Polynomials over Composites

Compact Root Representation via Ideals and Algorithmic Consequences

A Thesis Submitted

in Partial Fulfillment of the Requirements

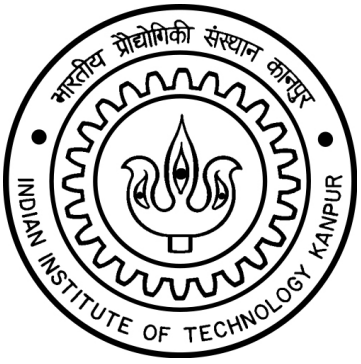
for the Degree of

Doctor of Philosophy

by

Ashish Dwivedi

17111261



to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

January, 2023

CERTIFICATE

It is certified that the work contained in the thesis entitled “Polynomials over Composites: Compact Root Representation via Ideals and Algorithmic Consequences”, by “Ashish Dwivedi”, has been carried out under our supervision and that this work has not been submitted elsewhere for a degree.

Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur

Rajat Mittal
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur

January, 2023

DECLARATION

This is to certify that the thesis titled "Polynomials over Composites: Compact Root Representation via Ideals and Algorithmic Consequences" has been authored by me. It presents the research conducted by me under the supervision of "Prof. Nitin Saxena" and "Prof. Rajat Mittal" at the Department of Computer Science and Engineering, IIT Kanpur. To the best of my knowledge, it is an original work, both in terms of research content and narrative, and has not been submitted elsewhere, in part or in full, for a degree. Further, due credit has been attributed to the relevant state-of-the-art and collaborations (if any) with appropriate citations and acknowledgements, in line with established norms and practices.

Ashish Dwivedi
Programme: Doctor of Philosophy
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur 208016

January, 2023

Synopsis

Polynomials with integer coefficients modulo a composite number N are natural generalization of polynomials modulo a prime p (i.e. Galois field). While the latter have been prominently studied in computer science and mathematics, former is less understood. We attempt to understand them by studying some fundamental problems centered around their factors and roots. These problems reduce to the ring of integers modulo prime power (i.e. Galois ring). In such a ring the polynomial may have 'exponentially' many roots which becomes difficult to handle by currently known methods. We employ 'polynomial ideals' as a tool to 'compactly' represent all the roots and make progress on the following problems.

First, we study the problem of counting roots of a univariate polynomial in a Galois ring. We solve this problem in deterministic polynomial time by constructing special ideals: 'split ideals'. Our algorithm also extends to count special factors, called basic-irreducible factors, in deterministic polynomial time.

We give an application to arithmetic-algebraic geometry. In deterministic polynomial time, we compute rational function form of Igusa's local zeta function (IZF) associated to a given univariate polynomial and a prime p. It is #P-hard to compute IZF for general multivariate polynomials. IZF is a very important generating function which encodes the number of roots of the associated polynomial modulo prime powers. In another application, we give the first deterministic polynomial time algorithm to count roots of a univariate (with multiplicity) over p-adic integers.

Factoring a univariate polynomial modulo a prime power (p^k) is a challenging open problem when k is 'constant'. We give a factoring algorithm for 'constant degree' polynomials in randomized polynomial time. Some of our methods also solve the problem completely (arbi-

trary degree) when k is at most 4; improving the state of the art beyond $k = 2$. Essentially, 'constant degree' factoring was achieved by efficiently reducing it to the next fundamental problem: Solving a system of n-variate polynomial equations modulo a prime-power p^k. Over Galois (finite) fields ($k = 1$) the problem is very well studied and even its decision version, i.e., to test for existence of a solution, is NP-hard for unbounded n. Assuming n constant, efficient randomized algorithm is known over finite fields. However no efficient algorithm is known over Galois rings even if $k = 2$. We generalise the result to Galois rings by solving the problem in randomized polynomial time when k is constant.

Acknowledgements

I have been very fortunate to be advised by Professor Nitin Saxena and Professor Rajat Mittal. I put my first steps in research as a master's student of Prof Nitin in January 2016, knowing nothing about theoretical computer science, and hopefully ended up becoming a decent researcher now. I am grateful to him for being my advisor and imparting all the knowledge through his courses and countless hours of discussions. The time of May 2018 is unforgettable to me when we used to have discussions from around 3 PM to 7 PM every single day. I am grateful for this generosity with time and teaching me a lot of algebra which boosted my confidence in research in the starting years of my PhD. I also thank him for being patient with me and letting me work with my own pace.

I thank Prof Rajat for his simple and intuitive explanations which was very helpful in start of my PhD where we have collaborated in my initial two publications. I have learned quite a bit from him about presentation of ideas in talks or in general. I owe anything good in my presentations to him and the bad aspects are totally mine. His enthusiasm for teaching and lucid explanations reflect in the teaching awards he has received from the institute. I also thank him for guidance, encouragements and showing confidence in me at several occasions.

I also thank both Prof Nitin and Rajat for their immense support during May 2021 when I was hospitalised due to COVID-19. I thank them for contributing and collecting funds, despite my reluctance and ignorance, and continuously monitoring my health progress. I am extremely grateful to all the individuals (known and unknown) who have helped me financially and morally during that hard time. I apologise for not being able to put the long list here.

I would like to thank my teachers at IITK namely, Professors– Nitin Saxena, Surender Baswana, Rajat Mittal, Manindra Agrawal, Raghunath Tewari, Sunil Simon, Subhajit Roy

and Harish Karnik. I am specially thankful to Prof Surender Baswana for personal encouragements and providing an amazing course experience. I thank the staff members at CSE department for their help during my stay at IITK, in particular– Sahu ji, Rajesh ji, Akash ji and Aradhana ji. The results in this thesis are based on joint work with Sayak Chakrabarti and my advisors. I thank them for all the discussions and learning.

I thank Prahladh, Ramprasad, Srikanth and ICTS Bangalore for organising the wonderful workshop WACT19 and letting me present my work before the wide audience there. It was a great learning experience. I had discussions with Ramprasad at various occasions; his enthusiasm is inspiring. I thank Research-I foundation of Infosys to fund my travel to WACT19 and CCC 2019. I am thankful to my friend Chi-Ning Chou for inviting me to present my work at Harvard TGINF, for his hospitality and all the interesting discussions. I also thank Prof Madhu Sudan for giving me his time during my visit to Harvard.

I thank Pranav for remaining a true friend in all the happy and sorrow moments of master's and PhD journey and showing his trust and support in tough times. His clarity of thoughts and managing things even in adverse situations is inspiring. I thank Samik for his friendship and all his help while staying and while leaving IITK. I have shared a lot of memorable moments with both, often hanging out at food outlets. Initial years of my PhD has been my best time sheerly due to my two senior colleagues and friends– Amit and Sumanta. I am grateful to have had them as my mentors and friends. I thank Amit for always imparting positivity and for being available as a go to person for anything. I thank Abhishek Rose for being a great friend since the master's and being part of some thrilling experiences :) I thank my other friends and colleagues at IITK for having wonderful time together: Anindya, Bhargava, Diptajit, Mahesh, Pranjal, Prateek, Priyanka, Rajendra, Utsav and Zeyu.

Many thanks to my childhood and best friend Nitesh. I thank my other friends Vijay, Sudhanshu, Bhavneet and Divyanshu. I thank my little brother Shrestha, little sister Sindhuja, my inspirations Nanaji-Dadaji and my late father for providing me everything I needed and the reason for where I am. I am grateful to all my gurus for always showing me the right path. Finally, the one whom I can't even thank but just acknowledge for my existence and for everything I am and to whom I dedicate this thesis: My mother Gayatri.

Contents

List of Publications

[DMS19] **Counting basic-irreducible factors mod p^k in deterministic poly-time and p-adic applications**

with Rajat Mittal and Nitin Saxena.

34th Computational Complexity Conference (CCC), 15:1–15:29, 2019.

[DS20] **Computing Igusa's local zeta function of univariates in determinstic polynomial-time**

with Nitin Saxena.

14th Algorithmic Number Theory Symposium, ANTS-XIV, *Open Book Series*, vol.4, 197–214, 2020.

[DMS21] **Efficiently Factoring Polynomials Modulo p^4**

with Rajat Mittal and Nitin Saxena.

44th International Symposium on Symbolic and Algebraic Computation (ISSAC), 139–146, 2019.

Full version in Journal of Symbolic Computation, 104:805–823, 2021.

[CDS22] **Solving polynomial systems over non-fields and applications to modular polynomial factoring**

with Sayak Chakrabarti and Nitin Saxena.

Submitted to a Journal, 2022.

In Part I, Chapters 3, 4, 5 are based on [DMS19] while Chapter 6 is based on [DS20]. In Part II, Chapter 7 is based on [DMS21] and Chapter 8 is based on a part of [CDS22] while the remaining part appears in the thesis [Cha22].

List of Figures

Chapter 1

Introduction

Polynomials are one of the most fundamental mathematical objects known, perhaps after numbers such as integers and rationals. In fact, polynomials give definition to new numbers, e.g., complex numbers, and help solve problems about numbers, e.g., primality testing. Behaviour of a polynomial crucially depends on the domain in which they are defined. One such important domain in computer science is integers modulo a prime, or more generally a finite field. In finite fields, the fundamental problems of finding roots and factors of polynomials have been instrumental in some of the major achievements in computer science.

This thesis studies the polynomials in a more general regime– Polynomials over ring of integers modulo a composite N. Though a natural generalization of polynomials modulo a prime, polynomials over composites are less understood. The structural properties over finite fields and the standard methods to understand polynomials fail to hold in such a ring. For example, a univariate polynomial may have exponentially many roots over composites (opposite to at most degree-many roots over a field). We attempt to understand these polynomials by studying a few natural fundamental problems centred around their roots and factors. These problems include finding and counting roots/factors of univariate polynomials, computing p-adic zeta function (encoding number of roots) and solving a system of multivariate polynomial equations. This thesis develops a common framework of polynomial ideals where ideals 'compactly' represent all the exponentially many roots. The nice algebraic-geometric properties of these ideals allows us to efficiently retrieve the required information about roots.

1.1 Polynomials over Composites

Polynomials over a composite N have several applications in computer science and mathematics. In complexity theory, they are used to represent boolean functions and to prove lower bounds [AB01, BBR94, TMB98]. They have also been useful in primality testing [AB03, AKS04], in communication complexity [BGL06] and in combinatorics to construct Ramsey graphs and extremal set systems [Gop14, Gro00]. For more details we refer to the PhD thesis of Gopalan [Gop06].

Polynomials over a composite N show bizarre behaviour which is quite counter-intuitive as compared to behaviour over fields. Shamir gave an interesting example which shows that factoring a polynomial in such a ring is as hard as factoring integers. This, in contrast to a field, is opposite to the belief in complexity theory community. It is believed that integer factoring is intractable but we know efficient algorithms for polynomial factoring whether the field is rational or finite.

Example 1 ([Sha93])**.** *: Let $N = p_1p_2$ where p_1, p_2 are primes. The polynomial $f = x$ factors modulo N as $(p_1^2 + p_2^2)^{-1}(p_1x + p_2)(p_2x + p_1)$. Thus knowing the factors gives us p_1 and p_2.*

The example also shows another bizarre behaviour that even a linear monic polynomial factors in such a ring. Thus to have efficient algorithms for our problems we assume that factorization of N into its prime-power factors is given.

Now the classical Chinese remainder theorem allows us to work over the ring of integers modulo prime-powers. Let $N = \prod_{i=1}^{w} p_i^{k_i}$ where p_is are prime numbers and $k_i \in \mathbb{N}$ for all $i \in [w]$. Then,

$$\frac{\mathbb{Z}}{\langle N \rangle} \cong \frac{\mathbb{Z}}{\langle p_1^{k_1} \rangle} \oplus \cdots \oplus \frac{\mathbb{Z}}{\langle p_w^{k_w} \rangle}$$

where right hand side is the direct product of rings. The isomorphism map is given as,

$$a \bmod N \to (a \bmod p_1^{k_1}, \ldots, a \bmod p_w^{k_w}).$$

Thus from now on we assume N to be a prime power p^k i.e., we will study polynomials with coefficients in a Galois ring such as $\mathbb{Z}/\langle p^k \rangle$. The use of Chinese remaindering in the case of polynomial factoring is discussed in more details in [vzGH98].

Understanding polynomials over ring of integers modulo prime powers are important for many reasons. Besides being natural generalisation of integers modulo a prime, these rings are helpful for understanding operations over p-adic integers $\mathbb{Z}_p$. Informally, p-adic integers $\mathbb{Z}_p$ are the set of numbers in $\mathbb{Z}/\langle p^k \rangle$ as $k \to \infty$ i.e., they have infinite precision and characteristic zero. Thus due to requirement of infinite precision in representing them (as for reals), one deals with their truncation modulo a prime-power p^k in practical applications.

We now move to discuss our problems related to roots and factors of polynomials over Galois rings. Galois rings are very different than finite (Galois) fields, for e.g., a univariate polynomial over Galois rings may have exponentially many roots or factors. Perhaps this is why we don't have efficient algorithms for many fundamental problems, such as polynomial factoring, which have efficient algorithms over finite fields.

1.2 Univariate Factoring and Root Counting

Factoring a univariate over finite fields have many efficient algorithms [Ber67, CZ81, Kal92, KU11, vzGP01] and have found many applications in mathematics and computing [FS15, Kal92, LN94, Sud97, vzGP01]. We consider the following (Galois) ring generalization of this question ($k > 1$):

Problem 1.2.1 (Factoring)**.** *Given $f \in \mathbb{Z}[x]$ of degree d and a prime power p^k, can we find a non-trivial factor of f of degree $\delta < d$ in time $\mathsf{poly}(d, k \log p)$?*

Though this problem is studied since the time of Hensel [Hen18] and it finds a section in many textbooks on elementary number theory [NZM13], yet there is no efficient algorithm known. The issue arises as $f \bmod p^k$ may possess *exponentially* many factors; for e.g., $f = x^2 \bmod p^2$ has a factor $(x + p\alpha)$, for each $\alpha \in \{0, \ldots, p-1\}$. This happens because the ring $\mathbb{Z}/\langle p^k \rangle$ is not a unique factorization domain. A standard method in the literature is to find a factor of $f \bmod p$ and then try to lift this factor modulo a power of p. The following example illustrates the difficulty of lifting.

Example 2. *Let $f = x^2 + p^2$ and $(p, k) := (5, 3)$. The factorization $f = x \cdot x \bmod 5$ lifts to $p = 5$ factorizations of $f = x^2 \bmod 5^2$, as discussed above. But only the factorization*

$f = (x + 2 \cdot 5) \cdot (x + 3 \cdot 5) = (x + 10) \cdot (x + 15) \bmod 5^2$ *lifts to mod* 5^3.

This example raises the question: How to efficiently determine which factorization out of the $p^{O(dj)}$ (exponentially many) factorizations modulo p^j will lift to the higher precision i.e., modulo p^{j+1}?

Hensel's lemma efficiently gives factoring when $f \bmod p$ has two *coprime* factors. Basically, the problem of picking a good factorization out of exponentially many factorizations do not happen with coprime factorization of $f \bmod p$. This is because, Hensel's lemma guarantees that a coprime factorization modulo p lifts *uniquely* to any power of p (Lemma 2.2.2).

Example 3. *Let* $f = x^2 + 10x + 21$. *Then* $f \equiv x(x + 1) \bmod 3$ *and Hensel lemma lifts this factorization uniquely mod* 3^2 *as* $f(x) \equiv (x + 1 \cdot 3)(x + 1 + 2 \cdot 3) \equiv (x + 3)(x + 7) \bmod 9$. *Similarly, this lifting extends to any power of* 3.

Thus, the hard case is to factor f which is power of an irreducible polynomial modulo p as in Example 2. Interestingly, in the hard case, using an extension of Hensel's lemma [BS86, vzGH98], one can solve the problem when k is 'large' i.e., p^k does not divide the discriminant of f ($\mathrm{disc}(f)$). In this case, [CL01, vzGH98] show that irreducible factors of $f \bmod p^k$ correspond to unique p-adic factors, which we get via efficient p-adic factoring algorithms [CG00, Chi87, Chi94, GNP12] (over $\mathbb{Z}_p$).

When k is 'small' the behaviour of factorization pattern is completely elusive. The main issue is that a p-adic irreducible factor could become reducible modulo p^k. Moreover, two different factorizations could be completely unrelated i.e., there is no one to one correspondence between irreducible factors of two different factorizations. This is not the case when k is large due to the connection with unique p-adic factorization of factorizations modulo p^k. Below are examples taken from Gathen and Hartlieb [vzGH96] who first discussed these issues in detail:

Example 4 ([vzGH96])**.** *Polynomial* $f = x^2 + 3^k$ *is irreducible over* $\mathbb{Z}/\langle 3^{k+1}\rangle$, *and so over* 3*-adic field. But,* $f = x^2 \bmod 3^k$ *is reducible.*

Example 5 ([vzGH96])**.** $f = (x^2 + 243)(x^2 + 6)$ *is an irreducible factorization over* $\mathbb{Z}/\langle 3^6\rangle$. *There is another completely unrelated factorization* $f = (x + 351)(x + 135)(x^2 + 243x + 249) \bmod 3^6$.

