Integer and Polynomial Factoring Ideas

CS498A: Undergraduate Project (UGP) report submitted to

Indian Institute of Technology Kanpur

Bachelor of Technology

in

Computer Science and Engineering

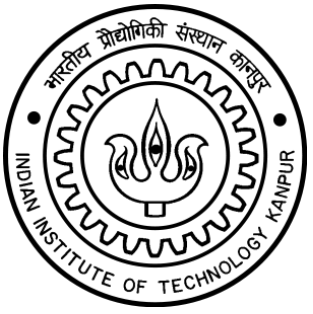
by

Rishabh Batra

(170568)

Under the supervision of

Dr. Nitin Saxena



Department of Computer Science and Engineering

Indian Institute of Technology Kanpur

Winter Semester, 2020-21

December 8, 2020

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the project and giving their details in the references.

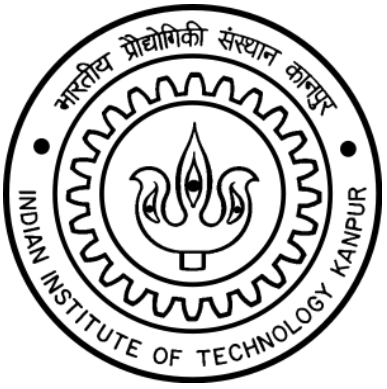
Date: December 8, 2020 (Rishabh Batra)

Place: Kanpur (170568)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

KANPUR - 208016, INDIA



CERTIFICATE

This is to certify that the project report entitled "**Integer and Polynomial Factoring Ideas**" submitted by **Rishabh Batra** (Roll No. 170568) to Indian Institute of Technology Kanpur towards no/partial fulfilment of requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering is a record of bona fide work carried out by him under my supervision and guidance during Winter Semester, 2020-21.

Date: December 8, 2020

Place: Kanpur

Dr. Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur - 208016, India

Abstract

Name of the student: **Rishabh Batra** Roll No: **170568**

Degree for which submitted: **Bachelor of Technology**

Department: **Computer Science and Engineering**

Project title: **Integer and Polynomial Factoring Ideas**

Project supervisor: **Dr. Nitin Saxena**

Month and year of Project submission: **December 8, 2020**

A very important open problem in Computer Science is getting an efficient algorithm for factoring integers. Almost all the currently used methods of security work on the assumption that integer-factoring is hard. But it was shown by Shor that integer factoring is possible efficiently on a Quantum Computer. In this project, we try to explore some classical approaches for efficient integer factoring.

The problem of polynomial factoring also has been widely studied in computer science and has widespread applications. There are known efficient algorithms for factoring polynomials over finite fields [Ber68]. But the question of factoring polynomials over rings is a much harder question. In this report, we studied previous efforts of efficiently factoring polynomials $\mod p^k$ and tried to extend the results. We also studied about the Hilbert's Nullstellensatz problem (and its extension to mod p^k and how it could help in factoring polynomials).

Acknowledgements

I am deeply indebted to Dr. Nitin Saxena for giving me this opportunity to work under him; this project would have been nothing short of impossible had he not guided me for this project and helped me to complete . I thank him for being patient with me and also for the frequent discussions we had, which were extremely edifying. I would also like to thank my family members for the incredible and unconditional support they bestowed upon me during the span of this project.

A part of this work was done in collaboration with Sayak Chakraborti and Diptajit Roy. I would like to thank Ashish Diwedi for helpful discussions on ideas for factoring polynomials.

Contents

Chapter 1

Introduction

A very famous and important open problem in Computer Science is getting an efficient algorithm for factoring integers. Almost all the currently used methods of security and cryptography (eg RSA) work on the assumption that integer-factoring is hard. But it was shown by Shor that integer factoring is possible efficiently on a Quantum Computer. In this, we try to explore some classical approaches for efficient integer factoring. We describe a paper [APR83] that gives a probabilistic primality test and our efforts to modify that algorithm to factorise integers. We also describe ideas of using automorphisms and endomorphisms of cyclotomic fields to factorise integers and how bivariate extensions for solving the problem may help.

Polynomial factorization has been a famous question in the fields of mathematics and theoretical computer science. There has been extensive work in this area and various factoring algorithms over various fields are present, for e.g. over finite fields [Ber68] , over rationals , number fields, p-adic fields etc. However factorization in rings of the form $\mathbb{Z}_{p^k}[x]$ becomes increasingly difficult. Given a univariate integral polynomial f(x) and a prime power p^k , with p prime, efficient algorithms are not known. In this report, we study the previous work done in the area of factoring polynomials mod p^k and try to come up with ideas to extend it.
We describe an algorithm [DMS19] which solves the problem of factorization for $k \leq 4$, i.e., this paper solves this problem upto modulo p^4 (gives factorization modulo p^2, p^3 and p^4) and states the problems faced for modulo p^5 with increased number of variables. If mod p, the factorisation was into co-prime factors then Hensel's

lemma could have been used. So the only case that needs to be dealt with is when f is power of an irreducible polynomial modulo p. This is solved by reducing this problem to "root finding" over some other idea. Our goal is to try to extend these ideas to factorisation $\mod p^k$ for $k \geq 5$ We also give how our problem of factoring polynomials $\mod p^k$ related to extending the Hilbert's Nullstellensatz problem.

Chapter 2

Preliminaries

2.1 Preliminaries

2.1.1 Representative Roots

Rings of the form $\mathbb{Z}/\langle p^k \rangle[x]$ for some prime p and integer $k \geq 1$ are not unique factorization domains and hence polynomials of degree d can have more than d roots (infact exponentially many are possible). [JBQ13] devised a compact datastructure for this to represent these exponentially many roots in polynomial space, which was further used in [DMS19].

Representative roots are roots of a polynomial written in the form $a + p^i*$ which means that all the roots have the form $a + p^i y$ for every $y \in \mathbb{Z}_{p^{k-i-1}}$. The symbol $*$ represents the entire ring $y \in \mathbb{Z}_{p^{k-i-1}}$ and a is a p-adic integer. This can be seen as a compact datastructure to represent exponentially many values (exponential in $\log p$). For a in the form $a_0 + a_1 p + a_2 p^2 + \ldots a_{i-1} p^{i-1}$ for some a_j's $\in \mathbb{Z}_p$ then we have the set of representative roots as

$$S_a = \{a_0 + a_1 p + a_2 p^2 + \ldots a_{i-1} p^{i-1} + y_1 p^i + y_2 p^{i+1} + \cdots + y_{k-i} p^{k-1} | y_j \in \mathbb{Z}_p\}$$

We will consider $*$ as an entire ring, R and the following convention of writing will be followed:

- $u + \{*\} = \{u + *\}, u\{*\} = \{u*\} \forall u \in R$
- $c + \{a + b*\} = \{(a + c) + b*\}, c\{a + b*\} = \{ca + cb*\} \forall a, b, c \in R$

2.1.2 Quotient Ideal

Given two ideals I and J of a commutative ring R, we define the quotient of I by J as [DMS19],

$$I : J := \{a \in R | aJ \subset I\} \tag{2.1}$$

Claim 1 (Cancellation): Suppose I is an ideal of ring R and a, b, c are three elements in R. By definition of quotient ideals, $ca \equiv cb \mod I$ iff $a \equiv b \mod I : \langle c \rangle$.
Claim 2: Let p be a prime and $\phi \in (Z/\langle p^k \rangle)[x]$ be such that $\phi \not\equiv 0 \mod p$. Given an ideal $I := \langle p^l, \phi^m \rangle$of Z[x],
1.$I : \langle p^i \rangle = \langle p^{l-i}, \phi^m \rangle$, for $i \leq l$, and
2.$I : \langle \phi^j \rangle = \langle p^l, \phi^{m-j} \rangle$ for $j \leq m$.
Proof: We prove part 1. Part 2 is similar.If $c \in \langle p^{l-i}, \phi^m \rangle$ then there exists $c_1, c_2 \in Z[x]$, such that,$c = c_1 p^{li} + c_2 \phi^m$. Multiplying by p^i, we get $p^i c = c_1 p^l + p^i c_2 \phi^m \implies c \in I : \langle p^i \rangle$
For reverse direction, suppose $c \in I : \langle p^i \rangle$, then then there exists $c_1, c_2 \in Z[x]$, such that,$p^i c = c_1 p^l + c_2 \phi^m$. Since $i \leq l$ and $p \not| \phi$, $p^i | c_2$. So $c = c_1 p^{l-i} + (\frac{c_2}{p^i})\phi^m \implies c \in \langle p^{l-i}, \phi^m \rangle$.

