Basic Algorithms in Number Theory

Francesco Pappalardi

Polynomials, Hensel's Lemma, Chinese Remainder Theorem and more.

July 22th 2010

Let's play with $2^{2067}+131$

Let $p = 2^{2067} + 131$. Is it prime?

Do we believe Mathematica?

No we do not believe her!!!
So let us check it with Solovay Strassen (from yesterday Lab)

Exercise: Check that she is right with Miller-Rabin Test.

Can we prove that certainly p is prime maybe by factoring $p-1$?

Answer: NOWAY!!

We want to compute the square root of $5 \bmod p$

Can we do it? We ask Mathematica.

Yes, so let us have a look at the slide about it on Lecture 2.

PROBLEM 9. Square Roots Modulo a prime:

Given an odd prime p and a quadratic residue a, find x s. t. $x^2 \equiv a \bmod p$

It can be solved efficiently if we are given a quadratic nonresidue $g \in (\mathbb{Z}/p\mathbb{Z})^*$

1. We write $p-1 = 2^k \cdot q$ and we know that $(\mathbb{Z}/p\mathbb{Z})^*$ has a (cyclic) subgroup G with 2^k elements
2. Note that $b = g^q$ is a generator of G and that $a^q \in G$
3. Use the Pohlig-Hellmann Algorithm to compute t such that $a^q = b^t$.
4. Finally set $x = a^{(p-q)/2} b^{t/2}$ and observe that
$$x^2 = a^{(p-q)} b^t = a^p \equiv a \bmod p.$$

Solution of $X^2 \equiv 5(\mathrm{mod} 2^{2067} + 131)$

The first thing we need is a quadratic residue modulo p and we ask Mathematica.

Exercise: Find the least quadratic non residue.

Now we observe that $p - 1 = 2 \times q$ with q odd so that $q = (p-1)/2$.

Hence Part 2. is easy since $b = g^{(p-1)/2} \equiv p - 1 \bmod p$ and what about $5^{(p-1)/2}$?

We do NOT ask Mathematica since we know that it is one!

Therefore $t = 0$ (even as expected) and

$x = 5^{(p-q)/2}(-1)^{t/2} \bmod p$ DONE!

Exercise (To do in Mathematica). Compute the roots of $X^2 \equiv 6(\mathrm{mod} 2^{2067} + 2949)$ and of $X^2 \equiv 10(\mathrm{mod} 2^{2067} + 2949)$

Polynomials in $(\mathbb{Z}/n\mathbb{Z})[X]$

A polynomial $f \in (\mathbb{Z}/n\mathbb{Z})[X]$ is

$$f(X) = a_0 + a_1 X + \cdots + a_k X^k \quad \textbf{where} \quad a_0, \ldots, a_k \in \mathbb{Z}/n\mathbb{Z}$$

The degree of f is $\deg f = k$ when $a_k \neq 0$.

Example: If $f(X) = 5 + 10X + 21X^3 \in \mathbb{Z}[x]$, then we can "reduce" it modulo n. So

$f(X) \equiv X^3 \bmod 5$ which is the same as saying: $f(X) = X^3 \in \mathbb{Z}/5\mathbb{Z}[X]$.

$f(X) \equiv 2 + X \bmod 3$ which is the same as saying: $f(X) = 2 + X \in \mathbb{Z}/3\mathbb{Z}[X]$.

$f(X) \equiv 5{+}3X \bmod 7$ which is the same as saying: $f(X) = 5{+}3X \in \mathbb{Z}/7\mathbb{Z}[X]$.

For the time being we restrict ourselves to the case of $f \in \mathbb{Z}/p\mathbb{Z}[X]$. The fact that $\mathbb{Z}/p\mathbb{Z}$ is a field is important. (Notation $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ to remind us this)

We can add, subtract and multiply polynomials in $\mathbb{F}_p[X]$.