Due to these difficulties we don't even have an efficient randomized algorithm to factor $f \bmod p^3$ [Săl05, Sir17]. Thus, the major open question in factoring $f \bmod p^k$ is when k is *constant*.

Problem 1.2.2. *Can we find a non-trivial factor of given univariate polynomial $f \in \mathbb{Z}[x]$, of degree d, modulo a constant power of prime p in random $\mathsf{poly}(d, \log p)$-time?*

The related problems of root finding and counting modulo p^k is of significant interest and finds application in arithmetic-algebraic geometry [CS23, DH01, DS20, Zhu20, ZG03], factoring [CG00, Chi87, Chi94], coding theory [BLQ13, Săl05], and hyper/elliptic curve cryptography [Lau04]. Understanding root counting also give better understanding of *root-sets* modulo p^k (i.e. which subsets of $\mathbb{Z}/\langle p^k \rangle$ are zero-sets of some polynomial?). Their combinatorial properties are of significant interest in mathematics [Sie55, CP56, Bha97, DM97, Mau01].

Berthomieu et al. [BLQ13] gave the first efficient randomized algorithm to find and count all the roots of $f(x) \bmod p^k$. Note that $f \bmod p^k$ can have exponentially many roots as discussed before. Berthomieu et al. [BLQ13] efficiently partitioned these roots into at most degree many efficiently representable subsets, which we call *representative roots* (Chapter 2). These subsets allow efficient root-counting as well as finding an arbitrary root. However, derandomization of [BLQ13] is still an open question. Derandomization at least requires efficient deterministic algorithm for root finding and counting over finite fields. Deterministic root finding is a big open problem but efficient deterministic algorithm is known for root counting over finite fields via the Frobenius morphism: $x \mapsto x^p$. Thus we consider the following question:

Problem 1.2.3. *Given an integral univariate polynomial $f(x)$ and a prime power p^k. Can we count all the roots (possibly exponentially many) in deterministic polynomial time?*

Efficiently solving above counting problem immediately gives an efficient deterministic way to test the existence of a root of $f \bmod p^k$. Before this work, the best known deterministic algorithm for root counting is due to Cheng et al. [CGRW18] which takes time exponential in the parameter k.

1.3 Multivariate System of Equations

Finding a common root of a system of multivariate polynomial equations (Search Hilbert's Nullstellensatz or SHN) is a fundamental problem in algebraic geometry [CLO13]. Over finite fields (characteristic p), the problem is well-studied and very important in cryptography [Din21a, KPG99, Pat96] even for $p = 2$ and systems of degree $d = 2$.

The problem has been studied mainly in two parameter settings over finite fields. In small characteristic $p = 2$, there has been a flurry of work [BFSS13, LPT+17, BKW19, Din21b, BDT21] to improve the brute force time complexity from $\hat{O}(2^n)$[1] (for n-variate polynomial equations) to finally $\hat{O}(2^{0.6943n})$ by Dinur [Din21b] which even outperforms $\hat{O}(2^{0.792n})$ complexity for random system of equations (Bardet et al. [BFSS13]).

For large p but constant n, Huang and Wong [HW99] gave an efficient randomized $\mathsf{poly}(d, m, \log p)$ time algorithm to find a common zero of a system of m-many degree-d polynomials in n variables. The decision version of the problem was derandomized by Kayal [Kay05] in same time complexity. Note that the decision version of the problem, i.e., testing existence of a common solution, is NP-complete for *un*bounded number of variables n, even if $p = 2$ and $d = 2$ [EK90, GGL08]. Extending Huang and Wong [HW99] we consider the following problem over Galois rings.

Problem 1.3.1. *Find a common zero of a given system of n-variate m integral polynomials, for a fixed n, of degree at most d modulo a prime power p^k in random $\mathsf{poly}(d, m, k \log p)$-time.*

Galois rings are important in the study of algebraic codes [HKC+94]. Efficiently solving polynomial systems in Galois rings may be fruitful in study of such codes. For example, univariate root finding [BLQ13] has application in Guruswami-Sudan type list-decoding in Galois rings.

When $k > 1$, the classical methods of algebraic geometry fail; which is why perhaps we are not aware of many works on it. Starting $k = 2$, we are unaware of any efficient way to solve SHN; and there is no analogue of the famous theorems like Hilbert's Nullstellensatz (we refer to [Bro87, GS20, Dwi17] to read more about the rich machinery).

[1] $\hat{O}$ subsumes polynomial dependence on degree, number of variables and number of polynomials.

To understand the difficulty, consider a system with just one polynomial f such that $f(y_1, \ldots, y_n) =: \varphi^e \bmod p$, for *ramification* $e > 1$ and φ being *absolutely* irreducible (i.e. irreducible over all extension fields of $\mathbb{F}_p$). In this case, φ must have exponentially many (in $\log p$) roots and all those are *singular* roots (Section 8.3.1) of $f \bmod p$. So, there is no easy way to determine which root will lift, and which root will not, to modulo p^2 (i.e. they *ramify* in a complicated way).

Example 6. *$f(x, y) := (x - y)^2 + p \bmod p^2$. Thus, $f = (x - y)^2 \bmod p$ has p roots which are exponential in $\log p$. Suppose $(x, y) = (a_0 + pa_1, b_0 + pb_1)$ is a zero of $f \bmod p^2$ where $a_0, a_1, b_0, b_1 \in \{0, \ldots, p-1\}$ and $a_0 = b_0$. Putting values of x and y we get $f \equiv p \not\equiv 0 \bmod p^2$. Thus $(x, y) = (a_0 + pa_1, b_0 + pb_1)$ can't be a zero of $f \bmod p^2$. So, $f(x, y)$ has no zero $\bmod p^2$ i.e., none of the p zeros $\bmod p$ lifts to $\bmod p^2$.*

Example 7. *Perturb the above example only slightly to $f(x, y) := (x - y)^2 + px \bmod p^2$. Again, $f \bmod p$ has p roots. However, now $(0, 0)$ is the unique root that lifts $\bmod p^2$.*

The above two examples explain that finding a zero even modulo p^2 is non-trivial. One of the main issue is: how to sample a potentially 'good' zero modulo p out of abundance of zeros which actually lifts? The second example excludes the possibility of a blind random sampling as it may happen that only one zero modulo p lifts. Thus a first good step is to consider the problem when k is constant.

1.4 Computing p-adic Zeta Function

Zeta functions are one of the most important class of generating functions which encode the count of objects encompassing some mathematical structure. Over the years, the study of zeta functions have played a foundational role in the development of mathematics and finds applications in diverse science disciplines. Often zeta functions show special analytic, or algebraic properties, the study of which reveals various striking information about the encoded object which is hidden otherwise. Inevitably it has driven the whole new areas of mathematical research and this makes zeta functions specially interesting to mathematicians.

A classic example is the famous Riemann zeta function [Rie59] which encodes the density, and distribution, of prime numbers [Con03, Tit86]. Riemann zeta function is a *global* zeta function. Later many *local* zeta functions, i.e., associated to a specific prime p, had been studied such as Hasse-Weil zeta function [Wei48, Wei49] which encodes the count of zeros of a system of polynomial equations over finite fields (of a specific characteristic p). It lead to the development of modern algebraic-geometry (see [Del74, Gro64]).

In this thesis our object of interest are polynomials over integers modulo prime powers and the local zeta function which encodes the count on their roots is known as Igusa's local zeta function. Formally, *Igusa's local zeta function* $Z_{f,p}(s)$, attached to a polynomial over p-adic integers, $f(\mathbf{x}) \in \mathbb{Z}_p[x_1, \ldots, x_n]$, is defined as,

$$Z_{f,p}(s) \ := \ \int_{\mathbb{Z}_p^n} |f(\mathbf{x})|_p^s \cdot |d\mathbf{x}|$$

where s is a complex variable with $Re(s) > 0$, $|.|_p$ denotes the absolute value over p-adic numbers $\mathbb{Q}_p$, and $|d\mathbf{x}|$ denotes the Haar measure on $\mathbb{Q}_p^n$ normalized so that $\mathbb{Z}_p^n$ has measure 1. These zeta functions, defined by Weil [Wei64, Wei65], bear the name of Jun-Ichi Igusa who studied them extensively [Igu74, Igu75, IR78] and, using the method of resolution of singularities, proved that $Z_{f,p}(s)$ converges to a rational function. A fundamental computational problem in arithmetic geometry is,

Problem 1.4.1 (Computing IZF)**.** *Compute the rational function form of Igusa's local zeta function* $Z_{f,p}(s)$ *for a given integral polynomial* f *and prime* p*.*

Original definition of Igusa's local zeta function is not necessarily needed to compute its rational form as it can be computed by computing rational form of a closely related *Poincaré series* $P(t)$. $P(t)$, attached to f and p, is defined as

$$P(t) \ := \ \sum_{i=0}^{\infty} \frac{N_i(f)}{p^{ni}} \cdot t^i$$

where t is a complex variable with $|t| < 1$, and $N_i(f)$ is the count on roots of f mod p^i. The trivial root count $N_0(f)$ is defined to be 1. Note that p^{ni} is maximum root count of any n-variate polynomial modulo p^i and hence the term $\frac{N_i(f)}{p^{ni}} \leq 1$. The relation between $Z_{f,p}(s)$

and $P(t)$ has been shown in [Igu00] as

$$P(t) \;=\; \frac{1 - t \cdot Z_{f,p}(s)}{1 - t} \quad \text{with} \quad t =: p^{-s}.$$

So rationality of $Z_{f,p}(s)$ implies rationality of $P(t)$ and vice versa. This relation with Poincaré series makes it clear how Igusa's local zeta function directly encodes the root count of polynomials modulo prime powers.

Many researchers have tried to calculate the expression for Igusa zeta function for various polynomial families. However, not much has been said about their algorithmic aspect except the recent work of Chakrabarti and Saxena [CS23] which takes exponential time in the input size. Indeed, the computation of Igusa zeta function for a general multivariate polynomial seems to be an intractable problem since root counting of a multivariate polynomial over a finite field is known to be #P-hard [EK90].

In this thesis, we focus on the computation of Igusa zeta function when the associated polynomial is *univariate*. In the case of univariate polynomials one naturally expects an elementary proof of convergence, as well as an efficient algorithm to compute the Igusa zeta function. Indeed, Zúñiga-Galindo [ZG03] gives deterministic polynomial-time algorithm for univariates in restricted case where the univariate completely splits over $\mathbb{Q}$ (with the factorization given in the input). We ask,

Problem 1.4.2 (Computing univariate IZF)**.** *Is there a deterministic polynomial time algorithm to compute the rational function form of Igusa zeta function, associated to a given univariate polynomial $f \in \mathbb{Z}[x]$ and prime p? Is there an elementary proof of convergence?*

Igusa zeta function for a univariate polynomial f is connected to the previously discussed problem of root counting of f modulo prime powers p^k, i.e., computing $N_k(f)$. However, it is not clear how computing $N_k(f)$ for few values of k is sufficient for computing associated Poincaré series though an explicit expression for $N_k(f)$ may help. We ask,

Problem 1.4.3 (Computing $N_k(f)$)**.** *Can we compute explicit expression for number of roots of $f \in \mathbb{Z}[x]$ modulo prime-power p^k (if one exists) in deterministic polynomial time?*

1.5 Our Contribution

One of the core contribution of this thesis is to develop a framework of polynomial ideals which we call— The method of ideals. A polynomial ideal is a collection of one or more polynomials which together act as a unit with its own set of operations and behaves in many ways similar to a polynomial. All the problems we make progress on, partially or completely, are either about roots of polynomials over Galois rings or reduced to the one. The method works to handle these roots in two ways:

Firstly, it partitions the set of all roots (possibly exponentially many) into a relatively small number of subsets. The method returns a set of special ideals in the end, each one effectively and efficiently represents one of these subsets. That means, a large set of roots are compactly represented by a small set of special ideals which provide the information about these roots in an efficient manner.

Secondly, the method constructs those special ideals incrementally by giving an efficient way to lift a possibly large set of roots compactly and simultaneously from a local space, i.e., modulo p, to a global space, i.e., p-adic integers $\mathbb{Z}_p$. This lifting is implicit without individual access to roots and happens using a set of polynomials with nice algebraic-geometric properties modulo p. This especially helps in those situations where direct access to roots are prohibited either due to the requirement of determinism or their number being exponential as evident from examples in previous sections. Further the method inherits those nice 'local' algebraic-geometric properties modulo p to the final special ideals over $\mathbb{Z}_p$. It is these properties which gives us required information about roots such as their count, p-adic multiplicity or the root itself.

Our method is versatile and applies in the following two different contexts:

- Derandomizing known randomized algorithms and applications.

- Facilitating random sampling for inefficient brute force algorithms.

1.5.1 Derandomization via Ideals and Applications

Derandomization forms the first part of the thesis where we completely solve some of the problems using the method of ideals. Basically we completely derandomize the algorithms of [BLQ13] for univariate root counting modulo prime powers and extend it to arbitrary Galois rings. We also derandomize univariate root counting over p-adic integers and its unramified extensions. As applications, we count special factors called *basic-irreducible* factors modulo prime powers and completely solve the problem of computing Igusa zeta function for univariate polynomials in deterministic polynomial time.

Counting Roots of a Univariate Polynomial

We give the first deterministic polynomial time algorithm to count all the roots of a univariate polynomial.

Theorem (Theorem 4.0.1)**.** *Let p be a prime, $k \in \mathbb{N}$ and $f(x) \in \mathbb{Z}[x]$ of degree d. Then all the roots of $f \bmod p^k$ can be counted in deterministic poly$(d, k \log p)$-time.*

Theorem 4.0.1 is proved in Section 4.1.3. We also introduce an algorithm that efficiently partitions the (possibly exponentially large) set of roots, into at most d subsets. This partitioning via special *split ideals*, is reminiscent of the age-old fact that there are at most $\deg(g)$ roots of a polynomial $g(x)$ over a field. The parameters of a split ideal, *degree* and *length*, immediately give us the count on the number of roots represented by that split ideal.

Next, we extend the ideas for counting roots to count all the basic-irreducible factors of $f \bmod p^k$. A *basic-irreducible* factor of $f \bmod p^k$ is a factor that is irreducible modulo p.

Theorem (Theorem 5.0.1)**.** *Let p be a prime, $k \in \mathbb{N}$ and $f(x) \in \mathbb{Z}[x]$ of degree d. Then all the basic-irreducible factors of $f \bmod p^k$ can be counted in deterministic poly$(d, k \log p)$-time.*

We achieve this by extending the idea of counting roots to a general Galois ring.

Theorem (Informal Corollary 5.0.2)**.** *We can count all the roots of a univariate f with integer coefficients over a given Galois ring $\mathbb{G}$ in deterministic time polynomial in size of description of f and $\mathbb{G}$.*

Our method generalizes to efficiently count all the roots of a given polynomial $f(x) \in (\mathbb{F}[t]/\langle h(t)^k \rangle)[x]$ for a given polynomial h (respectively $f \in \mathbb{F}[[t]][x]$ with power-series coefficients); assuming that $\mathbb{F}$ is a field over which root counting is efficient. For example $\mathbb{F}$ could be $\mathbb{Q}, \mathbb{R}, \mathbb{F}_p$ and their algebraic extensions.

Computing Igusa's Local Zeta Function

We will compute the Igusa zeta function $Z_{f,p}(s)$ by finding the related Poincaré series $P(t) =: A(t)/B(t)$.

Theorem (Theorem 6.0.1)**.** *We are given a univariate integral polynomial $f(x) \in \mathbb{Z}[x]$ of degree d, with coefficients magnitude bounded by $C \in \mathbb{N}$, and a prime p. Then, we compute Poincaré series $P(t) = A(t)/B(t)$, associated with f and p, in deterministic poly$(d, \log C + \log p)$ time.*

The degree of the integral polynomial $A(t)$ is $\tilde{O}(d^2 \log C)$ and that of $B(t)$ is $O(d)$.

Our method gives an elementary proof of rationality of $Z_{f,p}(s)$ as a function of $t = p^{-s}$. Previously, Zúñiga-Galindo [ZG03] gave a deterministic polynomial time algorithm to compute $Z_{f,p}(s)$, if f completely splits over $\mathbb{Q}$ and the roots are provided. While, our Theorem 6.0.1 works for *any* input $f \in \mathbb{Z}[x]$. Cheng et al.[CGRW18] could compute $Z_{f,p}(s)$ in deterministic polynomial time, in the special case where the degree of $A(t), B(t)$ is constant.

We achieve the rational form of $Z_{f,p}(s)$ by getting an explicit formula for the number of zeros $N_k(f)$, of $f \bmod p^k$, which sheds new light on the properties of the function $N_k(\cdot)$. Eventually, it gives an elementary proof of the rationality of the Poincaré series $\sum_{i=0}^{\infty} N_i(f) \cdot (p^{-1}t)^i$.