2.1.3 Gauss Quadratic Sums

Let p be an odd prime number and a an integer. Then the Gauss sum modulo p, g(a; p) [Wiki], is the following sum of the pth roots of unity:

$$g(a; p) = \sum_{n=0}^{p-1} e^{\frac{2\pi i a n^2}{p}} = \sum_{n=0}^{p-1} \zeta_p^{an^2}, \qquad \zeta_p = e^{\frac{2\pi i}{p}}.$$

The exact value of the Gauss sum, computed by Gauss, is given by the formula

$$g(1; p) = \sum_{n=0}^{p-1} e^{\frac{2\pi i n^2}{p}} = \begin{cases} \sqrt{p} & \text{if } p \equiv 1 \pmod 4, \\ i\sqrt{p} & \text{if } p \equiv 3 \pmod 4. \end{cases}$$

$g(1;p)^2 = \left(\frac{-1}{p}\right) p$ was easy to prove and led to one of Gauss's proofs of quadratic reciprocity. However, the determination of the sign of the Gauss sum turned out to be considerably more difficult and even Gauss could only establish it after several years' work.

2.1.4 B-near prime ideals

We define the concept of B-near prime ideals which are necessary for our algorithms for power residue computation:

B-near prime numbers- A natural number N is said to be B-near prime if $N = p\prod_{i=1}^{k} p_i$ with $p_i \leq B, p_i's$ can be repeated.

B-near prime ideals-Ideal a of R is called B-near prime if its norm $N(a)$ is B-near prime.

Advantage of B-near prime ideals-They are efficiently recognisable and computable in poly-time when B is polynomially bounded by the degree n=[K:Q]. Also, their factorization can be computed efficiently.

Chapter 3

Integer Factoring-Previous work

We describe the main ideas from the paper [APR83] and how we tried to extend these further. We will

The paper gives an algorithm for primality testing, which I will call APR algorithm from now. We tried to modify it for factoring. Given n, the [APR83] algorithm decides in time $(logn)^{c\,log\,log\,log\,n}$ (nearly polynomial time) whether it is prime or composite. It works on principles of Power residue computations on cyclotomic fields and consists of many pseudo-primality tests such that if n passes all the tests, its factors lie in a small, explicit set.

3.1 Main Ideas of APR algorithm

Let I be a set of primes. A Euclidean prime with respect to I is a prime q such that q-1 is sqaure free and every prime factor of q-1 lies in I. In the algorithm, we try to find a small set of initial primes I=I(n) ($=\{p_1, p_2, ...\}$) such that the product of Euclidean primes(denoted by $q_1, q_2, ...$) exceeds $\sqrt{n}$.

Let t_q be a primitive element of the field F_q. Then $Ind_q(x)$ is the smallest non-negative integer for which x = $t_q^{Ind_q(x)}$ mod q

For a possible r ($\leq \sqrt{n}$) dividing n, we want to find r mod q for each Euclidean

prime q. As the product of these Euclidean primes is greater than r, we can get r from the above using Chinese Remaindering. So we try to find $Ind_q(r)$ for every Euclidean prime q which would give us r mod q for each q.
For this, we try to calculate $Ind_q(r)$ mod p for each initial prime p dividing q-1.Now the size of the multiplicative group F_q^* is q-1. So by Chinese Remaindering would give us $Ind_q(r)$ mod (q-1) which is the same as $Ind_q(r)$.

So our goal- Try to know $Ind_q(r)$ mod p for each initial prime p dividing q-1. This would give us r.

Transition data- For a fixed initial prime p, we find "transition data" relating $Ind_q(r)$ mod p for all q such that p divides q-1 such that if one of them is known, the others can be calculated in terms of it. The transition data is obtained by using some "mock residue symbols" (which are related to p^{th} power residue symbols) and "Extraction lemma".

3.2 Tools used

Cyclotomic field of a prime p - Its irreducible polynomial over Q is $\phi_p(x) = x^{p-1} + x^{p-2}...x + 1$. We know $\phi_p(x)$ factorises mod q into equidegree irreducible factors of degree f = order of q in (Z/pZ)*
The polynomial $\phi_p(x)$ factors mod q as $\phi_p(x) = \prod_{i=1}^{g} h_i(x)$ mod q where $h_i(x)$ are distict monic polynomials of degree f which are irreducible mod q and g=$\frac{p-1}{f}$

We now consider power residue symbols in the special case where q is a rational prime and p — q-1. Let $t = t_q$ be a primitive root for q. Then
$\phi_p(x) = \prod_{i=1}^{p-1}(x - t^{(\frac{q-1}{p})i}$ mod q (here we use the fact that $t^{(\frac{q-1}{p})}$ is a primitive pth root of unity)
So we have "canonical" prime Q lying over (q) in $Z[\zeta_p]$ as
$Q = (q, \zeta_p - t^{(\frac{q-1}{p})})$
Now, $(\frac{x}{Q})_p = x^{\frac{q-1}{p}}$ (Here size of Q is easily seen to be q)

$= t^{Ind_q(x)\frac{q-1}{p}} = \zeta_p^{Ind_q(x)} mod Q$
Thus knowing $(\frac{x}{Q})_p$ for the canonical prime Q is equivalent to knowing $Ind_q(x)$ mod p, which was our goal.

Jacobi Sums: If Q is a prime of $Q(\zeta_p)$ not dividing (p) and a,b $\in Z$ then the Jacobi sum is defined as:
$J_{a,b}(Q) = \sum' (\frac{xQ)}{p^{-a}(\frac{1-x}{Q})_p^{-b}}$ where the sum is over a set of coset representatives of $z[\zeta_p]/Q$ other than 0,1 mod Q. The prime factorisation of $(J_{a,b}(Q))$ is given by the following theorem

Theorem 3.1. *[APR83] Suppose a,b $\in$ Z with ab(a+b)$\neq$ 0 mod p. For u $\in$ Z, define* $\theta(u)_{a,b} = [\frac{a+b}{p}u] - [\frac{a}{p}u] - [\frac{b}{p}u]$ *(it is either 0 or 1), then*

$$(J_{a,b}(Q)) = \prod_{n=1}^{p-1} \sigma_u^{-1}(Q)^{\theta_{a,b}(u)}$$

Theorem 3.2. *[APR83] If p > 2, there exist a, b in Z such that ab(a + b)$\neq$ 0 mod p and we define*
$\hat{\theta}_{a,b} = \sum_{u=1}^{p-1} \theta_{a,b}(u).u^{-1} \neq 0$ *mod p where* u^{-1} *is inverse of u mod p.*