Polynomials in $\mathbb{F}_p[X]$

We can also divide them!! for $f, g \in \mathbb{F}_p[X]$ there exists q, $r \in \mathbb{F}_p[X]$ such that

$$f = qg + r \quad \text{and} \quad \deg r < \deg g.$$

Example: Let $f = X^3 + X + 1, g = X^2 + 1 \in \mathbb{F}_3[X]$. Then

$$X^3 + X + 1 = (X^2 + X + 2)(X + 1) + 2 \quad \text{so that } q = X^2 + X + 2, r = 2$$

In Mathematica:

```
PolynomialQuotientRemainder[x^ 3 + x + 1, x + 1, x, Modulus -> 3]
```

finds p and r.

Polynomials in $\mathbb{F}_p[X]$

The complexity for summing or subtracting $f, g \in \mathbb{F}_p[X]$ with $\max\{\deg f, \deg g\} < n$, is $O(\log p^n)$. Why?

The complexity of multiplying or dividing $f, g \in \mathbb{F}_p[X]$ with $\max\{\deg f, \deg g\} < n$, can be shown to be $O(\log^2(p^n))$.

Important difference: Polynomials in $\mathbb{F}_p[X]$ are not invertible except when they are constant but not zero. So $\mathbb{F}_p[X]$ looks much more like $\mathbb{Z}$ than like $\mathbb{Z}/m\mathbb{Z}$.

But if $f, g \in \mathbb{F}_p[X]$, the $\gcd(f, g)$ exists and it is fast to calculate!!! why?

YES! The EEA also applies to $\mathbb{F}_p[X]$ (Indeed it applies when there is a true division)

Polynomials in $\mathbb{F}_p[X]$

Example Let $f = X^3 + X^2 + X + 1$, $g = X^3 + X + 1 \in \mathbb{F}_2[X]$, Then

- $f = 1(g) + X^2$;
- $g = X(X^2) + X + 1$;
- $X^2 = (X + 1)(X + 1) + 1$;
- $X + 1 = (X + 1)1 + 0$.

So the sequence of quotients are $1, X, X + 1, X + 1 \in \mathbb{F}_2[X]$ and we can apply the recursions to compute the Bezout Identity.

However in Mathematica:

```
PolynomialGCD[(x+1)^ 3,x^ 3+x, Modulus -> 2]
PolynomialExtendedGCD[1+X+X^ 2+X^ 3,1+X+X^ 3, Modulus -> 2]
```

Polynomials in $\mathbb{F}_p[X]$

As in $\mathbb{Z}$ every $f \in \mathbb{F}_p[X]$ can be written as the product of irreducible polinomials.

Mathematica Knows how to do it:

```
Factor[x^ 3-3x^ 2-2x+6,Modulus -> 3]
```

The polynomial $X^p - X \in \mathbb{F}_p[X]$ is very special. What is its factorization?

$$X^p - X = \prod_{a \in \mathbb{F}_p} (X - a) \in \mathbb{F}_p[X].$$

Why is it true?

FLT says that $a^p = a, \forall a \in \mathbb{F}_p$. Let's Look at one example.

PROBLEM 12. IRREDUCIBILITY TEST FOR POLYNOMIALS IN $\mathbb{F}_p$:

Given $f \in \mathbb{F}_p[X]$, determine if f is irreducible:

Theorem. *Let* $X^{p^n} - X \in \mathbb{F}_p[X]$. *Then*

$$X^{p^n} - X = \prod_{\substack{f \in \mathbb{F}_p[X] \\ f \text{ irreducible} \\ f \text{ monic} \\ \deg f \text{ divides } n}} f$$

We cannot prove it here but we deduce an algorithm:

```
Input:  f ∈ F_p[X] monic
Output: ``Irreducible'' or ``Composite''
1.  n := deg f
2.  For j = 1, ..., ⌈n/2⌉
            if gcd(X^{p^j} − X, f) ≠ 1 then
                Output ``Composite'' and halt.
3.  Output ``Irreducible''.
```

Polynomial equations modulo prime and prime powers

Often one considers the problem of finding roots of polynomial $f \in \mathbb{Z}/n\mathbb{Z}[X]$.