Theorem (Corollary 6.0.2)**.** *Let k be large enough, namely, $k \geq k_0 := O(d^2(\log C + \log d))$. Then, we give a closed form expression for $N_k(f)$ (in Theorem 6.4.1).*

Interestingly, if f has non-zero discriminant, then $N_k(f)$ is constant (independent of k) for all $k \geq k_0$.

The closed form expression for $N_k(f)$ involves the parameters e_i which are multiplicities of $\mathbb{Z}_p$ roots of f. When we compute $N_k(f)$ we also get the first deterministic polynomial time

algorithm to count *all* the $\mathbb{Z}_p$-roots of f with multiplicity.

Theorem (Corollary 6.0.3)**.** *We are given a univariate integral polynomial $f(x) \in \mathbb{Z}[x]$ of degree d, with coefficients magnitude bounded by $C \in \mathbb{N}$, and a prime p. Then, we can count all the p-adic integral roots (in $\mathbb{Z}_p$) of f in deterministic poly$(d, \log C + \log p)$-time.*

1.5.2 Random Sampling via Ideals

In the second part of the thesis we use the method of ideals to present randomized algorithms for some of the problems. Using our framework we could solve a given system of polynomial equations in random polynomial time modulo the constant power of a prime. We also give efficient randomized algorithm in special cases of univariate polynomial factoring by reducing it to the former problem.

Solving System of Polynomial Equations

We consider the problem of finding common zero of a given system of integral polynomials modulo p^k when k is constant.

Theorem (Simplified Theorem 8.1.3)**.** *Given a system of n-variate polynomials $f_1, \ldots, f_m$, with integer coefficients, of degrees at most d and a prime power p^k. We can find a common root of the system in randomized* poly$(d^{c_{nk}}, m, \log p)$*-time, where $c_{nk} \leq (nk)^{O((nk)^2)}$.*

Theorem 8.1.3 gives a polynomial time algorithm when $n+k$ is constant. Recall that SHN is intractable for growing n even when $k = 1$. Theorem 8.1.3 also resolves the open question asked in [RRZ21, Zhu20] to efficiently find a point on a curve mod p^k, for fixed k. Theorem 8.1.3 efficiently extends the root-finding result of Huang and Wong [HW99] from Galois fields to Galois rings of characteristic p^k, for k constant. In fact, the doubly exponential complexity in nk comes from the doubly exponential time complexity of [HW99] which we use for root finding modulo p. Our algorithm returns a set of special triangular ideals in nk variables, we call 'absolute' ideals. These ideals collectively encode all the root of the system and it is easy to find a root randomly from each of these ideals as they are absolutely irreducible modulo p.

Factoring a Univariate Polynomial

We see that even after a long series of efforts [BLQ13, CL01, Kli97, KRRZ20, Săl05, Sir17, vzGH96, vzGH98], efficient modular factoring has remained elusive even for $f \bmod p^3$. Naturally, we would like to first understand the difficulty of the problem when k is constant. In this direction we make our first progress by devising a unified method which solves the problem when $1 < k \leq 4$. Our first result is,

Theorem (Theorem 7.0.1)**.** *Let p be prime, $k \leq 4$ and $f(x)$ be a univariate integral polynomial. Then, $f(x) \bmod p^k$ can be factored (and tested for irreducibility) in randomized poly($\deg f, \log p$) time.*

The procedure to factor $f \bmod p^4$ also factors mod p^3 and mod p^2 (and tests for irreducibility) in randomized poly($\deg f, \log p$) time. This solves the open question of efficiently factoring $f \bmod p^3$ [Sir17] and generalizes [Săl05]. Our method can as well be used to factor a 'univariate' polynomial $f \in \left(\mathbb{F}_p[z]/\langle \psi^k \rangle\right)[x]$, for $k \leq 4$ and irreducible $\psi(z) \bmod p$, in randomized poly($\deg f, \deg \psi, \log p$) time.

Next, we do more than just factoring f modulo p^k for $k \leq 4$. Given that f is power of an irreducible mod p, which is the hard case of Hensel lemma, we show that our method works in this case to give all the lifts $g(x) \bmod p^k$ (possibly exponentially many) of any given factor $\tilde{g}$ of $f \bmod p$, for $k \leq 4$.

Theorem (Theorem 7.0.2)**.** *Let p be prime, $k \leq 4$ and $f(x)$ be a univariate integral polynomial such that $f \bmod p$ is a power of an irreducible polynomial. Let $\tilde{g}$ be a given factor of $f \bmod p$. Then, in randomized poly($\deg f, \log p$) time, we can compactly describe (and count) all possible factors of $f(x) \bmod p^k$ which are lifts of $\tilde{g}$ (or report that there is none).*

Theorem 7.0.2 can be seen as refinement of Hensel lifting method to $\mathbb{Z}/\langle p^k \rangle$, $k \leq 4$. To lift a factor f_1 of $f \bmod p$, Hensel lemma relies on a cofactor f_2 which is coprime to f_1. Our method needs no such assumption and it directly lifts a factor $\tilde{g}$ of $f \bmod p$ to (possibly exponentially many) factors $g(x) \bmod p^k$.

The known factoring methods mod p work by first reducing the problem to that of root finding mod p. In this work, we efficiently reduce the problem of factoring $f(x) \bmod p^k$ to

that of finding roots of some polynomial $E(y) \in (\mathbb{Z}[x])[y]$ modulo a *bi-generated* ideal $\langle p^k, \varphi^\ell \rangle$, where $\varphi(x)$ is an irreducible factor of $f(x) \bmod p$. With the help of the special structure of $E(y)$ we could efficiently find all the roots y (possibly exponentially many) mod $\langle p^k, \varphi^\ell \rangle$ when $k \leq 4$.

It remains open whether this technique extends to $k = 5$ and beyond. However our reduction of factoring to root finding of $E(y)$ was only partial and it reduces further. Basically it efficiently reduces the problem of finding a constant degree factor of $f \bmod p^k$ to the problem of finding a solution to a system of polynomial equations of constant degree in constant number of variables when k is constant. This advances the state-of-the-art by giving first general result: to efficiently compute a *constant-degree* factor of $f \bmod p^k$, when k is *constant*. In particular, we can factor a *fixed* degree univariate polynomial into irreducibles.

Theorem (Theorem 8.1.1)**.** *Given a univariate polynomial $f \in \mathbb{Z}[x]$ and a prime-power p^k, in binary, with k fixed. We can find a constant-degree factor g of $f \bmod p^k$ in randomized $\mathsf{poly}(\deg(f), \log p)$-time; or decide that none exists.*

The difficult case in factoring $f \bmod p^k$ happens when $f \bmod p$ has no two coprime factors; as this forbids the well-known Hensel's lifting. For example, $f \equiv \varphi^e \bmod p$ for a $\varphi \in \mathbb{Z}[x]$ which is irreducible mod p. For such an f, we call e to be the *ramification-degree* of f. In fact, our proof method provides more general factors:

Theorem (Corollary 8.1.2)**.** *Given $f \in \mathbb{Z}[x]$ and prime-power p^k, with k constant. We can find a factor g of $f \bmod p^k$ in randomized $\mathsf{poly}(\deg(f), \log p)$-time, where the* ramification-degree *of g is at most a given constant; or decide that no such factor exists.*

The brute-force approach takes time $p^{\Omega(k\delta)}$; which is clearly *exponential* (in $\log p$), even for fixed k and fixed ramification-degree δ. Thus, for constant k, our methods extend the results of [BLQ13] from *un*ramified factors to ramified factors; albeit of 'low' ramification-degree. Indeed, our algorithm is a first major step towards factoring polynomials modulo p^k for any constant $k \geq 5$.

1.6 Thesis Organization

We study polynomials over composites under the framework of what we call, *the method of ideals*. Chapter 2 gives the common preliminaries required for the later chapters. The presentation of thesis is in two parts:

Part I is about the derandomization results via the method of ideals and its applications. Chapter 3 introduces our algebraic tool, split ideals and their properties, required for efficient de-randomization of root counting in Chapter 4. It gives further applications of split ideals in deterministically, (1) counting strongly irreducible factors (Chapter 5) and (2) computing Igusa zeta function for univariates and derandomizing p-adic root counting (Chapter 6).

Part II is about the application of the method of ideals in random sampling. It turns to the problems of univariate factoring (Chapters 7, 8) and multivariate system solving (Chapter 8). Finally, Chapter 9 concludes the thesis and gives open problems and further directions where the framework developed in the thesis could be useful.

Chapter 2

Preliminaries

2.1 Basic Notations and Definitions

$\mathbb{F}$ denotes a field unless explicitly stated otherwise. All the rings we deal in this thesis are commutative rings with unity. $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ denotes natural numbers, integers, rational numbers, real numbers and complex numbers respectively. We usually denote finite fields by $\mathbb{F}_p$ or $\mathbb{F}_q$ where p is prime and q is a power of p. We use notation $[n]$, called range n, to denote the set $\{1, \ldots, n\}$. The function $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ takes argument a real number r such that $\lfloor r \rfloor$ is the greatest integer less than or equal to r and $\lceil r \rceil$ is the least integer greater than or equal to r. $\mathsf{Rad}(\cdot)$ denotes the radical of a polynomial or an ideal which means squarefree part of the polynomial or the ideal respectively. We use the notation $a := b$ to imply that a is defined to be equal to b. For a polynomial f, $\deg(f)$ denotes its total degree and $\deg_y(f)$ denotes its degree with respect to a variable y. The function $\mathsf{poly}(\cdot)$ is used to denote a polynomial function in its arguments.

Associated to a prime p, the field of p-adic rational numbers is denoted as $\mathbb{Q}_p$. This is also called non-archimedean local field. As $\mathbb{Z}$ is the ring of integers of $\mathbb{Q}$ there is an associated ring of integers $\mathbb{Z}_p$, called p-adic integers, to the field $\mathbb{Q}_p$. A p-adic integer is a formal power series $\sum_{i \geq 0} a_i p^i$ where $0 \leq a_i \leq p-1$. $\mathbb{Z}_p$ is an integral domain and $\mathbb{Z} \subset \mathbb{Z}_p$. In such a field $\mathbb{Q}_p$ there exists a non-archimedean valuation function $v_p : \mathbb{Q}_p \to \mathbb{Z} \cup \{\infty\}$. Formally, the *valuation* $v_p(a)$ of $a \in \mathbb{Z}_p$ ($\mathbb{Z}_p$ is a UFD) is defined to be the highest power of p dividing

a, when $a \neq 0$, and ∞ when $a = 0$. This definition extends to the rationals $\mathbb{Q}_p$ naturally as $v_p(\frac{a}{b}) := v_p(a) - v_p(b)$, where $b \neq 0$ and $a, b \in \mathbb{Z}_p$ (see [Kob77] as reference text).

An unramified extension K of $\mathbb{Q}_p$, denoted as $K/\mathbb{Q}_p$, is an algebraic extension $\mathbb{Q}_p[z]/\langle\varphi(z)\rangle$ where $\varphi \in \mathbb{Q}_p[z]$ is an irreducible polynomial modulo p. The degree of such an extension is the $\deg(\varphi)$. $K/\mathbb{Q}_p$ is called a *ramified* field extension, if φ is irreducible over $\mathbb{Z}_p$ but not modulo p. We denote the ring of integers of unramified extension $K/\mathbb{Q}_p$ as $\mathcal{O}_K$. These are more general p-adic integers than $\mathbb{Z}_p$. (we refer to [Kob77] for more on this).

Radical of a univariate polynomial $h(x)$ over a field $\mathbb{F}$ is defined to be the univariate polynomial, denoted by $\mathsf{Rad}(h)$, which is the product of coprime irreducible factors of h.

Discriminant of a polynomial $h(x) \in \mathbb{F}[x]$ is defined as $D(h) := h_m^{2m-1} \cdot \prod_{1 \leq i < j \leq m}(r_i - r_j)^2$, where $\mathbb{F}$ is a field, r_i's are the roots of $h(x)$ over the algebraic closure $\bar{\mathbb{F}}$, degree of h is m, and h_m is its leading coefficient.

The discriminant $D(h)$ is an element of $\mathbb{F}$. It is clear by the definition: all the roots of h are distinct iff $D(h) \neq 0$. For example, discriminant of radical is nonzero.

Now, we will present some basic concepts in commutative algebra. $\mathbf{x}$ denotes a variable tuple $(x_1, \ldots, x_n)$.

Zero-Divisors. An element $a \neq 0$ is a zero-divisor in a ring R if there is a $b \neq 0$ in R such that $ab = 0$.

Ideals. An ideal I of a commutative ring $R(+, \cdot)$ is a closed subset of R under addition such that for any $a \in R$ and $b \in I$ we have $a \cdot b \in I$. When R is a polynomial ring such as $\mathbb{F}[\mathbf{x}]$ then the ideal I is called a polynomial ideal.

Ideal-Generators. Let $f_1, \ldots, f_m \in R$, R is a commutative ring, and $I \subseteq R$ is defined as $I := \{f_1 g_1 + \cdots + f_m g_m \mid g_1, \ldots, g_m \in R\}$. Then I is called an ideal generated by $f_1, \ldots, f_m$ and denoted as $I = \langle f_1, \ldots, f_m\rangle$.

Hilbert basis theorem in commutative algebra says that every ideal of a Noetherian ring (e.g., polynomial rings over a field or a Galois ring) has finite number of generators.

Ideal sum. Let I, J be two ideals of the commutative ring R then $I + J$ is defined to be the set $I + J := \{f + g \mid f \in I, g \in J\}$. $I + J$ is an ideal and if $I := \langle f_1, \ldots, f_m\rangle$ and $J := \langle g_1, \ldots, g_\ell\rangle$ then $I + J = \langle f_1, \ldots, f_m, g_1, \ldots, g_\ell\rangle$.

Quotient ideals. Given two ideals I and J of a commutative ring R, we define the quotient of I by J as, $I : J := \{a \in R \mid aJ \subseteq I\}$. It can be easily verified that $I : J$ is an ideal.

2.2 Some Useful Results

This section is devoted to some useful preliminary results. First we will state results related to factoring and lifting factorization. We then define and prove some properties of our main ring object in this thesis– Galois rings.

2.2.1 Factoring and Lifting

The first efficient randomized algorithm to find all the roots of a univariate polynomial over a finite field $\mathbb{F}_q$ was given by Cantor-Zassenhaus [CZ81] (Equivalently, it finds all irreducible factors as well.). We state the following theorem, due to Kedlaya-Umans [KU11] which has best known time complexity over a general finite field $\mathbb{F}_q$.

Theorem 2.2.1 (Kedlaya-Umans [KU11])**.** *Given a univariate degree d polynomial $f(x)$ over a finite field $\mathbb{F}_q$. There is a randomized algorithm to find all the roots of f in $\mathbb{F}_q$ in $O(d^{1.5+o(1)} \log^{1+o(1)} q + d^{1+o(1)} \log^{2+o(1)} q)$ bit operations.*

Currently, it is a big open question to derandomize the preceding theorem. The known deterministic algorithms are 'inefficient' for example, the well known Berlekamp's algorithm [Ber67] takes time $\tilde{O}(p \cdot (dn)^{\omega})$ where $q = p^n$ for p prime and ω is matrix-multiplication exponent.

Below we state a lemma, originally due to Kurt Hensel [Hen18], for $\mathcal{I}$-adic lifting of factorization of a given univariate polynomial. Over the years, Hensel's lemma has acquired many forms in different texts, version presented here is due to Zassenhaus [Zas69].

Lemma 2.2.2 (Hensel's lemma [Hen18])**.** *Let R be a commutative ring with unity and the corresponding polynomial ring over it $R[x]$. Let $\mathcal{I} \subseteq R$ be an ideal of ring R. Given a polynomial $f(x) \in R[x]$, suppose f factorizes as*

$$f = gh \bmod \mathcal{I},$$

such that $gu + hv = 1 \bmod \mathcal{I}$ *(for some* $g, h, u, v \in R[x]$*). Then, given any* $\ell \in \mathbb{N}$*, given* g, h, u, v *we can efficiently compute* $g^*, h^*, u^*, v^* \in R[x]$*, such that,*

$$f = g^* h^* \bmod \mathcal{I}^\ell.$$

Here $g^* = g \bmod \mathcal{I}$*,* $h^* = h \bmod \mathcal{I}$ *and* $g^* u^* + h^* v^* = 1 \bmod \mathcal{I}^\ell$ *(i.e.,* pseudo-*coprime lifts). Moreover* g^* *and* h^* *are unique up to multiplication by a unit.*

Using Hensel's lemma, for the purpose of counting roots and basic-irreducible factors, a univariate polynomial $f(x) \in \mathbb{Z}[x]$ can be assumed to be a power of an irreducible mod p. Recall that a basic-irreducible factor is an irreducible factor which is also irreducible mod p.