Proof: Note that for 1 < u < p - 1, we have $[\frac{u}{p}] = 0$ and $[\frac{p-1}{p}u] = u - 1$
Then $\sum_{m=1}^{p-2} \hat{\theta}_{m,1} = \sum_{m=1}^{p-2}\sum_{u=1}^{p-1}([\frac{m+1}{p}u] - [\frac{m}{p}u]).u^{-1} = \sum_{u=1}^{p-1}[\frac{p-1}{p}u].u^{-1}$
$= \sum_{u=1}^{p-1}(u-1).u^{-1} = p - 1 \neq 0 \text{mod } p$
So there exists atleast one good pair (a,b) = (m,1) for m=1,2...p-2

3.3 Probablisitic Version of the APR Algorithm

3.3.1 Preparation

In the preparation step, we define the initial primes and calculate the transition data. We first compute f(n), the least square-free natural number such that

$$\prod_{q-1|f(n),q\in primes} q > n^{\frac{1}{2}}$$

We define initial primes for n to be the prime factors p of f(n) and the Euclidean primes for n to be the primes q for which $q-1|f(n)$

It is shown that $f(n) \leq (logn)^{c_o \, log\, log\, log\, n}$ for all large n

Note it can be easily seen that the time taken to compute f(n) will be of the order $f(n)^c$. We then compute and fix a primitive root t_q for each Euclidean prime q and check that n is not divisible by p or q.For each initial prime p > 2, find a, b ∈ Z such that 0 < a, b < p,

$$\hat{\theta}_{a,b} = \sum_{u=1}^{p-1} \theta_{a,b}(u).u^{-1} \neq 0 \text{ mod p}$$

Compute a "Jacobi sum" $J_p(q)$ for each initial prime p and Euclidean prime q with $p|q-1$ as follows:

If p = 2, put $J_p(q) = -q$

If p > 2, let $J_p(q) = -J_{a,b}(Q)$ where a,b are from previous step

We have $(J_p(q), r)_\lambda$=1

Then for each p,we factor n into ideals in $Z[\zeta_p]$ as it would happen if n was prime. We attempt to do this as follows. Let f be the order of n in $(Z/pZ)*$, put g = (p - 1)/f, and try to factor $\phi_p(x) = \prod_{i=1}^{g} h_i(x)$ mod n

where each $h_i(x) \in Z[x]$ is monic and has degree f. If n is in fact prime, then with high probability of success $\phi_p(x)$ can be so factored mod n using Berlekamp

3.3.2 Extraction

The mock residue symbol $\langle \frac{\alpha}{Q} \rangle_p$ is defined as :

$\langle \frac{\alpha}{Q} \rangle_p = \zeta_p^i \equiv \alpha^{\frac{NQ-1}{p}}$ mod Q if such a congruence holds

$\langle \frac{\alpha}{Q} \rangle_p$= 0 otherwise

For each initial prime p and each Euclidean prime q with $p|q-1$, compute the mock residue symbols $\langle \frac{J_p(q)}{Q_i} \rangle_p$ for i=1..g for ideals Q_i found during factorisation of the previous section.. If any of the Mock Residue Symbols is 0, declare n to be COMPOSITE.

For each initial prime p, if the mock residue symbols for p are not all equal to 1, choose some nontrivial one, $\langle \gamma/Q)_p$, and make it the distinguished symbol corresponding to the prime p. If they are all 1, we compute mock residue symbols $\langle \gamma/Q_i \rangle_p$ for γ's chosen at random in $Z[\zeta_p]$, until one is found to be not 0 or 1 and designate it the distinguished symbol.
(The algorithm may diverge here, but if n is prime the probability of an arbitrarily chosen γ working is roughly (p - 1)/p, so a very high probability)
For each pair p, q with $p|(q-1)$, compute the exponents $m_{i,q}$ such that
$\langle \frac{\gamma}{Q} \rangle_p^{m_{i,q}} = \langle \frac{J_p(q)}{Q_i} \rangle_p$

Lemma 3.3. *Extraction Lemma*[*APR83*]

Let Q and Q_1 be ideals of $Z[\zeta_p]$ such that $p \not| NQ = NQ_1$ and let R, R_1 be conjugate prime ideals dividing Q and Q_1 respectively.
Suppose there is some $\gamma \in Z[\zeta_p]$ such that $\langle \frac{\gamma}{Q} \rangle_p \neq 0$ or 1
Then $\langle \frac{\gamma}{Q} \rangle_p^m = \langle \frac{\alpha_1}{Q_1} \rangle_p$ ..... (A)
implies $(\frac{\gamma}{R})_p^m = (\frac{\alpha_1}{R_1})_p$ ..... (B)

Proof : Let valuations $v_p(t)$ is defined as the largest integer k such that $p^k|t$.
Now the initial hypothesis is that $\gamma^{\frac{NQ-1}{p}} \equiv \zeta_p^j \bmod Q, j \neq 0 \bmod p$
Reducing this to mod R we have, $\gamma^{\frac{NQ-1}{p}} \equiv \zeta_p^j \bmod R, j \neq 0 \bmod p$
So we have $v_p(NR-1) > v_p(\frac{NQ-1}{p})$ as $NR-1$ is the order of $(Z[\zeta_p]/R)^*$
First suppose $R = R_1$. Then reducing the relation (A) mod R and using $NR = NR_1$ we obtain
$\gamma^{\frac{NQ-1}{p}m} \equiv \langle \frac{\gamma}{Q} \rangle_p^m = \langle \frac{\alpha_1}{Q_1} \rangle_p = \alpha^{\frac{NQ-1}{p}} \bmod R$
Let T be a primitive root of R. Then $T^{(mInd(\gamma)-Ind(\alpha)\frac{NQ-1}{p}} = 1 \bmod R$. But order of T is $NR-1$. So $NR-1|(mInd(\gamma)-Ind(\alpha)\frac{NQ-1}{p}$. This implies $p|(mInd(\gamma)-Ind(\alpha)$ from the first equation of the proof.
This in turn means the following is true (we replace Q with R in the previous equation)
$T^{(mInd(\gamma)-Ind(\alpha)\frac{NR-1}{p}} = 1 \bmod R$
which gives $\gamma^{\frac{NR-1}{p}m} \equiv \alpha^{\frac{NR-1}{p}} \bmod R$ which is what we required (this is equation B).
The case in which $R \neq R_1$ can easily be reduced to the above case and hence Extraction lemma is also true in this case.

The Extraction Lemma and the relations among the mock residue symbols allow us to compute many power residue symbols in terms of one unknown one, via the following calculation.
Suppose r is a prime number dividing n and p > 2 be an initial prime. Suppose $\langle \frac{\gamma}{Q} \rangle$ is the distinguished symbol corresponding to p. Knowing this mock residue symbol does not allow us to compute the power residue symbol $(\gamma/(r,Q))_p$, but there are only p possibilities for it.