When $n = p$ is prime then one can exploit the extra properties coming from the identity

$$X^p - X = \prod_{a \in \mathbb{F}_p} (X - a) \in \mathbb{F}_p[X].$$

From this identity it follows that $\gcd(f, X^p - X)$ is the product of liner factor $(X - a)$ where a is a root of f.

Similarly we have that

$$X^{(p-1)/2} - 1 = \prod_{\substack{a \in \mathbb{F}_p \\ \left(\frac{a}{p}\right) = 1}} (X - a) \in \mathbb{F}_p[X].$$

This identity suggests the Cantor Zassenhaus Algorithm

Cantor–Zassenhaus Algorithm

```
CZ(p)
Input:  a prime p and a polynomial f ∈ F_p[X]
Output:  a list of the roots of f
1.  f := gcd(f(X), X^p − X) ∈ F_p[X]
2.  If deg(f) = 0 Output ‘‘NO ROOTS’’
3.  If deg(f) = 1,
      Output the root of f and halt
4.  Choose b at random in F_p
      g := gcd(f(X), (X + b)^((p−1)/2))
      If 0 < deg(g) < deg(f)
      Output CZ(g) ∩ CZ(f/g)
      Else goto step 3
```

The algorithm is correct since f in (Step 4) is the product of $(X - a)$ (a root of f). So g is the product of $X - a$ with $a + b$ quadratic residue.

CZ(p) has polynomial (probabilistic) complexity in $\log p^n$.

Polynomial equations modulo prime powers

There is an explicit contruction due to Kurt Hensel that allows to "lift" a solution of $f(X) \equiv 0 \bmod p^n$ to a solution of $f(X) \equiv 0 \bmod p^{2n}$.

Example: (Square Roots modulo Odd Prime Powers. Suppose $x \in \mathbb{F}_p$ is a square root of $a \in \mathbb{F}_p$.

Let $y = (x^2 + a)/2x \bmod p^2$ (y is well defined since $\gcd(2x, p^2) = 1$). Then

$$y^2 - a = \frac{(x^2 - a)^2}{4x^2} \equiv 0 \bmod p^2$$

since p^2 divides $(x^2 - a)^2$.

The general story if the famous Hensel's Lemma.

Polynomial equations modulo prime powers

Theorem (Hensel's Lemma). *Let p be a prime, $f(X) \in \mathbb{Z}[X]$ and $a \in \mathbb{Z}$ such that*

$$f(a) \equiv 0 \bmod p^k, \qquad f'(a) \not\equiv 0 \bmod p.$$

Then $b := a - f(a)/f'(a) \bmod p^{2k}$ is the unique integer modulo p^{2k} that satisfies

$$f(b) \equiv 0 \bmod p^{2k}, \qquad b \equiv a \bmod p^k.$$

Proof. Replacing $f(x)$ by $f(x+a)$ we can restric to $a = 0$. Then

$$f(X) = f(0) + f'(0)X + h(X)X^2 \quad \text{where } h(X) \in \mathbb{Z}[X].$$

Hence if $b \equiv 0 \bmod p^k$, then $f(b) \equiv f(0) + bf'(0) \bmod p^{2k}$. Finally $b = -f(0)/f'(0)$ is the unique lift of 0 modulo p^{2k} that satisfies $f(b) \equiv 0 \bmod p^{2k}$.□

Chinese Remainder Theorem

CHINESE REMAINDER THEOREM. *Let* $m_1, \dots, m_s \in \mathbb{N}$ *pairwise coprime and let* $a_1, \dots, a_s \in \mathbb{Z}$. *Set* $M = m_1 \cdots m_s$. *There exists a unique* $x \in \mathbb{Z}/M\mathbb{Z}$ *such that*

$$\begin{cases} x \equiv a_1 \bmod m_1 \\ x \equiv a_2 \bmod m_2 \\ \quad \vdots \\ x \equiv a_s \bmod m_s. \end{cases}$$

Furthermore if $a_1, \dots, a_s \in \mathbb{Z}/M\mathbb{Z}$, *then* x *can be computed in time* $O(s \log^2 M)$.