Lemma 2.2.3. *Suppose a univariate* $f(x) \in \mathbb{Z}[x]$ *factors uniquely, over* $\mathbb{F}_p$*, into coprime powers as,* $f \equiv \prod_{i=1}^{m} {\varphi_i}^{e_i}$ *, where each* $\varphi_i \in \mathbb{Z}[x]$ *is irreducible mod* p *and* $m, e_i \in \mathbb{N}$*. Then, for all* $k \in \mathbb{N}$*,*

1. f *factorizes mod* p^k *as* $f = g_1 g_2 \dots g_m$*, where* g_i*'s are mutually co-prime mod* p^k *and* $g_i \equiv {\varphi_i}^{e_i} \bmod p$*, for all* $i \in [m]$*.*

2. *any basic-irreducible factor of* $f(x) \bmod p^k$ *is a basic-irreducible factor of a unique* $g_j \bmod p^k$*, for some* $j \in [m]$*. Let* $B_k(h)$ *denote the number of (coprime) basic-irreducible factors of* $h(x) \bmod p^k$*. Then* $B_k(f) = \sum_{i=1}^{m} B_k(g_i)$ *.*

3. *any root of* f *mod* p^k *is a root of a unique* g_i *mod* p^k*. Let* $N_k(h)$ *denote the number of (distinct) roots of* $h(x) \bmod p^k$*. Then* $N_k(f) = \sum_{i=1}^{m} N_k(g_i)$*.*

Proof. We can apply Hensel's lemma by taking ring $R := \mathbb{Z}$ and ideal $\mathcal{I} := \langle p \rangle$. The co-prime factorization of $f \bmod p$ lifts to a unique coprime factorization $f \equiv g_1 g_2 \dots g_m \bmod p^k$, for any $k \in \mathbb{N}$ and $g_i \equiv {\varphi_i}^{e_i} \bmod p$.

Any basic-irreducible factor $h(x)$ of $f(x) \bmod p^k$ has to be $h \equiv \varphi_i \bmod p$ for some $i \in [m]$; otherwise, h will become reducible mod p. Since g_i's are co-prime and $h | f \bmod p^k$, h must divide a unique g_i. So, any basic-irreducible factor h of $f(x) \bmod p^k$ is a basic-irreducible factor of a unique $g_j \bmod p^k$. Clearly, any basic-irreducible factor of a g_i is also a basic-irreducible factor of $f \bmod p^k$. This proves $B_k(f) = \sum_{i=1}^{m} B_k(g_i)$.

The third part follows from a similar reasoning as the second part. □

2.2.2 Properties of Galois rings: Ring analogues of Finite Fields

A Galois ring $\mathbb{G}$ of characteristic p^k and size p^{kb} (p prime and $k, b \in \mathbb{N}$), denoted as $\mathbb{G}(p^k, b)$, is defined as the ring $\mathbb{G} := \mathbb{Z}[y]/\langle p^k, \varphi(y)\rangle$, where $\varphi(y) \in \mathbb{Z}[y]$ is an irreducible modulo p of degree b [McD74]. Galois ring $\mathbb{G}$ is the ring analogue of the finite field $\mathbb{F}_q := \mathbb{Z}[y]/\langle p, \varphi(y)\rangle$ of size p^b. Similar to finite fields, two Galois rings of same characteristic and size are isomorphic to each other. Let us prove some useful properties of $\mathbb{G}$ below.

Proposition 1. *Let $\chi(x) \in \mathbb{Z}[x]$ be any irreducible modulo p of degree b. Then there are b distinct roots of $\chi(x)$ in $\mathbb{G}$, each of them are of the form r^{p^i} modulo p ($i \in \{0, \ldots, b-1\}$) where $r \in \mathbb{G}$ is one of the roots of $\chi(x)$.*

Proof. $\mathbb{G}/\langle p\rangle$ is isomorphic to the finite field of degree b over $\mathbb{F}_p$. So, irreducible $\chi(x) \in \mathbb{F}_p[x]$ has exactly b roots in $\mathbb{G}/\langle p\rangle$ [LN94, Ch.2]. By Hensel Lemma 2.2.2, roots in $\mathbb{G}/\langle p\rangle$ can be lifted to $\mathbb{G}$ uniquely. Hence, $\chi(x)$ has exactly b distinct roots in $\mathbb{G}$. In $\mathbb{G}/\langle p\rangle$ these roots are conjugates (since $\chi(x) \in \mathbb{F}_p[x]$ is irreducible) so if r is one of the roots then all b conjugates are of the form r^{p^i} modulo p ($i \in \{0, \ldots, b-1\}$). □

Using Proposition 1, denote the roots of $\varphi(x)$ by $y_0, \ldots, y_{b-1}$ with $y_i \equiv y_0^{p^i} \bmod p$ for all $i \in \{0, \ldots, b-1\}$. Without loss of generality, take $y = y_0$ (Since $\mathbb{G} := \mathbb{Z}[y]/\langle p^k, \varphi(y)\rangle$). We will use y and y_0 interchangeably in the proof of following proposition.

Proposition 2 (Symmetries of $\mathbb{G}$)**.** *Let a map $\psi_j : \mathbb{G} \to \mathbb{G}$ is defined as $\psi_j(y_0) = y_j$ for $j \in \{0, \ldots, b-1\}$. Then these ψ_js are only automorphisms of $\mathbb{G}$ under which the ring $R := \mathbb{Z}/\langle p^k\rangle$ remains fixed. Moreover, if j is co-prime to b then ψ_j fixes R and nothing else.*

Proof. Since coefficients of $\varphi(x)$ belong to R, an automorphism fixing R should map the root y_0 to another of its roots y_j. We only need to show that ψ_j is an automorphism (it is a valid map since we take $y = y_0$).

Writing elements of $\mathbb{G}$ as a polynomial in $y_0 = y$, it can be verified that $\psi_j(ab) = \psi_j(a)\psi_j(b)$ and $\psi_j(a+b) = \psi_j(a) + \psi_j(b)$, so ψ_j is a homomorphism.

Similarly, if $\psi_j(g) = 0$, writing g in terms of y_0, we get that $g = 0$. So, kernel of ψ_j is the set $\{0\}$; thus it is an isomorphism.

For the moreover part, let ψ_j be such that j is coprime to b. We will show a stronger statement by induction: for any $i \leq k-1$, if $a(y_0) = \psi_j(a(y_0))$ in $\mathbb{G}/\langle p^i \rangle$, then $a(y_0) \in \mathbb{Z}/\langle p^i \rangle$.

Base case: If $i = 1$ and $j = 1$, then $a(y_0) = \psi_1(a(y_0)) \bmod p \Rightarrow a(y_0) = a(y_0)^p \bmod p$. It means $a(y_0) \in \mathbb{Z}/\langle p \rangle$.

If j is coprime to b, then ψ_j generates ψ_1 modulo p. So, $a(y_0) = \psi_j(a(y_0)) \bmod p$ implies that, $a(y_0) \bmod p =: a_0 \in \mathbb{Z}/\langle p \rangle$.

This argument also proves: for any $i \leq k$, if $a(y_0) = a(y_j)$ in $\mathbb{G}/\langle p^i \rangle$, then $a(y_0) \in \mathbb{F}_p$ (in other words, $a(y_0)$ is y_0 free).

Induction step: Let us assume that $a(y_0) = \psi_j(a(y_0))$ in $\mathbb{G}/\langle p^i \rangle$. By the previous argument, $a(y_0) = a_0 + pa'(y_0)$, where $a_0 \in \mathbb{Z}/\langle p \rangle$ and $a'(y_0) \in \mathbb{G}/\langle p^{i-1} \rangle$.

From the definition, $a(y_0) = \psi_j(a(y_0))$ iff $a'(y_0) = \psi_j(a'(y_0))$ in $\mathbb{G}/\langle p^{i-1} \rangle$. By induction hypothesis, the latter is equivalent to $a'(y_0) \in \mathbb{Z}/\langle p^{i-1} \rangle$. So, $a(y_0) \in \mathbb{Z}/\langle p^i \rangle$.

Hence, the only fixed elements under the map ψ_j (j coprime to b) are integers; in $\mathbb{Z}/\langle p^k \rangle$.

□

2.3 Randomized Root Finding modulo Prime Powers

In this section we give a simplified exposition of the results of [Pan95, BLQ13] about finding and counting all the roots of a univariate polynomial in the ring of integers modulo prime powers. As discussed before, a univariate polynomial can have exponentially many roots in such a ring, unlike fields, but we get all the roots in a succinct representation, a small set of 'representative roots', which is efficient and contains all the roots in an order.

2.3.1 Representatives and Representative Roots

Let R be a commutative ring with addition $+$ and multiplication $\cdot$ and let S be a non-empty subset of R. The product of the set S with a scalar $a \in R$ is defined as $aS := \{as \mid s \in S\}$. Similarly, the sum of a scalar $u \in R$ with the set S is defined as $u + S := \{u + s \mid s \in S\}$. Note that the product and the sum operations used inside the set are borrowed from the underlying ring R. Also note that if S is the empty set then so are aS and $u + S$ for any

$a, u \in R$.

Representatives. The symbol '$*$' in a ring R, wherever it appears, denotes any possible choice of an arbitrary element of R. For example, suppose $R = \mathbb{Z}/\langle p^k \rangle$ for a prime p and a positive integer k. In this ring, we will use the notation $y = y_0 + py_1 + \cdots + p^i y_i + p^{i+1}*$, where $i + 1 < k$ and each $y_j \in R/\langle p \rangle$ ($R/\langle p \rangle$ is same as $\mathbb{Z}$ modulo p), to denote a set $S_y \subseteq R$ such that

$$S_y = \{y_0 + \cdots + p^i y_i + p^{i+1} y_{i+1} + \cdots + p^{k-1} y_{k-1} \mid \forall y_{i+1}, \ldots, y_{k-1} \in R/\langle p \rangle\}.$$

Notice that the number of distinct elements in R represented by y is $|S_y| = p^{k-i-1}$. We will call y as *representatives*.

We will sometimes write the set $y = y_0 + py_1 + \cdots + p^i y_i + p^{i+1}*$ succinctly as $y = v + p^{i+1}*$, where $v \in R$ stands for $v = y_0 + py_1 + \cdots + p^i y_i$.

We need to add and multiply the set $\{*\}$ with scalars from the ring R. Let us define these operations as follows ($*$ is treated as an unknown)

- $u + \{*\} := \{u + *\}$ and $u\{*\} := \{u*\}$, where $u \in R$.
- $c + \{a + b*\} = \{(a + c) + b*\}$ and $c\{a + b*\} = \{ac + bc*\}$, where $a, b, c \in R$.

Another important example of the $*$ notation: Let $R_0 = \mathbb{F}_p[x]/\langle \varphi(x)^k \rangle$ for a prime p and an irreducible φ mod p. In this ring, we use the notation $y = y_0 + \varphi y_1 + \cdots + \varphi^i y_i + \varphi^{i+1}*$, where $i + 1 < k$ and each $y_j \in R_0/\langle \varphi \rangle$, to denote a set $S_y \subseteq R_0$ such that

$$S_y = \{y_0 + \cdots + \varphi^i y_i + \varphi^{i+1} y_{i+1} + \cdots + \varphi^{k-1} y_{k-1} \mid \forall y_{i+1}, \ldots, y_{k-1} \in R_0/\langle \varphi \rangle\}.$$

Representative roots. Let $R = \mathbb{Z}/\langle p^k \rangle$ for a prime p. Any element in R can be written uniquely as $y = y_0 + py_1 + \cdots + p^{k-1} y_{k-1}$, where each y_j is in $\{0, \ldots, p-1\}$.

Let $g(y)$ be a polynomial in $R[y]$, then a set $y = y_0 + py_1 + \cdots + p^i y_i + p^{i+1}*$ will be called a *representative root* of g iff

- All elements in $y = y_0 + py_1 + \cdots + p^i y_i + p^{i+1}*$ are roots of g and,
- Not all elements in $y' = y_0 + py_1 + \cdots + p^{i-1} y_{i-1} + p^i *$ are roots of g.

We will sometimes represent the set of roots, $y = y_0 + py_1 + \cdots + py_i + p^{i+1}*$, succinctly as $y = v + p^{i+1}*$, where $v \in R$ stands for $y = y_0 + py_1 + \cdots + p^i y_i$. Such a pair, $(v, i+1)$, will be called a *representative pair*. We define the *length* of such a representative root to be $i+1$ and *size* to be p^{k-i-1} where $i+1 \leq k$.

We have analogous definitions for the ring $R_0 = \mathbb{F}_p[x]/\langle \varphi(x)^k \rangle$ where $\varphi \in \mathbb{F}_p[x]$ irreducible.

2.3.2 Root Finding modulo Prime Powers

Let us denote the ring $\mathbb{Z}/\langle p^k \rangle$ by R. In this section, we give an algorithm to find all the roots $y \in R$ of a polynomial $g \in R[y]$. To the best of our knowledge, the algorithm to find roots modulo p^k first appeared in Panayi's PhD thesis [Pan95]. Here, we adapt the algorithm by [BLQ13, Cor.4] ([Pan95] not available online) to find and count the roots in the form of representative roots. Recall the notation of $*$ and representative roots from Section 2.3.1.

Note that $R/\langle p^j \rangle = \mathbb{Z}/\langle p^j \rangle$, for $j \leq k$, and $R/\langle p \rangle = \mathbb{F}_p$ is the finite field of cardinality p. A root y of g in R has the following unique structure

$$y \;=\; y_0 + py_1 + p^2 y_2 + \cdots + p^{k-1} y_{k-1},$$

where each $y_j \in \{0, \ldots, p-1\}$ for all $j \in \{0, \ldots, k-1\}$.

The **output** of this algorithm is simply a set of at most $\deg g$ many representative roots of g. This bound of $\deg g$ is a curious by-product of the algorithm [BLQ13, Cor.4].

Algorithm 1 Root-finding in ring $R = \mathbb{Z}/\langle p^k \rangle$

1: **procedure** ROOT-FIND($g(y)$, p^k)

2: **If** $g(y) \equiv 0$ in $R/\langle p^k \rangle$ **return** $*$ (every element is a root).

3: Let $g(y) \equiv p^\alpha \tilde{g}(y)$ in $R/\langle p^k \rangle$, for the unique integer $0 \leq \alpha < k$ and the polynomial $\tilde{g}(y) \in R/\langle p^{k-\alpha} \rangle[y]$, such that, $\tilde{g}(y) \not\equiv 0$ in $R/\langle p \rangle$ and $\deg(\tilde{g}) \leq \deg(g)$.

4: Using randomized factoring (Theorem 2.2.1) find all the roots of $\tilde{g}(y)$ in $R/\langle p \rangle$.

5: **If** $\tilde{g}(y)$ has no root in $R/\langle p \rangle$ then **return** $\{\}$. (Dead-end)

6: Initialize $S = \{\}$.

7: **for** each root a of $\tilde{g}(y)$ in $R/\langle p \rangle$ **do**

8: Define $g_a(y) := \tilde{g}(a + py)$.

9: $S' \leftarrow$ROOT-FIND$(g_a(y),\, p^{k-\alpha})$.

10: $S \leftarrow S \cup (a + pS')$.

11: **return** S.

Note that in Step 8 we ensure: $p | g_a(y)$. This is because $g_a(y) = \tilde{g}(a + py) \equiv \tilde{g}(a) \equiv 0 \bmod p$. So, in every other recursive call to ROOT-FIND the second argument reduces by at least one. Basically, the algorithm works by reducing the problem of root finding of $g(x) \bmod p^k$ to root finding of $g_a(x) \bmod p^{k-\alpha}$ for at most $\deg(g)$-many polynomial g_a. The process repeats for each g_a and so on for k levels of recursion. This gives a rough estimate of $\deg(g)^k$ many recursions and so many representative roots in the returned set S. However, $|S| \leq \deg g$. The key reason is: The number of representative roots of g_a are upper bounded by the multiplicity of the root a of $\tilde{g}$.

The implication of Algorithm 1 is summed up in the following theorem due to [Pan95, BLQ13].

Theorem 2.3.1. *[Pan95, BLQ13] Given a univariate $g \in R[y]$ where $R = \mathbb{Z}/\langle p^k \rangle$, let $Z \subseteq R$ be the root set of $g(y)$. Then Z can be expressed as the disjoint union of at most $\deg(g)$ many representative pairs (a_0, i_0) ($a_0 \in R$ and $i_0 \in \mathbb{N}$).*

These representative pairs can be found in randomized poly$(\deg(g), k \log p)$ time.

Notice that this compact description of the root set Z allows us to calculate the size of Z too via the length of each representative root. Let us see how Algorithm 1 can be used to to factor the polynomial in the following example.

Example 8. *We have $g(x) = x^3+12x^2+3x+36$ and $p^k = 3^3$. So $g \equiv x^3+12x^2+3x+9 \bmod 27$. Using Theorem 2.2.1, the only root of $\tilde{g}(x) := g \bmod 3$ is 0. So we shift g as $g(0 + 3x) \equiv 9(x + 4) \bmod 27$.*

Dividing by 9 both sides we have, $g_a(x) \equiv x + 1 \bmod 3$ which has only root 2 modulo 3.

So we get exactly one representative root of $g \bmod 27$: $x = 0 + 3(2) + 3^2 *$.