Now $(\frac{J_p(q)}{r})_p = (\frac{r}{J_p(q)})_p (J_p(q), r)_\lambda = (\frac{r}{J_p(q)})_p$

Now $(\frac{r}{J_p(q)})_p = \prod_{i=1}^{p-1} (\frac{r}{\sigma_u^{-1} Q})_p^{\theta_{a,b}(u)} = \prod_{i=1}^{p-1} \sigma_u^{-1} (\frac{r}{Q})_p^{\theta_{a,b}(u)} == \prod_{i=1}^{p-1} (\frac{r}{Q})_p^{u^{-1}.\theta_{a,b}(u)} = (\frac{r}{Q})_p^{\hat{\theta}_{a,b}(u)}$

Also, $(\frac{J_p(q)}{r})_p = \prod_{i=1}^{g} (\frac{J_p(q)}{r, Q_i})_p = \prod_{i=1}^{g} (\frac{\gamma}{r, Q})_p^{m_{i,q}} = (\frac{\gamma}{r, Q})_p^{\sum_{i=1}^{g} m_{i,q}}$

Now $\hat{\theta}_{a,b}(u)$ is invertible. So $(\frac{r}{Q})_p = (\frac{\gamma}{r, Q})_p^{\hat{\theta}_{a,b}(u)^{-1} \sum_{i=1}^{g} m_{i,q}}$

Thus if $(\frac{\gamma}{r, Q})_p = \zeta_p^k$ then $Ind_q(r) \equiv k.\hat{\theta}_{a,b}(u)^{-1} \sum_{i=1}^{g} m_{i,q}$ mod p

This shows that the value of $(\gamma/(r,Q))_p$ completely determines each $Ind_q(r)$ mod p for every Euclidean prime q with $p|q-1$. [APR83]

3.3.3 Consolidation step

Main Idea for consolidation is that if $p_1, ...p_d$ are the initial primes and $\langle \frac{\gamma_i}{Q_i} \rangle_{p_i}$ are the corresponding distinguished symbols found in Preparation Step, then for each prime factor r of n, there are integers $k_1, .., k_d$ such that $(\frac{\gamma_i}{r_i, Q_i})_{p_i} = \zeta_{p_i i}^k$ for i=1...d
By the Chinese Remainder Theorem there is a single integer k defined modulo $\prod p_i$, such that $(\frac{\gamma_i}{r_i, Q_i})_{p_i} = \zeta_{p_i}^k$ for i=1...d
For each k, $1 \leq$ k $\leq$ f(n) $= \prod p_i$, we assemble and test a possible divisor r= r(k) of n.

Check whether $r(k)|n$. If it does and r(k) $\neq 1$ or n, declare n composite and halt. Otherwise continue with the next value of k. If none of these divide, we declare n to be prime. For if n is composite it must have a prime factor r $\leq n^{\frac{1}{2}}$

3.4 Ideas for Calculating the Power residue symbol efficiently

The idea of calculating the power residue symbol efficiently was introduced in [dBP17]. We briefly describe the algorithm that they introduced in which $(\frac{\alpha}{b})_m$ is calculated in 3 stages:

- Principalization- Reducing computation of $(\frac{\alpha}{b})_m$ to $(\frac{\alpha}{\beta})_m$ for some $\beta \in b$
- Reduction(Optional)- Reduce computation of $(\frac{\alpha}{\beta})_m$ for large $\alpha, \beta \in K$ to $(\frac{\alpha_i}{\beta_i})_m$ fo smaller $\alpha_i, \beta_i \in K$
- Evaluation-Computing $(\frac{\alpha_i}{\beta_i})_m$ directly using prime density results

3.4.1 Principalisation Algorithm:

The main idea used here is to use random sampling relatively small elements to get a B-prime ideal which can be efficiently factorized. Then using this previous factorization and the reciprocity of residue symbols, we can reduce the power residue computation to a simpler power residue computation.

3.4.2 Evaluation Algorithm:

Power Residue(α_o, β)
Note:$(\frac{\alpha_i \gamma^m}{\beta})_m = (\frac{\alpha_i}{\beta})_m$ This property is used here
The idea behind evaluation is to repeatedly choose a random γ and set $\alpha = \alpha_o \gamma^m$ until $\alpha \mod \beta$ has a small representative $\hat{\alpha}$ that generates a B near-prime ideal. We then use reciprocity to reduce $(\frac{\alpha_0}{\beta})_m$ to calculation of $U(\alpha_o, \beta)$ and $(\frac{\beta}{\alpha})_m$ (which can be done by the factorisation of B-near prime ideals).

3.4.3 Reduction

- This is optional. Usually done when in $(\frac{\alpha}{\beta})_m$ either one(case A) or both (case B)of α, β are large.
In case A, assume α is much larger than β. Here compute LLL-reduced basis M_β of ideal (β) and reduce $\alpha \mod M_\beta$. The reduced α is hoped to have similar size as β.
In case B, both inputs are large, say with bitsize g(n)>6n where n=[K:Q]. Consider the lattice L=$\{(\gamma_1, \gamma_2) \in O_K$ x $O_K | \gamma_1 a - \gamma_2 \in (\beta)\ \}$. This lattice has discriminant $N(\beta)$ and dimension 2n. Applying LLL, we get a short vector $(\gamma_1, \gamma_2) \in O_K^2$with $\sqrt{||\gamma_1||^2 + ||\gamma_2||^2} \leq 2^n N(\beta)^{\frac{1}{2n}} \approx 2^n ||\beta||^{\frac{1}{2}} \approx 2^{\frac{g(n)+2n}{3}} \leq 2^{\frac{2g(n)}{3}} \approx ||\beta||^{\frac{2}{3}}$
where it is assumed heuristically that $||\beta|| \approx N(\beta)^{\frac{1}{n}}$ and $||\beta|| \approx 2^{g(n)}$

Now $\gamma_1 \alpha = k\beta + \gamma_2$ So we can write $(\frac{\gamma_1}{\beta})_m (\frac{\alpha}{\beta})_m = (\frac{\gamma_2}{\beta})_m$
Or $(\frac{\alpha}{\beta})_m = (\frac{\gamma_2}{\beta})_m (\frac{\gamma_1}{\beta})_m^{-1}$.
So computation of power residue symbol is reduced to computation of 2 power residue symbols with smaller imput. This process can be some repeatedly to reduce the size of input forming a recursion tree.

This gives us an efficient method of calculating the power residue symbols.

Chapter 4

Integer Factoring-New Ideas

4.1 Extension of APR ideas for factoring integers

We know from the paper that transition data is obtained by using some "mock residue symbols" (which are related to p^{th} power residue symbols) and "Extraction lemma". We tried to replace the use of "mock residue" symbols by actual residue symbols which can be computed using the ideas of the previous section. But we saw that the particular form of mock residue symbols is essential for the extraction lemma to work which wont be possible with actual residue symbols. So the "Extraction lemma" in its current form can't be used with actual residue symbols. So we tried some modifications to the extraction lemma but they were not efficient.

Now the APR algorithm works on the following idea:
if our number (n=rr' for prime r, r') was passing all pseudo primality tests, it gives an explicit factor of n (ie, r or r'). Passing the pseudo primality tests would depend on the initial primes we take. So we try to select a "good" set of initial primes such that the pseudo-primality tests are passed .

Another thing we claim from the extraction lemma is that in APR algorithm, with high probability, none of the mock residue symbols would be 0 (and hence the algorithms tries to find an explicit factor of n=rr') if the behaviour of r and r is indistinguishable in terms of the mock-residue symbols. So our claim was that if $ord_p(r)|ord_p(n)$ and $ord_p(r')|ord_p(n)$ for all the initial primes p, we would be able to

factorise n. Basically, if n=rr' (for distinct primes r,r') then we clain that factoring would succeed as long as their algorithm steps are "identical mod r and mod r' ".

So our idea is we start with lots of initial primes p, and form lots of Euclidean primes q and find a "good" set of primes such that the behaviour of APR mod r and mod r' is same, and then run the APR-algo for this set which should factorize n. But doing this using brute force would lead to a huge time complexity which would make our algorithm inefficient. Thus we need to think of a clever way of selecting the initial primes.

