7}, \dots, 3^6 \pmod{7}\} = \mathbb{Z}_7^*$.

LEMMA 4.2. *Let p be an odd prime and $g \in \mathbb{Z}_p^*$ be any generator. Then, g^k is a quadratic residue if and only if k is even.*

PROOF. If k even: clear.

If k odd: Let $k = 2l + 1$ and assume for contradiction that there exists an $x \in \mathbb{Z}_p^*$ such that $x^2 \equiv g^{2l+1} \pmod{p}$. But since g is a generator, $x = g^m$ for some $m \geq 0$. Hence, $g^{2m} \equiv g^{2l+1} \pmod{p}$, and switching to the additive group modulo $\varphi(p)$, we can restate this as $2m \equiv 2l + 1 \pmod{\varphi(p)}$. Since $\varphi(p) = p - 1$, we can conclude that $(p - 1) \mid (2l - 2m + 1)$. But $p - 1$ is even and $2l - 2m + 1$ is odd, giving a contradiction. \square

THEOREM 4.3. (Euler's Criterion.) *For prime p , an element $a \in \mathbb{Z}_p^*$ is a quadratic residue if and only if*

$$a^{\frac{p-1}{2}} \equiv 1 \pmod{p}.$$

PROOF. Let g be any generator of the cyclic group \mathbb{Z}_p^* . If a is a quadratic residue then let $x = g^k$ be a square root of a . Hence $a \equiv g^{2k} \pmod{p}$, and therefore

$$a^{\frac{p-1}{2}} \equiv g^{k(p-1)} \equiv (g^{p-1})^k \equiv 1^k \equiv 1 \pmod{p}.$$

Conversely, if a is not a quadratic residue, then by the previous lemma we know that a is an odd power of g . Assuming that $a = g^{2l+1}$, we obtain

$$a^{\frac{p-1}{2}} \equiv g^{l(p-1)} g^{\frac{p-1}{2}} \equiv g^{\frac{p-1}{2}} \pmod{p}.$$

Since g has order $p - 1$, the last term cannot be congruent to 1. \square

Using Euler's Criterion, there exists an efficient algorithm for deciding quadratic residuity.

COROLLARY 4.4. For prime p and $a \in \mathbb{Z}_p^*$ it can be decided in polynomial time whether a is a quadratic residue modulo p .

5. Primality testing

Motivation: e.g., generation of large primes for cryptography.

Primality testing is a fascinating and rich topic, and in 2002 Agrawal, Manindran, and Saxena have succeeded to show that this problem can be decided in polynomial time, see e.g., the survey article F. Bornemann, Primes is in P: A breakthrough for everyman, Notices of the AMS 05/2003, <http://www.ams.org/notices/200305/200305-toc.html>.

Many primality tests (also the AKS algorithm) are essentially based on ideas of Fermat's Little Theorem. This theorem tells us: Let $N \in \mathbb{N}$. If an $a \in \mathbb{N}$, $0 < a < N$ with $a^{N-1} \not\equiv 1 \pmod{N}$ exists, then N is not prime.

DEFINITION 5.1. Let N be a composite number. N is called *pseudoprime* w.r.t the basis a if N satisfies the property $a^{N-1} \equiv 1 \pmod{N}$. N is called *Carmichael number* if N is pseudoprime for all $a \in \mathbb{Z}_N^*$.

The numbers $a \in \mathbb{Z}_N^*$ which satisfy the Fermat identity constitute a subgroup of \mathbb{Z}_N^* . The order of this subgroup is either $\varphi(N)$ or (in case of a proper subgroup) at most $\varphi(N)/2$. Thus:

THEOREM 5.2. (*r-fold Fermat test.*) Let $N \in \mathbb{N}$ such that there exists an $a \in \mathbb{Z}_N^*$ with $a^{N-1} \not\equiv 1 \pmod{N}$. Then for randomly chosen, independent $a_1, \dots, a_r \in \mathbb{Z}_N^*$,

$$\text{Prob}(a_i^{N-1} \equiv 1 \pmod{N} \text{ for } 1 \leq i \leq r) \leq 2^{-r}.$$

The r -fold Fermat test can recognize the compositeness of a non-Carmichael number with probability $\geq 1 - 2^{-r}$. It cannot recognize the compositeness of a Carmichael number. For many practical purposes, the Fermat test is sufficient, since Carmichael numbers occur very seldomly. In fact, until the early 90's it was even unknown if there are infinitely many Carmichael numbers at all.

The following randomized algorithm (*Miller-Rabin test*) extends the Fermat test to Carmichael numbers. Let N be odd and consider a^{N-1} for a random $a \in \mathbb{Z}_N \setminus \{0\}$. If this is not 1, then we have proved that N is composite. Otherwise, we keep replacing this (even) power of a by its precomputed square root until the result is something other than ± 1 or until we have reduced it to an odd power of a . If we reach a square root of 1 other than ± 1 then N is composite (because for a prime number N any quadratic residue has exactly two square roots). Otherwise the algorithm claims that N is prime, and this is the only place where it can make an error.

Primality test of Miller-Rabin:

Input: Odd number N .

Output: PRIME or COMPOSITE.

- (1) Compute r and R such that $N - 1 = 2^r R$, and R is odd.
- (2) Choose a uniformly at random from $\mathbb{Z}_N \setminus \{0\}$.
- (3) For $i = 0$ to r compute $b_i = a^{2^i R}$.
- (4) If $a^{N-1} = b_r \not\equiv 1 \pmod{N}$ then return COMPOSITE.
- (5) If $a^R = b_0 \equiv 1 \pmod{N}$ then return PRIME.
- (6) Let $j = \max\{i : b_i \not\equiv 1 \pmod{N}\}$.
- (7) If $b_j \equiv -1 \pmod{N}$ then Return PRIME Else Return COMPOSITE.

For prime N , this algorithm always returns PRIME. Moreover, it can be shown that the algorithm returns PRIME on a composite input N with probability at most $1/2$:

THEOREM 5.3. *Let $N \in \mathbb{N}$ be odd and composite, $N - 1 = 2^r R$ with R odd. Then for random $a \in \mathbb{Z}_N^*$ we have*

$$\text{Prob}\left(a^R \equiv 1 \pmod{N} \text{ or } \exists i \in \{0, \dots, r-1\} \quad a^{2^i R} \equiv -1 \pmod{N}\right) \leq \frac{1}{2}.$$

An **RP** algorithm \mathcal{A} for a given problem is a randomized algorithm running in worst-case polynomial time such that

- (1) for every YES instance $\Pr(\mathcal{A} \text{ yields YES}) \geq 1/2$;
- (2) for every NO instance $\Pr(\mathcal{A} \text{ yields YES}) = 0$.

Hence:

THEOREM 5.4. *The Miller-Rabin test is an **RP** algorithm for testing compositeness.*