Putting $0, 1$ and 2 in place of $*$ we get the roots $-21, -15$ and -3 modulo 27. These correspond to degree one factors $(x + 21), (x + 12)$ and $(x + 3)$ of $g \bmod 27$.

Part I

Derandomization via Ideals and Applications

Chapter 3

Introducing Split Ideals

This chapter deals with various aspects of triangular ideals which are defined in Section 3.1. Firstly in Section 3.2 we will define a special triangular ideal, called split ideals, and analyse their structure and properties. Then in Sections 3.3 and 3.4, we will see how to perform algebraic computations such as reduction, division, testing for zero-divisors and GCD modulo a triangular ideal (not necessarily a split ideal). These ideals and subroutines will be used in Chapter 4 to derandomize the root counting algorithm of [BLQ13] described in Section 2.3 of Chapter 2. These will have further applications in Chapters 5 and 6 in derandomizing the counting of basic irreducible factors, p-adic roots and computing Igusa zeta function.

3.1 Notations and Definitions

First we give some useful notations and definitions:

Monic polynomial: We call a polynomial $g \in \mathbb{F}[x_0, \ldots, x_l]$ ($\mathbb{F}$ be a ring) monic *with respect to* x_i if the leading coefficient of g with respect to x_i is 1.

We denote the ring $\mathbb{Z}/\langle p^k \rangle$ by R (ring $R/\langle p \rangle$ is the same as field $\mathbb{F}_p$). An element $a \in R$ can be seen in its p-adic representation as

$$a = a_0 + pa_1 + \ldots + p^{k-1}a_{k-1},$$

where $a_i \in \{0, \ldots, p-1\}$ for all $i \in \{0, \ldots, k-1\}$.

Tuple: A tuple of variables $(x_0, \ldots, x_l)$ will be denoted by $\bar{x}_l$. Often, an $(l+1)$-variate polynomial $a(x_0, x_1, \ldots, x_l)$ will be written as $a(\bar{x}_l)$, and the polynomial ring $\mathbb{F}[x_0, \ldots, x_l]$ as $\mathbb{F}[\bar{x}_l]$, where $\mathbb{F}$ is a ring.

Zero-Set of a polynomial: Set $\mathcal{Z}_{\mathbb{F}}(g) := \{r \in \mathbb{F} \mid g(r) \equiv 0 \text{ in } \mathbb{F}\}$ denotes the *zero-set* of a polynomial $g(x) \in \mathbb{F}[x]$ over a ring $\mathbb{F}$.

Triangular ideal: An ideal $I \subseteq \mathbb{F}[\bar{x}_l]$ is called a triangular ideal over a ring $\mathbb{F}$ if $I := \langle h_0(x_0), h_1(\bar{x}_1), \ldots, h_l(\bar{x}_l)\rangle$ where $h_i \in \mathbb{F}[\bar{x}_i]$. The length of I is defined as $l+1$ and its degree is defined as $\prod_{i=0}^{l} \deg_{x_i}(h_i)$.

For more general triangular ideals used in Groebner basis theory to perform algebraic operations, we refer to the textbook [CLO13]. Triangular ideals used here are different and more specific.

Restriction: Let $I_l \subseteq \mathbb{F}[\bar{x}_l]$ be a triangular ideal defined as $I_l := \langle h_0(x_0), h_1(\bar{x}_1), \ldots, h_l(\bar{x}_l)\rangle$. Then any ideal $I_j := \langle h_0, \ldots, h_j\rangle$, for all $0 \leq j < l$, is called its 'restriction'.

Zero-set of an ideal: Zero-set of an ideal $I \subseteq \mathbb{F}[x_0, \ldots, x_l]$, $\mathbb{F}$ be a ring, is defined as the intersection of the zero-sets of all polynomials in I,

$$\mathcal{Z}_{\mathbb{F}}(I) := \{\bar{a} = (a_0, \ldots, a_l) \in (\mathbb{F})^{l+1} \mid g(\bar{a}) \equiv 0, \forall g \in I\}.$$

We denote an ideal over R with a hat, for example $\hat{I}$, to distinguish with an ideal I defined over $\mathbb{F}_p$.

3.2 Split Ideals: Structure and Properties

In this section, we introduce our main tool: 'split ideals', which are the main content of our data structure $\mathcal{L}$ to implicitly hold the roots of given $f \bmod p^k$. It is the structure and properties of these ideals, we will study in this section, which helps us to extract, from $\mathcal{L}$, the count on the number of roots of $f \bmod p^k$.

We will be given a univariate polynomial $f(x) \in \mathbb{Z}[x]$ of degree d and a prime power p^k (for a prime p and a positive integer $k \in \mathbb{N}$). Without loss of generality, we assume that f is monic over $\mathbb{F}_p$.

We will heavily use ideals of the form $\hat{I} := \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l) \rangle$ satisfying the following condition: for any $i \in [l]$ and $\bar{a} \in \mathcal{Z}_R(\langle \hat{h}_0(x_0), \ldots, \hat{h}_{i-1}(\bar{x}_{i-1}) \rangle)$, the polynomial $\hat{h}_i(\bar{a}, x_i)$ splits completely into distinct linear factors. They are formally defined as follows:

Definition 3.2.1 (Split ideal)**.** *Given $f(x) \in R[x]$, an ideal $\hat{I} \subseteq R[\bar{x}_l]$, is called a* split ideal with respect to $f \bmod p^k$ *if,*
(1) $\hat{I}$ is a triangular ideal*, defined as $\hat{I} =: \langle \hat{h}_0(\bar{x}_0), \hat{h}_1(\bar{x}_1), \ldots, \hat{h}_l(\bar{x}_l) \rangle$, for some $0 \leq l \leq k-1$, where $\hat{h}_i(\bar{x}_i) \in R[\bar{x}_i]$ is monic with respect to x_i for all $i \in \{0, \ldots, l\}$,*
(2) $|\mathcal{Z}_{\mathbb{F}_p}(I)| = \prod_{i=0}^{l} \deg_{x_i}(h_i)$, where $h_i := \hat{h}_i \bmod p$ and $I := \langle h_0, \ldots, h_l \rangle \subseteq \mathbb{F}_p[\bar{x}_l]$, and
(3) For all $(a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I})$, $f(a_0 + pa_1 + \ldots + p^l a_l) \equiv 0 \bmod p^{l+1}$.
The length of $\hat{I}$ *is $l+1$ and its* degree *is* $\deg(\hat{I}) := \prod_{i=0}^{l} \deg_{x_i}(\hat{h}_i)$.

Split ideal $\hat{I}$ relates to possible roots of $f \bmod p^k$. Since f, p, k are fixed, we will call $\hat{I}$ a *split ideal*. The definition of split ideal enables us to precisely characterize the roots of f in $\mathbb{Z}/(p^l)$ for any $l \in \{0, \ldots, k-1\}$. Restriction of a split ideal is also a split ideal.

Lemma 3.2.2 (Restriction of a split ideal)**.** *Let $\hat{I}_l := \langle \hat{h}_0(\bar{x}_0), \ldots, \hat{h}_l(\bar{x}_l) \rangle$ be a split ideal in $R[\bar{x}_l]$, then the ideal $\hat{I}_j := \langle \hat{h}_0(\bar{x}_0), \ldots, \hat{h}_j(\bar{x}_j) \rangle$ is also a split ideal in $R[\bar{x}_j]$, for all $0 \leq j \leq l$.*

Proof. It is enough to show the lemma for $j = l - 1$. It is easy to observe that $\hat{I}_{l-1}$ is triangular. Define $h_i := \hat{h}_i \bmod p$ for all $i \in \{0, \ldots, l\}$ and $I_{l-1} := \langle h_0, \ldots, h_{l-1} \rangle$.

Looking at the second condition for being a split ideal, $|\mathcal{Z}_{\mathbb{F}_p}(I_{l-1})| \leq \prod_{i=0}^{l-1} \deg_{x_i}(h_i)$ follows because a degree $d \geq 1$ polynomial can have at most d roots in $\mathbb{F}_p$.

To show equality, notice that for any $\bar{a} = (a_0, \ldots, a_{l-1}) \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$, $\deg_{x_l}(h_l(\bar{a}, x_l))$ is bounded by $\deg_{x_l}(h_l)$. This implies $h_l(\bar{a}, x_l)$ can have at most $\deg_{x_l}(h_l)$ roots in $\mathbb{F}_p$. If $|\mathcal{Z}_{\mathbb{F}_p}(I_{l-1})| < \prod_{i=0}^{l-1} \deg_{x_i}(h_i)$ then $|\mathcal{Z}_{\mathbb{F}_p}(I_l)| < \deg_{x_l}(h_l) \cdot \prod_{i=0}^{l-1} \deg_{x_i}(h_i)$, contradicting that $\hat{I}_l$ is a split ideal.

For the third condition, since $\hat{I}_l$ is a split ideal, for any $(a_0, \ldots, a_{l-1}) \in \mathcal{Z}_R(\hat{I}_{l-1})$, $f(a_0 + pa_1 + \ldots + p^l a_l) \equiv 0 \bmod p^{l+1} \Rightarrow f(a_0 + pa_1 + \ldots + p^{l-1} a_{l-1}) \equiv 0 \bmod p^l$. □

The following corollary of Lemma 3.2.2 shows that every $\mathbb{F}_p$-zero of I_{l-1} 'induces' exactly $\deg_{x_l}(h_l)$ many $\mathbb{F}_p$-zeros of I_l.

Corollary 3.2.3. *Let $\hat{I}_l \subseteq R[\bar{x}_l]$ be a split ideal defined as $\hat{I}_l := \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\rangle$. For all $0 \leq j \leq l$, define $I_j := \langle h_0(x_0), \ldots, h_j(\bar{x}_j)\rangle$ where $h_i := \hat{h}_i \bmod p$ for all $i \in \{0, \ldots, l\}$. Then $h_0(x_0)$ splits completely over $\mathbb{F}_p$ and $h_j(\bar{a}, x_j)$ splits completely over $\mathbb{F}_p$ for all $j \in [l]$ and $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_{j-1})$.*

Lemma 3.2.4 generalises the point (2) in Definition 3.2.1.

Lemma 3.2.4. *Let $\hat{I}_l \subseteq R[\bar{x}_l]$ be a split ideal, defined as $\hat{I}_l := \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\rangle$, and $I_l := \hat{I}_l \bmod p$ be an ideal over $\mathbb{F}_p$. Then every zero $(\tilde{a}_0, \ldots, \tilde{a}_l) \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$ has a unique lift $(a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I}_l)$. In particular,*

$$|\mathcal{Z}_R(\hat{I}_l)| = \prod_{i=0}^{l} \deg_{x_i}(\hat{h}_i) = |\mathcal{Z}_{\mathbb{F}_p}(I_l)|.$$

Proof. Define $h_i := \hat{h}_i \bmod p$, for all $i \in \{0, \ldots, l\}$, so that $I_l = \langle h_0, \ldots, h_l\rangle$. Let $(\tilde{a}_0, \ldots, \tilde{a}_l) \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$ i.e., $h_0(\tilde{a}_0) = 0, h_1(\tilde{a}_0, \tilde{a}_1) = 0, \ldots, h_l(\tilde{a}_0, \ldots, \tilde{a}_l) = 0$. We prove the lemma by applying induction on the length l of the split ideal. By Corollary 3.2.3, $h_0(x_0)$ splits completely hence $\tilde{a}_0$ is a zero of multiplicity 1 of h_0. So we can apply Hensel's lemma (Lemma 2.2.2) to lift the zero $\tilde{a}_0$ of h_0 uniquely to a zero, say a_1, of $\hat{h}_0$.

By induction hypothesis, let the zero $(\tilde{a}_0, \ldots, \tilde{a}_{l-1})$ of $I_{l-1} := \langle h_0, \ldots, h_{l-1}\rangle$ lifts uniquely to a zero, say $(a_0, \ldots, a_{l-1})$, of $\hat{I}_{l-1} := \langle \hat{h}_0, \ldots, \hat{h}_{l-1}\rangle$. By Corollary 3.2.3, $\tilde{a}_l$ must be a multiplicity 1 zero of the univariate $h_l(\tilde{a}_0, \ldots, \tilde{a}_{l-1}, x_l)$. So by Hensel's lemma (Lemma 2.2.2), $\tilde{a}_l$ lifts 'uniquely' to a zero, say a_l, of $\hat{h}_l(a_0, \ldots, a_{l-1}, x_l)$. This proves the first part of the lemma.

Now we will show that $|\mathcal{Z}_R(\hat{I}_l)| = \prod_{i=0}^{l} \deg_{x_i}(\hat{h}_i) = |\mathcal{Z}_{\mathbb{F}_p}(I_l)|$. We know that any zero of $\hat{I}_l$ is a lift of a unique zero of I_l (just reduce mod p) and any zero of I_l lifts uniquely to a zero of $\hat{I}_l$, hence $|\mathcal{Z}_R(\hat{I}_l)| = |\mathcal{Z}_{\mathbb{F}_p}(I_l)|$. By Definition 3.2.1, each $\hat{h}_i(\bar{x}_i)$ is a monic polynomial, so $\deg_{x_i}(\hat{h}_i) = \deg_{x_i}(h_i)$. Thus $|\mathcal{Z}_{\mathbb{F}_p}(I_l)| = \prod_{i=0}^{l} \deg_{x_i}(\hat{h}_i)$. This proves the lemma. □

Lemma 3.2.5 shows that a split ideal $\hat{I}$ can be decomposed into an intersection of ideals of the form $\hat{I}_{\bar{a}} := \langle x_0 - a_0, \ldots, x_l - a_l\rangle$, where $\bar{a} =: (a_0, \ldots, a_l)$ is a root of $\hat{I}$.

To prove this structural lemma, following proposition is useful.

Proposition 3. *Let $a_1, \ldots, a_n \in R$ such that $a_i \neq a_j \bmod p$ for $i \neq j$ and $i, j \in [n]$. Then,*

$$\langle \prod_{i=1}^{n} (x - a_i) \rangle = \bigcap_{i=1}^{n} \langle x - a_i \rangle.$$

Proof. We prove the lemma for the case $n = 2$ as the proof of the generalised form is similar via induction.

Define the ideals $I := \langle (x - a_1)(x - a_2) \rangle$ and $I_{a_i} := \langle x - a_i \rangle$ for $i \in [2]$. It is easy to see that if $h \in I$ then $h \in I_{a_1}$ and $h \in I_{a_2}$.

Let $h \in I_{a_1} \cap I_{a_2}$. Let $\tilde{h}(x) := h(x) \bmod p$ such that $\tilde{h}(x) =: (x - \tilde{a}_1)^{e_1} \cdot (x - \tilde{a}_2)^{e_2} \cdot \tilde{g}_3(x)$ where $\tilde{a}_1 \equiv a_1 \bmod p$ and $\tilde{a}_2 \equiv a_2 \bmod p$ and $\tilde{g}_3$ is co-prime to both $(x - \tilde{a}_1)^{e_1}$ and $(x - \tilde{a}_2)^{e_2}$. Applying Hensel's lemma (Lemma 2.2.2), we get $h(x) =: g_1(x) \cdot g_2(x) \cdot g_3(x)$ where $g_1 \equiv (x - \tilde{a}_1)^{e_1} \bmod p$, $g_2 \equiv (x - \tilde{a}_2)^{e_2} \bmod p$ and $g_3 \equiv \tilde{g}_3 \bmod p$. Hensel's lemma (Lemma 2.2.2) also says that this is unique co-prime factorisation of $h(x)$ (g_1, g_2 and g_3 are mutually co-prime). So if $h_1 \in R[x]$ is an irreducible factor of h then h_1 divides exactly one of the g_1, g_2 and g_3. Hence, $(x - a_1)|h \Rightarrow (x - a_1)|g_1$ (since $(x - a_1)$ does not divide g_2 or g_3 modulo p). Similarly, $(x - a_2)|g_2$ which implies $(x - a_1)(x - a_2)|h$. Thus $h \in I$ proving the lemma. □

Now, we can show that a split ideal $\hat{I}$ can be decomposed in terms of its zeros.