4.2 Factoring n=pq using ideas of Automorphisms in Cyclotomic Fields

Let $R = (\mathbb{Z}/n\mathbb{Z})$. By Chinese remaindering, we see that $R \cong (\mathbb{Z}/p\mathbb{Z})$ x $(\mathbb{Z}/q\mathbb{Z})$. Now as p and q are primes, we see that $(\mathbb{Z}/p\mathbb{Z})$ and $(\mathbb{Z}/q\mathbb{Z})$ are fields. Now we want cyclotomic extensions of these fields. So we work in $R/(\phi_r(x))$ where $\phi_r(x) = x^{r-1} + x^{r-2}...x + 1$ for some prime r. Again by Chinese remaindering, we see that $R/(\phi_r(x)) \cong (\mathbb{Z}/p\mathbb{Z})/(\phi_r(x))$ x $(\mathbb{Z}/q\mathbb{Z})/(\phi_r(x))$. Thus we see that the ring in which we work can be seen as the product of two cyclotomic extensions of a field. $(\mathbb{Z}/p\mathbb{Z})/(\phi_r(x))$ can also be represented by $F_p(\zeta_r)$. The intuitive reason for this is that in this cyclotomic extension of F_p, powers of x are the r-th roots of the polynomial $x^r - 1$. Hence x can be thought of as ζ_r, ie, the r-th primitive root of unity.

First we see what kind of automorphisms can be present in a cyclotomic extension:

$\sigma \in Aut(F_p(zeta_r))$ iff $\sigma : \zeta_r \mapsto \zeta_r^i$ for some i coprime to r. The reason for this is that any automorphism σ can only map ζ_r to another root. (This can also be seen using the Frobenius automorphism. Any automorphism $\sigma : \zeta_r \mapsto \zeta_r^{p^i}$ which gives the same set of automorphisms as above). This gives that the only automorphisms possible in this cyclotomic extensions are $x \mapsto x^i$

Claim- If we get any map ϕ in $(\mathbb{Z}/n\mathbb{Z})/(\phi_r(x))$ that is an automorphism in one of the decomposed cyclotomic fields/rings but not in the other , then we will be able to factor $n = pq$

The reason is that since ϕ is an automorphism in only one of the cyclotomic extension (WLOG, say $(\mathbb{Z}/p\mathbb{Z})/(\phi_r(x))$) but not in the other field, we see that:

$\phi(ab) - \phi(a)\phi(b)$ would be zero mod p but not not q.

So we can take gcd(n,$\phi(ab) - \phi(a)\phi(b)$) to get p and hence factorise n=pq.

Also we see that any such map ϕ would be different from $x \mapsto x^i$. So our aim is to find a map ϕ different from $x \mapsto x^i$ that is an automorphism in the decomposed field. (We can also see that if such a map exists using counting argument. Also if we are able to find such a map, another way to factor n would be to take the gcd of the coefficients with n, as atleast some coefficient would have a non-trivial gcd with n in this case).

We tried to use interpolation ideas to set up equations to find such an automorphism (different from x^i). Idea- set-up eqns like $a(x) = x^2, a(x^6) = x^4$ and their swaps and then get the resultant polynomial using interpolation and see if it behaves as an automorphism. But on checking carefully, we saw that the interpolation idea in univariate field extension is problematic, as you don't have access to ζ. We only have 'x' and even though x can be thought of as the r-th root of unity, it behaviour is different from that of ζ.

For eg., suppose we have r=7 such that we have $\zeta^7 = 1$ and we set up an equation $a(x) = x^2, a(x^6) = x^4$. Then $a(\zeta) = \zeta^2 \Rightarrow a(\zeta^6) = \zeta^12 = \zeta^5 \neq \zeta^4$. so the two eqns are inconsistent. So our goal is that somehow we need BOTH ζ and x to play this strategy. We think that bivariate extension of the field should help in this case and we are currently exploring these.

Using python code, we also tried to see the pattern that the Frobenius morphism gives in the decomposed field such that we are able to find an algorithm for interpolation. We took particular value of p. I took p=13 = -1 mod 7 so that we get $(F_p^2)^3.Here(x^7 - 1)/(x - 1) = (x^2 + 3x + 1)(x^2 + 5x + 1)(x^2 + 6x + 1)$.

Objective: To find an a(x) other than x^i which gives an automorphism

A pattern we noticed was this: For $a(x) = x^i$ in the original ring, $x \mapsto x^i$ in the decomposed fields too but it also can introduce swaps.

In simple words, say I get an element e=$(3x, x, 0)$ (these components are using Chinese remaindering). Then with $a(x) = x^2$, e maps to $(0, 3x^2, x^2)$ and with a$(x) = x^3$, e maps to $(x^3, 0, 3x^3)$ and so on. So we think that to get a $\phi(x)$other than x^i, we need to find a map such that (x, x, x) maps to say (x, x^2, x) Instead of say (x^2, x^2, x^2) or something similar. This gives us insight into what kind of automorphisms in the decomposed field can help.

4.3 Factoring n=pq using Gauss Quadratic sums

We had the following idea for factorizing n=pq using Gauss sums. We know that for an odd number p $g(1;p)^2 = \left(\frac{-1}{p}\right) p$. The idea is to pick a random prime s. Define $r = s^2$ such that ϕ_r is irreducible mod p and mod q. We know that with high probability this would be true.
Also $s^2 = 1 \mod 4$, so $r = 1 \mod 4$. Now we compute the gauss sum g(1;r) in the ring $R = (Z/nZ)(\zeta_r) = (Z/nZ)[x]/(\phi_r(x))$ where ϕ_r is the r-th cyclotomic polynomial. So $g(1;r) = \sqrt{(s^2)} = s$ or $-s$. Also with high probability, we can assume that p and q have different quadratic residuosites mod r. WLOG assume that (p/r)=1 and (q/r)=-1. Then we compute the gcd(g-s,n). Then we know that as $g = \sqrt{r}$, we claimed that with high probability $g = s \mod p$ but $g \neq s \mod q$ due to our quadratic residuosity assumptions. So the gcd (g-s, n) should give us a non-trivial factor of n=pq. ANother way would be see the coefficients of g to get a factor

So our algorithm was as follows:

- Input n=p.q. Randomly pick prime r =1 mod 4. [r= polylog(n)]
- By Gauss theorem: $g^2 = r$. Here g is the Gauss Quadratic sum.
- With high probability (r/p)=(p/r)=1 AND (r/q)=(q/r)=-1.
- By (1) and cyclotomic-ring: $g(x)^2 = r \mod \langle p, \phi_r(x)\rangle$. By (2) : $g(x) \in F_p$.
- By (1) and cyclotomic-ring: $g(x)^2 = r mod \langle q, \phi_r(x)\rangle$. By (2): $g(x) \notin F_q$.

- By (3)-(4): $g(x) \mod \langle n, \phi_r(x) \rangle$ written in reduced form, has some coefficient that factors n, by gcd.

We wrote a code for this and saw that the algorithm was not giving the gcd correctly and hence not factoring n. The reason we saw for this is that $g(x) \in F_p$ was not coming out to be true. It was not giving a number. The reason for this was as follows, we know that as (r/p)=(p/r)=1, $\phi_r(x) \mod p$ has atleast 2 factors (say $f_1 \dots f_k$) as the number of factors are even. Then for $g(x)$ to be a number $\mod \langle p, \phi_r(x) \rangle$, we would need $g(x) \mod f_i$ to all give the same sign (ie, the Gauss sum should give the same sign in the decomposed field's). But we saw that the signs were not all same. Instead half of these gave + signs and half of these gave - signs which led to g not being a number $\mod \langle p, \phi_r(x) \rangle$. This creates a problem so that our algorithm cannot factorise n=pq.