Chinese Remainder Theorem continues

PROOF. Let us first assume that $s = 2$. Then from EEA we can write $1 = m_1x + m_2y$ for appropriate $x, y \in \mathbb{Z}$. Consider the integer

$$c = a_1m_2y + a_2m_1x.$$

Then $c \equiv a_1 \bmod m_1$ and $a \equiv a_2 \bmod m_2$. Furthermore if c' has the same property, then $d = c - c'$ is divisible by m_1 and m_2. Since $\gcd(m_1, m_2) = 1$ we have that m_1m_2 divides d so that $c \equiv c' \bmod m_1m_2$.

If $s > 2$ then we can iterate the same process and consider the system:

$$\begin{cases} x \equiv c \bmod m_1m_2 \\ x \equiv a_3 \bmod m_3 \\ \quad \vdots \\ x \equiv a_s \bmod m_s. \end{cases} \quad . \quad \square$$

In Mathematica, `ChineseRemainder[`$\{3, 4\}, \{4, 5\}$`]` coincides with

$$\begin{cases} x \equiv 3 \bmod 4 \\ x \equiv 4 \bmod 5 \end{cases}$$

Chinese Remainder Theorem (applications)

It can be used to prove the multiplicativity of the Euler φ function. More precisely, it implies that, if $\gcd(m,n)=1$, then the map:

$$(\mathbb{Z}/mn\mathbb{Z})^* \to (\mathbb{Z}/m\mathbb{Z})^* \times (\mathbb{Z}/n\mathbb{Z})^*, a \mapsto (a \bmod m, a \bmod n)$$

is surjective.

It can be used to glue solutions of congruence equations.

Let $f \in \mathbb{Z}[X]$ and suppose that $a, b \in \mathbb{Z}$ are such that

$$f(a) \equiv (\bmod n), \quad f(b) \equiv (\bmod m).$$

If $\gcd(m,n)=1$, then a solution c of

$$\begin{cases} x \equiv a \bmod n \\ x \equiv b \bmod m \end{cases}$$

has the property that $f(c) \equiv 0 (\bmod nm)$.

Algorithms to be implemented in Mathematica (Lectures 1)

1. Right-to-Left Exponentiation in $\mathbb{Z}/m\mathbb{Z}$
2. Left-to-Right Exponentiation in $\mathbb{Z}/m\mathbb{Z}$
3. Test of Primality using the factorization of $n-1$
4. Computation of Legendre/Jacobi Symbols (via recursive algorithm)
5. Solovay Strassen probabilistic Primality Test
6. Probabilistic Search of Quadratic Nonresidues
7. Deterministic Search of Quadratic Nonresidues
8. Power test via the newton Method
9. Miller Rabin probabilistic primality test
10. Implementation of RSA
11. Pollard ρ method and $n-1$ method

Algorithms to be implemented in Mathematica (Lectures 2/3)

1. Search for primitive root in $n = 2; 4; p^\alpha; 2p^\alpha$ (with resident commands)
2. Shank's BSGS for Discrete Logs
3. Pohlig-Hellman Algorithm for groups with $|G| = 2^\alpha$.
4. Algorithm to compute square root modulo a prime
5. Binary Euclidean Algorithms
6. Extended Euclidean Algorithm (EEA) for Bezout identity
7. Cantor--Zassenhaus Algorithm
8. Lifting roots modulo powers of primes
9. Chinese Remainder Theorem
10. *Finite fields on Mathematica*
11. *Elliptic curves in Mathematica*
12. *The Riemann Zeta function in Mathematica*