Lemma 3.2.5 (Split ideal structure)**.** *Let $\hat{I} := \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l) \rangle$ be a split ideal of $R[\bar{x}_l]$. Let $h_i := \hat{h}_i(\bar{x}_i) \bmod p$, for all $i \in \{0, \ldots, l\}$, and $I := \langle h_0, \ldots, h_l \rangle$ be an ideal of $\mathbb{F}_p[\bar{x}_l]$. Then,*

1. *$I = \bigcap_{\bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(I)} I_{\bar{b}}$ where $I_{\bar{b}} := \langle x_0 - b_0, \ldots, x_l - b_l \rangle$ corresponds to $\bar{b} =: (b_0, \ldots, b_l) \in \mathcal{Z}_{\mathbb{F}_p}(I)$ and,*
2. *$\hat{I} = \bigcap_{\bar{a} \in \mathcal{Z}_R(\hat{I})} \hat{I}_{\bar{a}}$ where $\hat{I}_{\bar{a}} := \langle x_0 - a_0, \ldots, x_l - a_l \rangle$ corresponds to $\bar{a} =: (a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I})$.*

Proof. We will prove only Part (2) of the lemma as proving Part (1) is similar and easier. We will prove this decomposition by applying induction on the length of the split ideal. For the base case, length of $\hat{I}$ is 1 and $\hat{I} = \langle \hat{h}_0(x_0) \rangle \subseteq R[x_0]$. By Corollary 3.2.3, $h_0(x_0) = \prod_{i=1}^{\deg(h_0)} (x_0 - \tilde{a}_{0,i})$ for *distinct* $\tilde{a}_{0,i} \in \mathbb{F}_p$. By Hensel's Lemma (Lemma 2.2.2) and Lemma 3.2.4 , $\hat{h}_0(x_0) = \prod_{i=1}^{\deg_{x_0}(\hat{h}_0)} (x - a_{0,i})$ where $a_{0,i} \equiv \tilde{a}_{0,i} \bmod p$. So, $\hat{I} = \bigcap_{a_{0,i} \in \mathcal{Z}_R(\hat{I})} \hat{I}_{a_{0,i}}$ by Proposition 3.

Let $\hat{I}$ be a split ideal of length $l+1$, $\hat{I} = \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\rangle \subseteq R[\bar{x}_l]$. Define ideal $\hat{J} := \langle \hat{h}_0, \ldots, \hat{h}_{l-1}\rangle$. By Lemma 3.2.2, $\hat{J}$ is a split ideal. From the induction hypothesis, we have $\hat{J} = \bigcap_{\bar{a}\in\mathcal{Z}_R(\hat{J})} \hat{J}_{\bar{a}}$, where $\hat{J}_{\bar{a}} := \langle x_0 - a_0, \ldots, x_{l-1} - a_{l-1}\rangle$ for a zero $\bar{a} =: (a_0, \ldots, a_{l-1})$ of $\hat{J}$. We know that,

$$\hat{I} \;=\; \hat{J} + \langle \hat{h}_l(\bar{x}_l)\rangle \;=\; \bigcap_{\bar{a}\in\mathcal{Z}_R(\hat{J})} \hat{J}_{\bar{a}} + \langle \hat{h}_l(\bar{x}_l)\rangle \;=\; \bigcap_{\bar{a}\in\mathcal{Z}_R(\hat{J})} \left(\hat{J}_{\bar{a}} + \langle \hat{h}_l(\bar{a}, x_l)\rangle\right). \tag{3.1}$$

By Definition 3.2.1, $\hat{h}_l$ is monic with respect to x_l and so $\deg(\hat{h}_l(\bar{a}, x_l)) = \deg_{x_l}(\hat{h}_l) = \deg(h_l(\bar{a}, x_l))$ for all $\bar{a} \in \mathcal{Z}_R(\hat{J})$. By Corollary 3.2.3, $h_l(\bar{a}, x_l)$ splits completely over $\mathbb{F}_p$ and so by Hensel's lemma (Lemma 2.2.2), $\hat{h}_l(\bar{a}, x_l)$ splits completely over R as $\hat{h}_l(\bar{a}, x_l) =: \prod_{i=1}^{\deg_{x_l}(h_l)}(x_l - a_{l,i})$. Thus by Proposition 3, $\langle \hat{h}_l(\bar{a}, x_l)\rangle = \bigcap_{i=1}^{\deg_{x_l}(h_l)}\langle x_l - a_{l,i}\rangle$. So, for any $\bar{a} \in \mathcal{Z}_R(\hat{J})$, $\hat{J}_{\bar{a}} + \langle h_l(\bar{a}, x_l)\rangle = \hat{J}_{\bar{a}} + \bigcap_{i=1}^{\deg_{x_l}(h_l)}\langle x_l - a_{l,i}\rangle = \bigcap_{i=1}^{\deg_{x_l}(\hat{h}_l)} \hat{I}_{\bar{a},a_{l,i}}$, where $(\bar{a}, a_{l,i})$ are the roots of $\hat{I}$ induced from $\bar{a}$. Hence from Eqn. 3.1, $\hat{I} = \bigcap_{\bar{b}\in\mathcal{Z}_R(\hat{I})} \hat{I}_{\bar{b}}$.

This finishes the inductive proof, completely factoring $\hat{I}$. □

Let $\hat{I} =: \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\rangle$ be a split ideal. Suppose some $\hat{h}_i$ factors as $\hat{h}_i(\bar{x}_i) = \hat{h}_{i,1}(\bar{x}_i)\cdots\hat{h}_{i,m}(\bar{x}_i)$. Define $\hat{I}_j := \langle \hat{h}_0(x_0), \ldots, \hat{h}_{i-1}(\bar{x}_{i-1}), \hat{h}_{i,j}(\bar{x}_i), \hat{h}_{i+1}(\bar{x}_{i+1}), \ldots, \hat{h}_l(\bar{x}_l)\rangle$, for $j \in [m]$. The following corollary of Lemma 3.2.5 is evident because root-sets of $\hat{I}_j$ partition the root-set of $\hat{I}$.

Corollary 3.2.6 (Splitting split ideals)**.** *Let $\hat{I} =: \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\rangle$ be a split ideal of $R[\bar{x}_l]$. Let some $\hat{h}_i(\bar{x}_i)$ factor as $\hat{h}_i(\bar{x}_i) = \hat{h}_{i,1}(\bar{x}_i)\cdots\hat{h}_{i,m}(\bar{x}_i)$. Then,*

$$\hat{I} = \bigcap_{j=1}^{m} \hat{I}_j,$$

where each $\hat{I}_j := \langle \hat{h}_0(x_0), \ldots, \hat{h}_{i-1}(\bar{x}_{i-1}), \hat{h}_{i,j}(\bar{x}_i), \hat{h}_{i+1}(\bar{x}_{i+1}), \ldots, \hat{h}_l(\bar{x}_l)\rangle$ is a split ideal.

Our list data structure $\mathcal{L}$ actually contains a list of special split ideals, called maximal split ideals, that partition the roots of f in $\mathbb{Z}/(p^k)$.

Definition 3.2.7 (Maximal Split Ideal)**.** *We call a split ideal $\hat{I}_l := \langle \hat{h}_0, \ldots, \hat{h}_l\rangle$ to be* maximal split ideal *if,*

1) for all $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I}_l)$, $g(x) := f(a_0 + pa_1 + \ldots + p^l a_l + p^{l+1}x)$ vanishes identically

mod p^k,

2) the restriction $\hat{I}_{l-1} := \langle \hat{h}_0, \ldots, \hat{h}_{l-1} \rangle$ *does not follow the previous condition.*

Lemma 3.2.8 shows that a root of a maximal split ideal represents a set of roots of f mod p^k and provides the size of that set.

Lemma 3.2.8 (Roots represented by a root of maximal split ideal)**.** *Let* $\hat{I}$ *be a maximal split ideal of length* $l+1$*, then a zero* $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I})$ *maps to ('represents') exactly* p^{k-l-1} *zeros of* f *in* $\mathcal{Z}_R(f)$*. In particular,* $\hat{I}$ *represents exactly* $\deg(\hat{I}) \cdot p^{k-l-1}$ *zeros of* f mod p^k*.*

Proof. By Definition 3.2.7 of a maximal split ideal, for any $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I})$, $g(x) \equiv 0 \bmod p^k$ where $g(x) := f(a_0 + pa_1 + p^2a_2 + \ldots + p^la_l + p^{l+1}x)$. In other words, $g(x)$ vanishes identically over R irrespective of any x in R. The distinct choices of x in R are p^k but $p^{l+1}x$ has only p^{k-l-1} distinct choices in R. The reason is as follows:

Any $b \in R$ can be written 'uniquely' in p-adic representation as $b = b_0 + pb_1 + \cdots + p^{k-1}b_{k-1}$ where each $b_i \in \{0, \ldots, p-1\}$. Hence $p^{l+1} \cdot b \equiv p^{l+1}(b_0 + pb_1 + \cdots + p^{k-l-2}b_{k-l-2}) \bmod p^k$ takes distinct values for distinct choices of $b_0, \ldots, b_{k-l-2}$ which are p^{k-l-1}. By Lemma 3.2.4 and Definition 3.2.1, $|\mathcal{Z}_R(\hat{I})| = \deg(\hat{I})$ so $\hat{I}$ represents exactly $\deg(\hat{I}) \cdot p^{k-l-1}$ zeros of f mod p^k. □

3.3 Reduction and Division modulo a Triangular Ideal

We will show that it is efficient to reduce a polynomial $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ modulo a triangular ideal $J_l = \langle b_0(x_0), b_1(\bar{x}_1), \ldots, b_l(\bar{x}_l) \rangle \subseteq \mathbb{G}[\bar{x}_l]$, where $\mathbb{G}$ is any Galois ring (in particular, $R = \mathbb{Z}/p^k$, or $\mathbb{F}_p$).

Note that J_l need not be a split ideal, though the algorithms of this section work for split ideals as they are triangular by Definition 3.2.1.

Assumptions: In the generators of the triangular ideal we assume $\deg_{x_i} b_i(\bar{x}_i) \geq 2$ (for $0 \leq i \leq l$). Otherwise, we could eliminate variable x_i and work with fewer variables (& smaller length triangular ideal). Additionally, each $b_i(\bar{x}_i)$ (for $0 \leq i \leq l$) is monic (leading coefficient is 1 with respect to x_i), and presented in a *reduced* form modulo the prior triangular ideal $J_{i-1} := \langle b_0(\bar{x}_0), \ldots, b_{i-1}(\bar{x}_{i-1}) \rangle \subseteq \mathbb{G}[\bar{x}_{i-1}]$.

Let us first define reduction modulo an ideal (assume $\mathbb{G}$ to be the Galois ring $G(p^k, b)$).

Definition 3.3.1 (Reduction by a triangular ideal)**.** *The* reduction *of a multivariate polynomial* $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ *by a triangular ideal* $J_l = \langle b_0(\bar{x}_0), \ldots, b_l(\bar{x}_l)\rangle \subseteq \mathbb{G}[\bar{x}_l]$ *is the unique polynomial* $\tilde{a}(\bar{x}_l) \equiv a(\bar{x}_l) \bmod J_l$*, where* $\deg_{x_i}(\tilde{a}) < \deg_{x_i}(b_i)$*, for all* $i \in \{0, \ldots, l\}$.

Idea of reduction: The idea behind the algorithm is inspired from the univariate reduction. If $l = 0$, then reduction of $a(x_0)$ modulo $b_0(x_0)$ is simply the remainder of the division of a by b_0 in the underlying polynomial ring $\mathbb{G}[x_0]$. For a larger l, the reduction of $a(\bar{x}_l)$ modulo the triangular ideal $J_l = \langle b_0(x_0), \ldots, b_l(\bar{x}_l)\rangle$ is the remainder of the division of $a(\bar{x}_l)$ by $b_l(\bar{x}_l)$ in the polynomial ring $(\mathbb{G}[x_0, \ldots, x_{l-1}]/J_{l-1})[x_l]$. The fact that b_l is monic, helps in generalizing 'long division' (Steps 8-12 in Algorithm 2 below).

Input: An $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ and a triangular ideal $J_l = \langle b_0(\bar{x}_0), \ldots, b_l(\bar{x}_l)\rangle \subseteq \mathbb{G}[\bar{x}_l]$.

Output: Reduction $\tilde{a}$ of a mod J_l as defined above.

Algorithm 2 Reduce $a(\bar{x}_l)$ modulo J_l

1: **procedure** REDUCE($a(\bar{x}_l)$, J_l)
2: **if** $l = 0$ **then**
3: [**Reduce** $a(x_0)$ **by** $b_0(x_0)$] return remainder of univariate division of a by b_0 in $\mathbb{G}[x_0]$.
4: $d_l(a) \leftarrow \deg_{x_l}(a)$ and $d_l(b) \leftarrow \deg_{x_l}(b_l)$.
5: Let $a(\bar{x}_l) =: \Sigma_{i=0}^{d_l(a)} a_i(\bar{x}_{l-1}) x_l^i$ be the polynomial representation of $a(\bar{x}_l)$ with respect to x_l.
6: Recursively reduce each coefficient $a_i(\bar{x}_{l-1})$ of a modJ_{l-1}:
$\tilde{a}_i(\bar{x}_{l-1}) \leftarrow$ REDUCE($a_i(\bar{x}_{l-1})$, J_{l-1}), for all $i \in \{0, \ldots, d_l(a)\}$.
7: **while** $d_l(a) \geq d_l(b)$ **do**
8: $a(\bar{x}_l) \leftarrow a - \left(a_{d_l(a)} \cdot x_l^{d_l(a)-d_l(b)} \cdot b_l\right)$
9: Update $d_l(a) \leftarrow \deg_{x_l}(a)$. Update a_i's such that $a(\bar{x}_l) =: \Sigma_{i=0}^{d_l(a)} a_i(\bar{x}_{l-1}) \cdot x_l^i$.
10: Call REDUCE($a_i(\bar{x}_{l-1})$, J_{l-1}) for all $i \in \{0, \ldots, d_l(a)\}$: recursively reduce each coefficient $a_i(\bar{x}_{l-1})$ mod J_{l-1} (like Step 7).
11: **return** $a(\bar{x}_l)$.

Following lemma shows that reduction modulo a triangular ideal (Algorithm 2) is efficient.

Lemma 3.3.2 (Reduction)**.** *Given* $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ *and* $J_l \subseteq \mathbb{G}[\bar{x}_l]$ *defined as* $J_l := \langle b_0(x_0), \ldots, b_l(\bar{x}_l)\rangle$ *with each* b_i *monic with respect to* x_i*. Define* $J_{l-1} := \langle b_0, \ldots, b_{l-1}\rangle$*. If* $\deg_{x_i}(a) \leq 2\deg_{x_i}(b_i)$ *for each* $i \in \{0, \ldots, l-1\}$ *then* REDUCE*(*$a(\bar{x}_l)$*,* J_l*) takes time* $\tilde{O}(\deg_{x_l}(a)^2 \cdot \deg(J_{l-1})^4 \cdot \log|\mathbb{G}|)$.

Proof. Let $T(d, l)$ denotes the time taken by Algorithm 2 to reduce a degree d polynomial (with respect to x_l) modulo J_l. Let $d_i(g) = \deg_{x_i}(g)$ for all $i \in \{0, \ldots, l\}$. Then we get the recurrence,

$$T(d_l(a), l) = d_l(a)^2 \cdot T(2d_{l-1}(b_{l-1}), l-1) + \tilde{O}(d_l(a)^2 \cdot \deg(J_{l-1})^2 \cdot \log|\mathbb{G}|). \tag{3.2}$$

Let us first see how we got the recurrence.

Firstly, Step-9 takes time (including the while loop) $\tilde{O}(d_l(a)^2 \cdot \deg(J_{l-1})^2 \cdot \log|\mathbb{G}|)$. This is because the while loop runs $d_l(a)$ times and $a_{d_l(a)}(\bar{x}_{l-1})$ is multiplied to $d_l(b_l)$ coefficients (with respect to x_l) of b_l. So there are in total $d_l(a) \cdot d_l(b_l) \leq d_l(a)^2$ multiplications (safely assuming $d_l(a) \geq d_l(b_l)$), each taking time $(\prod_{i=0}^{l-1} d_i(a_{d_l(a)}) \cdot d_i(b_l)) \cdot \tilde{O}(\log|\mathbb{G}|) = \tilde{O}((\prod_{i=0}^{l-1} d_i(a_{d_l(a)}) \cdot d_i(b_i)) \cdot \log|\mathbb{G}|)$ where $\tilde{O}(\log|\mathbb{G}|)$ comes from ring operations (addition/multiplication/division) in $\mathbb{G}$ (refer to [Sho09]). At Step-9, $a_{d_l(a)}$ is already reduced modulo J_{l-1} (due to Steps 7 and 11), so we have $d_i(a_{d_l(a)}) < d_i(b_i)$. Thus multiplications at Step-9 (including the while loop) takes time $\tilde{O}(d_l(a)^2 \cdot \deg(J_{l-1})^2 \cdot \log|\mathbb{G}|)$.

After multiplication at Step-9 we have $d_i(a) < 2d_i(b_i)$, for each $i \in \{0, \ldots, l-1\}$, at Step-10. So the calls to REDUCE at Step-11 subsumes the calls to REDUCE at Step-7. There are $d_l(a)^2$ such calls (including the loop) each taking time at most $T(2d_{l-1}(b_{l-1}), l-1)$. This gives us Recurrence 3.2.

Now we will use induction on l to show that $T(d_l(a), l) = \tilde{O}(\deg_{x_l}(a)^2 \cdot \deg(J_{l-1})^4 \cdot \log|\mathbb{G}|)$.

When $l = 0$, Algorithm 2 performs a univariate division at Step-3. Thus $T(d_0(a), 0) = d_0(a) \cdot d_0(b_0) \cdot \tilde{O}(\log|\mathbb{G}|) = \tilde{O}(d_0(a)^2 \cdot \log|\mathbb{G}|)$.