Chapter 5

Polynomial Factoring Ideas-Previous Work

We first present previous work done in this and our efforts to extend it. We present a report of[DMS19]. The main idea in this paper for factorization modulo p^k for some prime k is by using ROOT-FIND algorithm introduced in [JBQ13] and further modified in [DMS19]. For the polynomial factoring into coprime factors modulo p it can be easily lifted using Hensel Lifting [BS66] to modulo p^k for any k. So we consider only the hard case when $f(x) \mod p$ is a perfect power of some irreducible, i.e. $f(x) \equiv \phi^e \mod p$ for an irreducible polynomial $\phi(x) \in \mathbb{Z}[x]$. The given method works on factorizing it in the ring $(\mathbb{Z}/p^k\mathbb{Z})[x]$ by reducing it to root-finding of a bivariate polynomial $E(y)$, with coefficients of y in $\mathbb{Z}[x]$, in the ring $(\mathbb{Z}[x]/\langle p^k, \phi^{ak}\rangle)[y]$ for some $a \leq e$.

Theorem 5.1. [*DMS19*] *Given a prime p, $k \leq 4$ and a univariate polynomial $f(x)$ with integral coefficients. Then $f(x) \mod p^k$ can be efficiently factored and tested for irreducibility in randomized poly$(deg(f), \log p)$ time.*

Theorem 5.2. [*DMS19*] *Given a prime p, an integer $k \leq 4$ and a univariate polynomial $f(x)$ which is the power of an irreducible modulo p and g is a factor of f mod p, we can count all the factors $\tilde{g}$ (which are lifts of g) of $f \mod p^k$ (or decide that it is irreducible) and represent them in a compact datastructure.*

Proof idea for Theorem 5.1: We only consider the hard case, ie, assume that $f(x)$, of degree d satisfies $f(x) \equiv \phi(x)^e \mod p$ for some irreducible integral polynomial ϕ and $\deg(\phi)e \leq d$. Otherwise we can easily use Hensel's lifting to find a factorization. So we have f of the form $\phi^e + pf'$ for some polynomial $f'(x) \in (\mathbb{Z}/\langle p^k \rangle)[x]$. The factors of f will be of the form $\phi^a - py$ for $y \in (\mathbb{Z}/\langle p^k \rangle)[x]$. The idea of this approach is so divide $f(x)$ by $h(x)$ and binomially expand $\frac{1}{1-x} = 1 + x + x^2 + x^3 + \dots$ and using this, we can reduce the problem of factorization to that of root finding in bigenerated ideal which in the case of p^4 can be efficiently found.

Proof idea for Theorem 5.2: We want to find, for each $a < e$, polynomials of the form $\tilde{g}(x) \equiv \phi^a \mod p$ such that it lifts to modulo p^k and divides f in $\mathbb{Z}_{p^k}$. For a fixed a we can find all y's modulo p^{k-1} from the proof of theorem 1, which can be obtained from the modification to the root find algorithm. Then we can sum over all the a's for which it lifts.

5.1 Main Results

5.1.1 Reduction of Factoring to Root-Finding

In this subsection, we describe a polynomial $E(y)$ which will help in finding a factor of $f \equiv \phi^e + pl \mod p^k$ which is $h \equiv \phi^a - py \mod p^k$ where $a \leq e/2$. This is true as if g is a polynomial such that $f \equiv gh \mod p^k$, then looking modulo p we will have exactly one of them contributing to the power of ϕ less than or equal to $e/2$. We choose this factor as h.
Throughout the rest of this report we denote $\mathbb{Z}[x]/\langle p^k, \phi^{ak} \rangle$ as R and $\mathbb{Z}[x]/\langle p, \phi^{ak} \rangle$ as R_0. Now we define the polynomial $E(y) \in R[y]$ as

$$E(y) = f(x)(\phi^{a(k-1)} + \phi^{a(k-2)}py + \phi^{a(k-3)}(py)^2 + \cdots + \phi(py)^{k-2} + (py)^{k-1}) \quad (5.1)$$

We now reduce the factoring $f(x)$ modulo p^k to root-finding of $E(y)$ modulo the bi-generated ideal $\langle p^k, \phi^{ak} \rangle$.

Theorem 5.3. *Reduction Theorem[DMS19]*

Given two polynomials $f(x) \equiv \phi^e + pl \mod p^k$ *and* $h(x) \equiv \phi^a - py \mod p^k$ *for* $a \leq e$*, we will have* $h(x)$ *as a factor of* $f(x)$ *if and only if* $E(y) \equiv 0 \mod \langle p^k, \phi^{ak} \rangle$ *(i.e.* y *is a root of E(y) in R), for notations as defined above.*

Now if we denote $y = y_0 + py_1 + p^2y_2 + \dots p^{k-1}y_{k-1}$, the next lemma reduces the precision of y.

Lemma 5.4. [*DMS19*] *If* $y = y_0 + py_1 + p^2y_2 + \dots p^{k-1}y_{k-1}$*, then every element of the set* $y_0 + py_1 + \dots p^{k-3}y_{k-3} + p^{k-2}*$ *is a root of* $E(y)$ *modulo* $\langle p^k, \phi^{ak} \rangle$*.*
Proof: Notice that in the expansion of $E(y)$ *all the y's that are present are multiplied by p, which implies* y_{k-1} *will have coefficient divisible by* p^k*, which is* 0 *in R. Also for* y_{k-2}*, all the terms of the form* $(py)^i$ *for* $i \geq 2$ *do not contribute as they are zero in R. The only remaining term is* $f(x)\phi^{a(k-2)}py$*. Now* $f(x)\phi^{a(k-2)} \equiv \phi^{e+ak-2a} \mod \langle p, \phi^{ak} \rangle$ *which also vanishes modulo* $\langle p^k, \phi^{ak} \rangle$*, as* $e \geq 2a$*.*

Next we can have this following modification of the root-find algorithm discovered in[JBQ13]. This modification in due to [DMS19]

Theorem 5.5. [*DMS19*] *Given a bivariate polynomial* $g(y) \in R_0[y]$*, let* $Z \subset R_0$ *be the root set of* $g(y)$*. Then* Z *can be written as the disjoint union of atmost* $deg_y(g(y))$ *many representative pairs* (a_0, i_0)*,* $a_0 \in R_0, i_0 \in R_0$*. These representative roots can be found in randomized poly- time.*

For the next subsection and the rest of the report we fix $k = 4$. Similar techniques can be applied for $k = 2, 3$ as well.

5.1.2 Reduction to Root-finding modulo $F_p[x]/\langle\phi^{ak}\rangle$

In this subsection we reduce root finding modulo $\langle p^4, \phi^{4a}\rangle$ to root finding in $R_0 = \mathbb{F}_p[x]/\langle\phi^{4a}\rangle$. We consider $E(y) \equiv 0 \mod \langle p^4, \phi^{4a}\rangle$ as

$$f \times (\phi^{3a} + \phi^{2a}py + \phi^a(py)^2 + (py)^3) \equiv 0 \mod \langle p^4, \phi^{4a}\rangle$$

Putting $y = y_0 + py_1$, as it only depends on these two coordinates, we obtain the equation

$$f \times (\phi^{3a} + \phi^{2a}p(y_0 + py_1) + \phi^a p^2(y_0^2 + 2py_0y_1) + (py)^3) \equiv 0 \mod \langle p^4, \phi^{4a}\rangle$$

We now attempt to solve this modulo $\langle p^3, \phi^{4a}\rangle$. Now y_1 is is redundant modulo this ideal by considering $f \equiv \phi^e \mod p$ and using the fact that $e \geq 2a$. So we will have some representative roots of y_0 of the form $a_0 + \phi^{i_0}*$, but only for some selected values of $*$ the lifting will be possible. So we intend to find y_0's such that $a_0 + \phi^{i_0}y_0 + py_1$ are roots of E modulo $\langle p^3, \phi^{4a}\rangle$. The next lemma establishes this.

Lemma 5.6. *(Reduction to Characteristic p)[DMS19] We can efficiently find a unique set S_0 of representative pairs (a_0, i_0), $a_0 \in R_0, i_0 \in \mathbb{N}$ such that*

$$E((a_0 + \phi^{i_0}y_0) + py_1) = p^3E'(y_0, y_1) \mod \langle p^4, \phi^{4a}\rangle$$

for $E'(y_0, y_1) \in R_0[y_0, y_1]$ depending on the representative root pair. Also we will have:

1. *$|S_0| \leq 2$. If the algorithm fails to find any such E' then $E(y) \equiv 0 \mod \langle p^4, \phi^{4a}\rangle$ has no solutions*
2. *$E'(y_0, y_1) = E_1(y_0) + E_2(y_0)y_1$ where $E_1(y_0), E_2(y_0) \in R_0[y_0]$, E_1 is a cubic in y_0 and E_2 is a linear in y_0*
3. *For every root $y \in R$ of $E(y) \equiv 0 \mod \langle p^4, \phi^{4a}\rangle$, $\exists(a_0, i_0) \in S_0$, and $(a_1, a_2) \in R \times R$ such that $y = a_0 + \phi^{i_0}a_1 + pa_2$ and $E'(a_1, a_2) \equiv 0 \mod \langle p, \phi^{4a}\rangle$*

5.1.3 Finding roots of $E'(y_0, y_1)$ modulo $\langle p, \phi^{4a} \rangle$

Now we are given with a bivariate polynomial $E(y_0, y_1)$ modulo $\langle \phi^{4a} \rangle$ in $\mathbb{F}_p[x]$ which is of the form $E_1(y_0) + E_1(y_0)y_1$. The idea is to go over all r, $0 \leq r \leq 4a$ and find y_0 such that

- $E_1(y_0)$ has valuation over ϕ greater that or equal to r
- $E_2(y_0)$ has valuation over ϕ exactly r

The next lemma shows that this contains all the solutions.

Lemma 5.7. [*DMS19*] *A pair $(u_0, u_1) \in R \times R$ satisfies an equation of the form $E_1(y_0) + E_2(y_0)y_1 \equiv 0 \mod \langle p, \phi^{4a} \rangle$ if and only if $val_\phi(E_1(u_0)) \geq val_\phi(E_2(u_0))$.*
Proof: Let $r = val_\phi$, $r \in \{0, 1, \ldots 4a\}$. Then we will have $u_1 \equiv -\frac{E_1(u_0)/\phi^r}{E_2(u_0)/\phi^r} \mod \langle p, \phi^{4a-r} \rangle$. Conversely if $r' = val_\phi(E_1(u_0)) < val_\phi(E_2(u_0)) \leq 4a$ then for every u_1 we will have $val_\phi(E_1(u_0) + u_1 E_2(u_0)) < 4a$ which means this cannot be zero modulo $\langle p, \phi^{4a} \rangle$.

Hence we can efficiently find all representative pairs for y_0, at most 3, such that $E_1(y_0)$ has valuation over ϕ as exactly r. The next lemma shows that we can filter these y_0's such that $E_2(y_0)$ has valuation exactly r over ϕ.

Lemma 5.8. *x Given a linear polynomial $E_2(y_0) \in R_0[y_0]$ and an $r \in \{0, 1 \ldots 4r - 1\}$, let (b, i) be a representative root pair modulo $\langle p, \phi^r \rangle$. Considering the quotient polynomial $E'(y_0) = E_2(b + \phi^i y_0)/\phi^r$, if $E_2(y_0)$ does not vanish identically modulo $\langle p, \phi \rangle$, then there is exactly one $\theta \in R_0/\langle \phi \rangle$ such that $E_2(\theta) \equiv 0 \mod \langle p, \phi \rangle$, and this θ can be efficiently computed.*

Proof: We assume $E_2(y_0)$ is of the form $u + vy_0$. Now since $u + vy \equiv 0 \mod \langle p, \phi^r \rangle$ and y_0 is a variable, we will have $\text{val}_\phi(u) \geq r$ and $\text{val}_\phi(v) \geq r$. Considering cases when

- $\text{val}_\phi(u) \geq r$ and $\text{val}_\phi(v) = r$, we will have $E_2(\theta) \equiv 0 \mod \langle p, \phi \rangle$ only when $\theta = \frac{-(u/\phi^r)}{(v/\phi^r)} \mod \langle p, \phi \rangle$

- $\text{val}_\phi(u) = r$ and $\text{val}_\phi(v) > r$, $E_2(\theta)$ will not be zero modulo $\langle p, \phi \rangle$ for any θ in $R_0/\langle \phi \rangle$
- $\text{val}_\phi(u) > r$ and $\text{val}_\phi(v) > r$, then we will have some greater value of r.

Thus in this way θ can be efficiently computed.

5.1.4 Algorithm to find Roots of $E(y)$

With these aforementioned lemmas and theorems we present the algorithm to find roots of $E(y)$ modulo $\langle p^4, \phi^{4a} \rangle$ of $\mathbb{Z}[x]$[DMS19]

Algorithm 1 Finding all roots of $E(y)$ in R

Given $E(y_0 + py_1)$, using Lemma 11, get the set S_0 of all representative pairs (a_0, i_0), where $a_0 \in R_0$ and $i_0 \in \mathbb{N}$, such that $p^3 | E((a_0 + \varphi^{i_0} y_0) + py_1) \bmod \langle p^4, \varphi^{4a} \rangle$.
Initialize sets $Z = \{\}$ and $Z' = \{\}$; seen as subsets of R_0.
for each $(a_0, i_0) \in S_0$ **do**
    Substitute $y_0 \mapsto a_0 + \varphi^{i_0} y_0$, let $E'(y_0, y_1) = E_1(y_0) + E_2(y_0) y_1 \bmod \langle p, \varphi^{4a} \rangle$ be the polynomial obtained from Lemma 11.
    If $E_2(y_0) \not\equiv 0 \bmod \langle p, \varphi \rangle$ **then** find (at most one) $\theta \in R_0/\langle \varphi \rangle$ such that $E_2(\theta) \equiv 0 \bmod \langle p, \varphi \rangle$. Update $Z \leftarrow Z \cup (a_0 + \varphi^{i_0} *)$ and $Z' \leftarrow Z' \cup (a_0 + \varphi^{i_0}(\theta + \varphi *))$.
    for each possible valuation $r \in [4a]$ **do**
        Initialize sets $Z_r = \{\}$ and $Z_r' = \{\}$.
        Call ROOT-FIND(E_1, φ^r) to get a set S_1 of representative pairs (a_1, i_1) where $a_1 \in R_0$ and $i_1 \in \mathbb{N}$ such that $E_1(a_1 + \varphi^{i_1} y_0) \equiv 0 \bmod \langle p, \varphi^r \rangle$.
        for each $(a_1, i_1) \in S_1$ **do**
            Analogously consider $E_2'(y_0) := E_2(a_1 + \varphi^{i_1} y_0) \bmod \langle p, \varphi^{4a} \rangle$.
            Call ROOT-FIND(E_2', φ^r) to get a representative pair (a_2, i_2) ($\because E_2'$ is linear), where $a_2 \in R_0$ and $i_2 \in \mathbb{N}$ such that $E_2'(a_2 + \varphi^{i_2} y_0) \equiv 0 \bmod \langle p, \varphi^r \rangle$.
            if $r = 4a$ **then**
                Update $Z_r \leftarrow Z_r \cup (a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2} *))$ and $Z_r' \leftarrow Z_r' \cup \{\}$.
            else if $E_2'(a_2 + \varphi^{i_2} y_0) \not\equiv 0 \bmod \langle p, \varphi^{r+1} \rangle$ **then**
                Get a $\theta \in R_0/\langle \varphi \rangle$ (Lemma 13), if it exists, such that $E_2'(a_2 + \varphi^{i_2}(\theta + \varphi y_0)) \equiv 0 \bmod \langle p, \varphi^{r+1} \rangle$. Update $Z_r' \leftarrow Z_r' \cup (a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2}(\theta + \varphi *)))$.
                Update $Z_r \leftarrow Z_r \cup (a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2} *))$.
            end if
        end for
        Update $Z \leftarrow Z \cup (a_0 + \varphi^{i_0} Z_r)$ and $Z' \leftarrow Z' \cup (a_0 + \varphi^{i_0} Z_r')$.
    end for
end for
Return Z and Z'.