When $l = 1$, $T(d_1(a), 1) = d_1(a)^2 \cdot T(2d_0(b_0), 0) + \tilde{O}(d_1(a)^2 \cdot \deg(J_0)^2 \cdot \log|\mathbb{G}|)$. Now, $T(2d_0(b_0), 0) = \tilde{O}((2d_0(b_0))^2 \cdot \log|\mathbb{G}|) = \tilde{O}(d_0(b_0)^4 \cdot \log|\mathbb{G}|)$ since $d_0(b_0) \geq 2$. Thus $T(d_1(a), 1) = \tilde{O}(d_1(a)^2 \cdot \deg(J_0)^4 \cdot \log|\mathbb{G}|)$ as $\deg(J_0) = d_0(b_0)$.

By induction hypothesis, $T(2d_{l-1}(b_{l-1}), l-1) = \tilde{O}((2d_{l-1}(b_{l-1}))^2 \cdot \deg(J_{l-2})^4 \cdot \log|\mathbb{G}|) = \tilde{O}(\deg(J_{l-1})^4 \cdot \log|\mathbb{G}|)$ as $2 \leq d_{l-1}(b_{l-1})$ and $\deg(J_{l-1}) = d_{l-1}(b_{l-1}) \cdot \deg(J_{l-2})$. Substituting

value of $T(2d_{l-1}(b_{l-1}), l-1)$ in Equation 3.2 we get, $T(d_l(a), l) = \tilde{O}(d_l(a)^2 \cdot \deg(J_{l-1})^4 \cdot \log|\mathbb{G}|)$. □

Lemma 3.3.3 (Division modulo triangular ideal)**.** *Given a triangular ideal $J_l \subseteq \mathbb{G}[\bar{x}_l]$ defined as $J_l := \langle b_0(x_0), \ldots, b_l(\bar{x}_l)\rangle$ with each b_i monic with respect to x_i. If $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]/J_l$ is a unit then we can compute $a^{-1} \bmod J_l$, in reduced form, in time $\tilde{O}(\deg(J_l)^4 \cdot \log|\mathbb{G}|)$.*

Proof. Let $u(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]/J_l$ be such that $u \cdot a \equiv 1 \bmod J_l$. We can write u as

$$\sum_{\substack{\bar{e} \geq \bar{0} \\ \forall\ 0 \leq i \leq l,\ e_i < \deg_{x_i}(b_i)}} u_{\bar{e}} \cdot \bar{x}_l^{\bar{e}}\ .$$

We want to find the unknowns $u_{\bar{e}}$ in $\mathbb{G}$, satisfying $u \cdot a \equiv 1 \bmod J_l$. First we compute the product $u \cdot a =: c$ (where $\deg_{x_i}(c) \leq 2\deg_{x_i}(b_i)$) and then reduce, using Algorithm 2, $c \bmod J_l$ (now $\deg_{x_i}(c) \leq \deg_{x_i}(b_i)$). This gives us a linear system in the unknowns of size $n \times n$ where $n = O(\deg(J_l))$. Since there exists a unique u, our linear system is efficiently solvable (in $\deg(J_l)$) by standard linear algebra.

The time taken is dominated by the time to reduce $c \bmod J_l$ where $\deg_{x_i}(c) \leq 2\deg_{x_i}(b_i)$. Using Lemma 3.3.2, it takes time $\tilde{O}((2\deg_{x_l}(b_l))^2 \cdot \deg(J_{l-1})^4 \cdot \log|\mathbb{G}|) = \tilde{O}(\deg(J_l)^4 \cdot \log|\mathbb{G}|)$ as $2 \leq \deg_{x_l}(b_l)$ and $\deg(J_l) = \deg_{x_l}(b_l) \cdot \deg(J_{l-1})$.

□

3.4 Testing for Zerodivisors and GCD Computation

In this section we will work over a finite field $\mathbb{F}_q$ instead of general Galois ring $\mathbb{G}$. Similar to Section 3.3, we will work with a triangular ideal $I_l \subseteq \mathbb{F}_q[\bar{x}_l]$ defined as $I_l := \langle h_0(x_0), \ldots, h_l(\bar{x}_l)\rangle$ where each h_i is assumed to be monic with respect to x_i and reduced modulo $I_{i-1} := \langle h_0, \ldots, h_{i-1}\rangle$.

Test-Zero-Div$(a(\bar{x}_l), I_l)$, for the triangular ideal I_l, either reports that $a(\bar{x}_l)$ is not a zerodivisor modulo I_l, or *returns a non-trivial factorization* of some generator $h_i =: h_{i,1} \cdots h_{i,m}$ (into monic, with respect to x_i, factors modulo the prior ideal $I_{i-1} := \langle h_0, \ldots, h_{i-1}\rangle$).

Idea and Overview: The main idea is inspired from the univariate case: if $a(x) \in \mathbb{F}_q[x]$

is a zero divisor modulo $h(x) \in \mathbb{F}_q[x]$ then a and h must share a non-trivial common factor. Generalising this, Algorithm 3 is based on the following idea:

In the quotient ring $\mathbb{F}_q[\bar{x}_l]/\langle I_l \rangle$, a monic (with respect to x_i) polynomial $a(\bar{x}_i)$ is a zero-divisor iff it contains a non-trivial factor, with respect to x_i, of the generator $h_i(\bar{x}_i)$ modulo the triangular ideal I_{i-1}.

So, firstly the algorithm 'recursively' checks if the given polynomial $a(\bar{x}_l)$ is monic with respect to x_l. While in recursion if it fails, it outputs $True$ and a non-trivial factorization of some generator h_i for $i < l$. After making $a(\bar{x}_l)$ monic the algorithm takes the gcd of a and h_l: similar to taking univariate gcd with respect to x_l over the coefficient ring $\mathbb{F}_q[\bar{x}_{l-1}]/I_{l-1}$. if it finds non-trivial gcd it factors h_l, else $a(\bar{x}_l)$ is not a zero-divisor.

Algorithm 3 Zerodivisor test of $a(\bar{x}_l)$ modulo I_l

1: **procedure** TEST-ZERO-DIV($a(\bar{x}_l)$,I_l)

[Steps 2 to 9 handle base case. A zero-divisor $a(x_0)$ in $\mathbb{F}_q[x_0]/\langle h_0(x_0) \rangle$ must share a factor with $h_0(x_0)$.]

2: **if** $l = 0$ **then**

3: $gcd \leftarrow \gcd(a(x_0), h_0(x_0))$. [Taking univariate GCD]

4: **if** gcd is non-trivial **then**

5: Factorize $h_0(x_0) =: gcd \cdot \frac{h_0}{gcd}$; **return** $(True, gcd, \frac{h_0}{gcd})$.

6: **else**

7: **return** $(False)$.

[Steps 10 to 15 either make the given $a(\bar{x}_l)$ monic, with respect to x_l, or the algorithm ends factoring a generator $h_i(\bar{x}_i)$ of I_l.]

8: Let the leading coefficient of $a(\bar{x}_l)$ with respect to x_l be $\tilde{a}(\bar{x}_{l-1})$.

9: Test if the leading coefficient of $a(\bar{x}_l)$ is a unit by calling TEST-ZERO-DIV($\tilde{a}(\bar{x}_{l-1})$, I_{l-1}).

10: **if** The test returned $True$ **then**

11: **return** $(True, h_{i,1}, \ldots, h_{i,m})$; where a generator $h_i(\bar{x}_i)$ of I_l factors as $h_i =: h_{i,1} \cdots h_{i,m}$.

12: Compute $1/\tilde{a} \bmod I_{l-1}$ (using Lemma 3.3.3) and update $a \leftarrow (1/\tilde{a}) \cdot a \bmod I_l$.

[Steps 17-25 take the gcd of a and h_l, with respect to x_l, using iterated division method (Euclid's method).]

13: Define $b(\bar{x}_l) \leftarrow h_l(\bar{x}_l)$.

14: **while** $b(\bar{x}_l) \neq 0$ **do**

[Steps 18-22 make $b(\bar{x}_l)$ monic, with respect to x_l, to make reduction at Step 23 possible.]

15: Let $\tilde{b}(\bar{x}_{l-1})$ be the leading coefficient of $b(\bar{x}_l)$ with respect to x_l.

16: **if** TEST-ZERO-DIV($\tilde{b}(\bar{x}_{l-1})$, I_{l-1}) $= True$ **then**

17: **return** ($True$, a non-trivial factorization of some generator $h_i(\bar{x}_i)$).

18: Compute $1/\tilde{b}$ mod I_{l-1} (using Lemma 3.3.3) and update $b \leftarrow (1/\tilde{b}) \cdot b$ mod I_l. [b is now monic.]

19: Let $c(\bar{x}_l) \leftarrow$ REDUCE($a(\bar{x}_l)$, $I_{l-1} + \langle b(\bar{x}_l) \rangle$) (same as taking remainder of $a(\bar{x}_l)$ when divided by the monic polynomial $b(\bar{x}_l)$ modulo I_{l-1}).

20: $a(\bar{x}_l) \leftarrow b(\bar{x}_l)$, $b(\bar{x}_l) \leftarrow c(\bar{x}_l)$.

[The loop ends eventually as $\deg_{x_l}(b)$ falls with each iteration.]

[Gcd of original $a(\bar{x}_l)$ and $h_l(\bar{x}_l)$ mod I_l is stored in $a(\bar{x}_l)$.]

21: **if** gcd $a(\bar{x}_l)$ is non-trivial **then**

22: **return** ($True$, a non-trivial factorization of $h_l(\bar{x}_l)$).

23: **else**

24: **return** ($False$). [$a(\bar{x}_l)$ is not a zerodivisor.]

Lemma 3.4.1 (Efficiency of testing zero-divisors)**.** *Assuming $a(\bar{x}_l)$ is reduced modulo I_l, Algorithm 3 takes time $\tilde{O}(\deg(I_l)^4 \cdot \log|\mathbb{F}_q|)$.*

Proof. Denote $\deg_{x_i}(h_i)$ by d_i. Since $a(\bar{x}_l)$ is reduced modulo I_l, we have $\deg_{x_i}(a) \leq d_i$ for all $i \in \{0, \ldots, l\}$. Let $T(l)$ denote the time taken by the call TEST-ZERO-DIV($a(\bar{x}_l)$, I_l).

The time taken to compute gcd of a and h_l i.e., the while loop (Steps 17-25) subsumes the time taken by all other major steps. Since the while loop runs at most d_l times (as $\deg_{x_l}(a) \leq d_l$ and $\deg_{x_l}(b) \leq d_l$), we get the recurrence

$$T(l) = d_l \cdot (T(l-1) + \tilde{O}(d_l^2 \cdot \deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)).$$

The term $\tilde{O}(d_l^2 \cdot \deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)$ in the recurrence comes from the time taken by Steps 22 and 23. Computing inverse at Step 22 takes time $\tilde{O}(\deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)$ by Lemma 3.3.3. Update $b \leftarrow (1/\tilde{b}) \cdot b \bmod I_l$ takes $\deg_{x_l}(b) \leq d_l$ reductions modulo I_{l-1} which by Lemma 3.3.2 takes time $d_l \cdot \tilde{O}(\deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)$. The call REDUCE$(a(\bar{x}_l),\ I_{l-1} + \langle b(\bar{x}_l)\rangle)$ at Step 23 takes time $\tilde{O}(d_l^2 \cdot \deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)$ (by Lemma 3.3.2). Hence we get the desired recurrence $T(l) = d_l \cdot T(l-1) + \tilde{O}(d_l^3 \cdot \deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)$.

To prove the time complexity we will apply the induction on l. It is easy to see the base case $l = 0$ which is just a univariate gcd. By induction hypothesis, we have $T(l-1) = \tilde{O}(\deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|)$. Substituting this in the recurrence we get $T(l) = \tilde{O}(d_l^3 \cdot \deg(I_{l-1})^4 \cdot \log|\mathbb{F}_q|) = \tilde{O}(\deg(I_l)^4 \cdot \log|\mathbb{F}_q|)$ as $\deg(I_l) = d_l \cdot \deg(I_{l-1})$. □

GCD$(a(\bar{x}_l, x),\ b(\bar{x}_l, x),\ I_l)$ either successfully computes a 'monic' polynomial $g(\bar{x}_l, x)$ which is the gcd of the polynomials $a(\bar{x}_l, x)$ and $b(\bar{x}_l, x)$ modulo the triangular ideal $I_l = \langle h_0(x_0), \ldots, h_l(\bar{x}_l)\rangle$ or returns *False* and outputs a *non-trivial factorization* of some generator $h_i(\bar{x}_i)$ in case it fails to compute the gcd.

Algorithm 4 GCD computation modulo I_l

1: **procedure** GCD$(a(\bar{x}_l, x),\ b(\bar{x}_l, x),\ I_l)$
2: Let $\tilde{b}(\bar{x}_l)$ be the leading coefficient of b with respect to x.
3: **if** TEST-ZERO-DIV$(\tilde{b}(\bar{x}_l),\ I_l) = True$ **then**
4: **return** *False*, TEST-ZERO-DIV$(\tilde{b}(\bar{x}_l),\ I_l)$ factors some generator $h_i(\bar{x}_i)$.
5: Let $c(\bar{x}_l, x) \leftarrow$REDUCE$(a,\ I_l + \langle b/\tilde{b}\rangle)$.
6: **if** $c = 0$ **then**
7: **return** $b/\tilde{b}$.
8: **else**
9: **return** GCD$(b(\bar{x}_l, x),\ c(\bar{x}_l, x),\ I_l)$.

Lemma 3.4.2 (GCD modulo a triangular ideal)**.** *Algorithm 4 either factors a generator h_i (& outputs False), or computes a monic polynomial $g(\bar{x}_l, x) \in \mathbb{F}_q[\bar{x}_l, x]$, such that, g divides a, b modulo I_l. Moreover, $g = ua + vb \bmod I_l$, for some $u(\bar{x}_l, x), v(\bar{x}_l, x) \in \mathbb{F}_q[\bar{x}_l, x]$.*

Let a and b are in reduced form modulo I_l and $\deg_x(a) \geq \deg_x(b)$ then Algorithm 4 takes

time $\tilde{O}(\deg_x(a)^3 \cdot \deg(I_l)^4 \cdot \log|\mathbb{F}_q|)$.

Proof. Algorithm 4 is just an implementation of univariate Euclidean gcd algorithm, with respect to x, over the coefficient ring $\mathbb{F}_q[\bar{x}_l]/I_l =: R'$. If the algorithm outputs $g(\bar{x}_l, x) \in R'[x]$ then, by standard Euclidean gcd arguments (using recursion), there exists $u(\bar{x}_l, x), v(\bar{x}_l, x) \in R'[x]$, such that, $ua + vb = g$, and g divides both a and b modulo I_l.

The algorithm works fine if in each step it was able to work with a monic divisor. Otherwise, it gets stuck at a 'division' step, implying that the divisor's leading-coefficient is a zero-divisor, factoring some generator of I_l.

For time complexity, each recursive step makes one call each to Test-Zero-Div, Reduce, and division procedures. They take time $\tilde{O}(\deg_x(a)^2 \cdot \deg(I_l)^4 \cdot \log|\mathbb{F}_q|)$ ($\because$ coefficients of a and b are in reduced form mod I_l, $\deg_x(a)$ is assumed to be at least $\deg_x(b)$, and use Lemmas 3.3.2, 3.3.3 & 3.4.1). Since number of recursive steps are bounded by $\deg_x(a) \geq \deg_x(b)$, total time is bounded by $\tilde{O}(\deg_x(a)^3 \cdot \deg(I_l)^4 \cdot \log|\mathbb{F}_q|)$. □

Chapter 4

Derandomizing Univariate Root Counting modulo Prime Powers

In this chapter we will provide an algorithm to count all the roots of a univariate polynomial $f(x)$ with integer coefficients modulo a prime power p^k in deterministic polynomial-time. Thus we also solve the problem of testing the existence of a root of f mod p^k in deterministic polynomial-time.

Theorem 4.0.1 (Root count)**.** *Let p be a prime, $k \in \mathbb{N}$ and $f(x) \in \mathbb{Z}[x]$. Then all the roots of f mod p^k can be counted in deterministic poly($\deg f, k \log p$)-time.*

The outline is similar to that of randomized root counting (Chapter 2, Section 2.3) but the data structure, split ideals, returned by our algorithm contains roots implicitly. Moreover, if randomization is allowed, the data structure naturally *provides* all the roots efficiently.

Notations: In this chapter we will use the notations defined and used in Chapter 3.

4.1 Counting All the Roots of $f(x)$ modulo Prime Power p^k

The algorithm to compute a compact data-structure which stores roots of f mod p^k will be described in Section 4.1.1. The correctness of our algorithm will be proved in Section 4.1.2, which involves studying the algebraic structure underlying the algorithm. Its efficiency will be

shown in Section 4.1.3, by devising an auxiliary structure called roots-tree and the important notion of 'degree of a node'.

4.1.1 Algorithm to Implicitly Partition the Root-Set

The input and output to our algorithm are as follows.