Theorem 5.9. [*DMS19*] *The output of Algorithm 1 (set Z - Z_0) contains exactly those $y_0 \in R_0$ for which there exist some $y_1 \in R_0$, such that, $y = y_0 + py_1$ is a root*

of E(y) in R. We can easily compute the set of y_1 *corresponding to a given* $y_0 \in Z$ *-* Z_0 *in poly(deg f, log p) time. Thus, we efficiently describe (and exactly count) the roots* $y = y_0 + py_1 + p^2y_2$ *in R of E(y), where* $y_0, y_1 \in R_0$ *are as above and* y_2 *can assume any value from R.*

5.2 Wrapping up the theorems

5.2.1 Theorem 5.1

The only non trivial case is when f is a power of an irreducible mod p. In this case, we can assume $\phi^a - py$ is a factor of $f(x)$ and use Algo 1 to check if this is possible or not. We do this for all $a \leq \frac{e}{2}$ and hence is a polytime algorithm.

5.2.2 Theorem 5.2

As g is a factor of f mod p, we know $g = \tilde{v}\phi(x)^a$ where $\tilde{v}$ is a unit. Any lift of this to mod p^4 is of the form $\tilde{g}(x) = v(x)(\phi(x)^a - py) \mod p^4$ for a unit $v(x) \in (\tilde{v} + p*) \subset (Z/\langle p^4\rangle)[x]$. Any such $\tilde{g}(x)$ maps uniquely to $g_1(x) = v(x)^{-1}\tilde{g} = (\phi(x)^a - py) \mod p^4$. So we only need to consider lifts of $\phi(x)^a \mod p$
Now any lift of $\phi(x)^a \mod p$ which is a factor of f is of the form $\phi(x)^a - py(x) \mod p^4$ for a polynomial $y(x) \in (Z/\langle p^4\rangle)[x]$. From the previous discussion, this can be done in poly time using root finding in a different ring (using previous reductions) for all $a \leq e/2$
For $a > e/2$ replace $b := e - a \leq e/2$ which can be solved as before.

Chapter 6

Polynomial Factoring-New Ideas

We describe some efforts made by us during this project to extend the existing results for factorising and the problems that we faced.

6.1 Using Multivariate Representative Roots

Claim: If we can generate representative roots mod $\langle x^l, p^k \rangle$, then we could inductively extend this process inductively from k to k+1.

Proof: Our aim is to find roots mod $\langle x^l, p^k \rangle$ of a given polynomial g . We know we can find representative roots mod $\langle x^l, p \rangle$ from before. Let the representative root that we get be r. (we treat this representative root as a multivariate)

So we can write $g(r) = \alpha.x^l + \beta.p$ Here α, β are polynomials in multiple variables. Now we want this to be 0 mod $\langle x^l, p^k \rangle$. So this reduces to finding roots of β mod $\langle x^l, p^{k-1} \rangle$

This method works well for the extending k=1 to k=2. But for higher k's (say k=3), we would need representative roots of multivariates mod $\langle x^l, p \rangle$. Now finding a single root of such an equation over multivariate is possible but there is no such known method for finding multivariate representative roots in the previous case. So this is where this idea fails.

For eg. an elliptic curve $y^2 = x^3 + 1 \mod p$, we can easily find a "root"... but no efficient algorithm (or even representation is known for multivariate case. In fact in this case, it seems that there are just way too many isolated roots to list (each root roots seems its own representative).

Our aim is to find an efficient representation and algorithm for finding representative roots in the multivariate case.

Another thing that we explored was trying to reduce powers of x in the previous idea as compared to powers of p.

6.2 Relation to HN

We tried to extend the previous idea of treating the representative roots as multivariates to the following. We tried to think of representative root as:

Representative root $r := \sum_{i,j} u_{ij}.x^i.p^j$; where u_{ij} belong to the field F_p. Now we to tried set $g(r) = 0 \mod \langle p^k, x^l \rangle$ to get a variety in constant-vars. This we thought could be solved using HN ideas.

So we considered the following algorithm:

- We need $(\phi^a - pY)$ dividing f(x) mod $\langle p^k, \phi(x)^l \rangle$; where $(f = \phi^e \mod p)$.
- Using this reduction we first find E(Y) (as done previously using binomial expansion); then we directly get the variety from: $E(\sum_{ij} Y_{ij}.p^i.\phi(x)^j) = 0$ mod $\langle p^k, \phi(x)^l \rangle$. (the variety is obtained by comparing the coefficients of powers of x, the RHS has all of them as 0)
- This variety is in vars Y_{ij}. You want these roots in the finite field $F_p[z]/\langle phi(z) \rangle$.
- Use Hilbert's Nullstellensatz related ideal computations to test whether this ideal is empty.

However we noticed that HN reduction is faulty. For eg., from $a_0 + p.a_1 = 0 \mod p^2$ we can deduce: $a_0 = 0 \mod p$ and $a_0/p + a_1 = 0 \mod p$. But we can't say: $a_1 = 0$ mod p. This stops simple reduction from HN mod p^k to HN mod p. Hence factoring

deg=4 polynomial f(x) mod p^2 using this idea of reduction to HN in poly(log p)-time is open.

In simple terms, the problem is that when we try to reduce the HN mod p^k to HN mod p , we do not get a single instance of HN mod p but a large (exponential) number of possibile instances for HN mod p which can't be solved efficiently. We are considering ideas from algebraic geometry to solve these.

So we see that if we are able to solve HN mod p^k efficiently, we will be able to factor polynomials mod p^k efficiently for any k which is our goal (and an open question right now).

Bibliography

[APR83] Leonard M. Adleman, Carl Pomerance, and S. Robert. On distinguishing prime numbers from composite numbers. *Annals of Mathematics*, 1983.

[Ber68] Elwyn R Berlekamp. Factoring polynomials over finite fields. *Algebraic Coding Theory.*, 1968.

[dBP17] Koen de Boer and Carlo Pagano. Calculating the power residue symbol and ibeta. *ISSAC*, 2017.

[DMS19] Ashish Dwivedi, Rajat Mittal, and Nitin Saxena. Efficiently factoring polynomials modulo p^4. *ISSAC*, 2019.

[JBQ13] Gregoire Lecerf Jeremy Berthomieu and Guillaume Quintin. Polynomial root finding over local rings and application to error correcting codes. *Applicable Algebra in Engineering, Communication and Computing,*, 2013.