Input: A monic univariate polynomial $f \in \mathbb{Z}[x]$ of degree d and a prime-power p^k (in binary).
Output: A list $\mathcal{L}$ of at most d maximal split ideals whose roots partition the root-set of $f \bmod p^k$.

A maximal split ideal $\hat{I}_j =: \langle \hat{h}_0(\bar{x}_0), \ldots, \hat{h}_l(\bar{x}_l) \rangle$ has $|\mathcal{Z}_R(\hat{I}_j)| = \prod_{i=0}^{l} \deg_{x_i}(\hat{h}_i)$ zeros (Lemma 3.2.4), and each such zero 'represents' p^{k-l-1} actual zeros of $f \bmod p^k$ (Lemma 3.2.8). Thus this algorithm gives an exact count on the number of zeros of f in R.

Overview of Algorithm 5: Since any root of $f \bmod p^k$ is a lift of a root modulo p, the algorithm starts by initializing a stack S (Steps $1-3$) with the ideal $\hat{I} := \langle \hat{h}_0(x_0) \rangle$, where where $\hat{h}_0$ is some 'monic' lift of $h_0(x_0) := \gcd(x_0^p - x_0, f(x_0))$ over R. This is indeed a split ideal containing all the roots of $f \bmod p$ (for every $r_0 \in \mathcal{Z}_R(\hat{I}_0)$ we have $f(r_0) \equiv 0 \bmod p$).

At every intermediate iteration (Steps $4-21$), we *pop* a split ideal from the stack and *try* to increase the precision of its root-set (equivalently, lengthen the split ideal) before pushing it back to Stack S. This step mostly results in two cases: either we succeed and get a split ideal whose root-set has increased precision (Step 18) by a new placeholder x_{l+1}, or the split ideal factors into more split ideals increasing the size of the stack S (Steps $10, 14, 20$). We update the relevant 'part of f' to $f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$ ($\hat{J}$ is the new split ideal) that we carry around with each split ideal. This helps in efficiently increasing the precision of roots in the next iteration. Otherwise, computing $f\left(x_0 + px_1 + \cdots + p^l x_l + p^{l+1}x\right)/p^\alpha \bmod \hat{I}$ is too expensive, in Step 7, due to the underlying degree-d $(l+1)$-variate monomials blowup.

If we reach a maximal split ideal (Step 6), it is moved to a list $\mathcal{L}$. Sometimes the split ideal cannot be extended and we get a *dead-end* (Step 16). The size of the stack decreases when we get a maximal split ideal or a dead-end. The algorithm terminates when stack becomes empty. List $\mathcal{L}$ contains maximal split ideals which partition, and cover, the root-set of f (implicitly). This becomes our output.

The main intuition behind our algorithm: If two roots of a split ideal (representing potential roots of f) give rise to different number of roots of f, the split ideal will get split further. Though not at all apparent immediately, we will show that the algorithm takes only polynomial number of steps (Section 4.1.3).

We will use four subroutines to perform standard ring arithmetic modulo split ideals; they are described in the Sections 3.3 & 3.4.

1. Modify f (Steps 3, 18, 20) whenever pushing in the stack (Lemma 4.1.1 & 4.1.2).
2. REDUCE$(a(\bar{x}_l), J_l)$ gives the reduced form of a mod triangular ideal J_l (over a Galois ring).
3. TEST-ZERO-DIV$(a(\bar{x}_l), I_l)$ either reports that a is a not a zero-divisor modulo triangular ideal I_l or outputs a non-trivial factorization of one of the generators of I_l when true.
4. GCD$(a(\bar{x}_l, x), b(\bar{x}_l, x), I_l)$ either successfully computes a monic gcd, with respect to x, of $a(\bar{x}_l, x)$ and $b(\bar{x}_l, x)$ modulo a triangular ideal I_l, or encounters a zero-divisor in intermediate computation (outputting $False$ and a non-trivial factorization of one of the generators of I_l).

Algorithm 5 Root-counting mod p^k

1: Let $\mathcal{L} = \{\}$ be a list and $S = \{\}$ be a stack (both initially empty).

2: Let $\tilde{f}(x_0) := f(x_0) \bmod p$ for a monic univariate $\tilde{f} \in \mathbb{F}_p[x_0]$ of degree d.

3: [**Initializing the stack S**] Let $h_0(x_0) := \gcd(\tilde{f}(x_0), x_0^p - x_0)$, $I := \langle h_0 \rangle$. Let $\hat{I} \subseteq R[x_0]$ be a lift of I defined as $\hat{I} = \langle \hat{h}_0(x_0) \rangle$ where $\hat{h}_0$ is a 'monic' lift of h_0 over R. Compute $f_I(x_0, x) := f(x_0 + px) \bmod \hat{I}$ using Lemma 4.1.1. Update $S \leftarrow push((\{\hat{h}_0\}, f_I))$.

4: **while** S is not empty **do**

5: $S_{top} \leftarrow pop(S)$. Let $S_{top} = (\{\hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\}, f_I(\bar{x}_l, x))$ where $\hat{I} = \langle \hat{h}_0, \ldots, \hat{h}_l \rangle \subseteq R[\bar{x}_l]$ is a split ideal and $f_I(\bar{x}_l, x)$ is already computed in reduced form modulo $\hat{I}$.

6: [**Maximal split ideal found**] **if**$(f_I \equiv 0 \bmod \hat{I})$ **then** update $\mathcal{L} \leftarrow \mathcal{L} \cup \{\hat{I}\}$. Go to Step 4.

7: [**Valuation computation**] Compute $\alpha \in \mathbb{N}$ and $g \in R[\bar{x}_l, x]$ such that $f_I \equiv p^\alpha g(\bar{x}_l, x) \bmod \hat{I}$ and $p \not| g \bmod \hat{I}$. [Note: $\alpha < k$ due to check at Step 6.]

8: Let $\tilde{g} := g(\bar{x}_l, x) \bmod I$ (where $I := \hat{I} \bmod p$) be the polynomial in $\mathbb{F}_p[\bar{x}_l, x]$, and let $g_1(\bar{x}_l)$ be the leading coefficient of $\tilde{g}(\bar{x}_l, x)$ with respect to x.

9: **if** TEST-ZERO-DIV($g_1(\bar{x}_l)$, I)= $True$ **then**

10: TEST-ZERO-DIV$(g_1(\bar{x}_l), I)$ returns a factorization $h_i(\bar{x}_i) =: h_{i,1}(\bar{x}_i) \cdots h_{i,m}(\bar{x}_i) \bmod I_{i-1}$ of some generator $h_i(\bar{x}_i)$ of I. Use univariate Hensel lifting, with respect to x_i (Lemma 2.2.2), to lift each $h_{i,j} \in \mathbb{F}_p[\bar{x}_i]$ to $\hat{h}_{i,j} \in R[\bar{x}_i]$ and get the factorization $\hat{h}_i = \hat{h}_{i,1} \cdots \hat{h}_{i,m} \bmod \hat{I}_{i-1}$. Go to Step 20.

[**Filter out distinct virtual $\mathbb{F}_p$-roots by taking gcd with $x^p - x$**]

11: Recompute $\tilde{g} = g(\bar{x}_l, x) \cdot g_1(\bar{x}_l)^{-1} \bmod I$ (Lemmas 3.3.3, 3.3.2). Compute x^p by repeatedly squaring and reducing modulo the triangular ideal $I + \langle \tilde{g} \rangle$ (Algorithm 2 and Lemma 3.3.2). This yields $\tilde{h}_{l+1}(\bar{x}_l, x) := x^p - x \bmod I + \langle \tilde{g} \rangle$ in a reduced form. [Note: $\gcd(x^p - x, \tilde{g}) = \gcd(\tilde{g}, (x^p - x) \bmod \tilde{g})$.]

12: **if** GCD($\tilde{g}$, $\tilde{h}_{l+1}$, I) = $False$ **then**

13: The call GCD$(\tilde{g}, \tilde{h}_{l+1}, I)$ returns a factorization $h_i(\bar{x}_i) =: h_{i,1}(\bar{x}_i) \cdots h_{i,m}(\bar{x}_i) \bmod I_{i-1}$ of some generator $h_i(\bar{x}_i)$ of I. Use univariate Hensel lifting, with respect to x_i (Lemma 2.2.2), to lift each $h_{i,j} \in \mathbb{F}_p[\bar{x}_i]$ to $\hat{h}_{i,j} \in R[\bar{x}_i]$ and get the factorization $\hat{h}_i = \hat{h}_{i,1} \cdots \hat{h}_{i,m} \bmod \hat{I}_{i-1}$. Go to Step 20.

14: **else if** $\tilde{g}$ and $\tilde{h}_{l+1}$ are coprime **then**

15: [**Dead End**] The ideal $\hat{I}$ cannot grow more, go to Step 4.

16: **else**

17: [**Grow the split ideal I**] Here $\gcd_x(\tilde{g}, \tilde{h}_{l+1}) \bmod I$ is non-trivial, say $h_{l+1}(\bar{x}_l, x)$ (monic with respect to x). Substitute x by x_{l+1} in $h_{l+1}(\bar{x}_l, x)$ and update $\hat{J} \leftarrow \hat{I} + \langle \hat{h}_{l+1}(\bar{x}_{l+1}) \rangle$ where $\hat{h}_{l+1}$ is a 'monic' lift, with respect to x_{l+1}, of h_{l+1} over R. Substitute x by $x_{l+1} + px$ in $f_I(\bar{x}_l, x)$, and compute $f_J(\bar{x}_{l+1}, x) := f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$ using Lemma 4.1.1. Update $S \leftarrow push((\{\hat{h}_0, \ldots, \hat{h}_{l+1}\}, f_J))$, and go to Step 4.

18: [**Factoring split ideals**] We have a factorization $\hat{h}_i(\bar{x}_i) = \hat{h}_{i,1}(\bar{x}_i)\cdots\hat{h}_{i,m}(\bar{x}_i) \bmod \hat{I}_{i-1}$ of a generator $\hat{h}_i$ of $\hat{I}$. Push S_{top} back in stack S. For every entry $(U, f_{\langle U\rangle}) \in S$, where $\hat{h}_i(\bar{x}_i)$ appears in U, find m (smaller) split ideals $\langle U_j\rangle$ (using Corollary 3.2.6); using Lemma 4.1.2 compute U_j and $f_{\langle U_j\rangle}$ in reduced form and push $(U_j, f_{\langle U_j\rangle})$ in S, for $j \in [m]$.

19: Return $\mathcal{L}$ (the list of maximal split ideals partitioning the root-set $\mathcal{Z}_R(f)$).

First, we explain how Algorithm 5 (Steps 3, 18) computes reduced f_J modulo the newly computed split ideal $\hat{J}$, when x is replaced by $x_{l+1} + px$ in the intermediate polynomial $f_I(\bar{x}_l, x)$.

Lemma 4.1.1 (Updating stack with reduced polynomial)**.** *Let $\hat{I} \subseteq R[\bar{x}_l]$ be a split ideal defined as $\hat{I} := \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\rangle$ and $f_I(\bar{x}_l, x) \in R[\bar{x}_l, x]$ be reduced modulo $\hat{I}$. Define split ideal $\hat{J} \subseteq R[\bar{x}_{l+1}]$ as $\hat{J} := \hat{I} + \langle \hat{h}_{l+1}(\bar{x}_{l+1})\rangle$ where $\deg_{x_{l+1}}(\hat{h}_{l+1}) \le \deg_x(f_I)$.*

Then, in time $\tilde{O}(\deg_x(f_I)^3 \cdot \deg(\hat{I})^4 \cdot \log|R|)$, we can compute a reduced polynomial f_J modulo $\hat{J}$ defined by, $f_J(\bar{x}_{l+1}, x) := f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$.

Proof. Since $f_I(\bar{x}_l, x)$ is already reduced modulo $\hat{I}$, $\deg_{x_i}(f_I) < \deg_{x_i}(\hat{h}_i)$. Define $D := \deg_x(f_I)$, perform the shift $x \to x_{l+1} + px$ in f_I, and expand f_I using Taylor series,

$$f_J(\bar{x}_l, x) \;=\; f_I(\bar{x}_l, x_{l+1} + px) \;=:\; g_0(\bar{x}_{l+1}) + g_1(\bar{x}_{l+1})(px) + \ldots + g_D(\bar{x}_{l+1})(px)^D \;,$$

where g_i could also be seen as the i-th derivative of $f_I(\bar{x}_l, x_{l+1})$ (with respect to x_{l+1}) divided by $i!$. To compute $f_J \bmod \hat{J}$, we call $\textsc{Reduce}(g_i, \hat{J})$ (for all i) to get the reduction of each term mod $\hat{J}$.

To calculate the time complexity of $\textsc{Reduce}(g_i, \hat{J})$, note that coefficients of each g_i, with respect to x_{l+1}, is already reduced mod $\hat{I}$. Since $\hat{J} = \hat{I} + \langle h_{l+1}\rangle$, using Lemma 3.3.2, time complexity of reducing each g_i by $\hat{J}$ is at most $\tilde{O}(\deg_{x_{l+1}}(g_i)^2 \cdot \deg(\hat{I})^4 \cdot \log|R|)$.

Since $\deg_{x_{l+1}}(g_i) \le \deg_x(f_I)$ (for $i \le D$) and there are $\deg_x(f_I) + 1$ many g_is, total time complexity is $\tilde{O}(\deg_x(f_I)^3 \cdot \deg(\hat{I})^4 \cdot \log|R|)$. □

Next, we explain Step 20 in Algorithm 5 a bit more.

Lemma 4.1.2 (Ideal factors in reduced form)**.** *Consider the tuple* $(U, f_{\langle U\rangle}(\bar{x}_l, x)) \in S$*, where* $U := \{\hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l)\}$*, and consider a non-trivial factorization* $\hat{h}_i =: \hat{h}_{i,1} \cdots \hat{h}_{i,m}$ *for some* $\hat{h}_i \in U$ *where each factor* $\hat{h}_{i,j}$ *is monic with respect to* x_i*.*

Then we can compute the factor-related tuples $(U_j, f_{\langle U_j\rangle})$*, for all* $j \in [m]$*, in time* $\tilde{O}(m \cdot \deg_x(f_{\langle U\rangle}) \cdot \deg(\langle U\rangle)^4 \cdot \log|R|)$ *(*$f_{\langle U_j\rangle}$ *will be in reduced form modulo* $\langle U_j\rangle$*).*

Proof. We have $U = \{\hat{h}_0(\bar{x}_0), \ldots, \hat{h}_l(\bar{x}_l)\}$. Using Corollary 3.2.6 we get splits of U in non-reduced from as: $U_j = \{\hat{h}_0, \ldots, \hat{h}_{i-1}, \hat{h}_{i,j}, \hat{h}_{i+1}, \ldots, \hat{h}_l\}$ for all $j \in [m]$.

So we first successively reduce $\hat{h}_{i+t}$, for $1 \leq t \leq l-i$, modulo the triangular ideal $\hat{I}_{i+t-1,j}$ defined as $\hat{I}_{i+t-1,j} := \langle \hat{h}_0, \ldots, \hat{h}_{i-1}, \hat{h}_{i,j}, \hat{h}_{i+1}, \ldots, \hat{h}_{i+t-1}\rangle$. Time complexity of each of these steps is bounded by $\tilde{O}(\deg(\langle U\rangle)^4 \cdot \log|R|)$ (Lemma 3.3.2). We replace each $\hat{h}_{i+t}$ in U_j with the new *reduced* $\hat{h}_{i+t}$ to get reduced U_j.

Then $f_{\langle U_j\rangle}$ can be calculated by reducing each $\deg_x(f_{\langle U\rangle}) + 1$ coefficients of $f_{\langle U\rangle}$ (with respect to x) by the split ideal $\langle U_j\rangle$. Since coefficients (with respect to x) of $f_{\langle U\rangle}$ were already reduced modulo $\langle U\rangle$, by Lemma 3.3.2 the computation time is bounded by $(\deg_x(f_{\langle U\rangle}) + 1) \cdot \tilde{O}(\deg(\langle U\rangle)^4 \cdot \log|R|) = \tilde{O}(\deg_x(f_{\langle U\rangle}) \cdot \deg(\langle U\rangle)^4 \cdot \log|R|)$. Since $j \in [m]$, total computation time is $\tilde{O}(m \cdot \deg_x(f_{\langle U\rangle}) \cdot \deg(\langle U\rangle)^4 \cdot \log|R|)$. □

We now illustrate Algorithm 5 with the help of the following example.

Example 9. *Let we are given* $f(x) = x^4 - 8x^3 + 7x^2$ *and* $p^k = 7^5$*.*

In Steps 1-3 the algorithm computes $\tilde{f} := f \bmod 7 = x^4 - x^3 = x^3(x-1)$ *and takes gcd of* $\tilde{f}$ *with* $x^7 - x$ *which is* $x(x-1)$*. This gives* $h_0(x_0) = x_0^2 - x_0$*. Let monic R-lift of* h_0 *is defined as* $\hat{h}_0 := x_0^2 - x_0$ *and* $\hat{I}_0 = \langle \hat{h}_0\rangle$*.*

Substituting $x \to x_0 + 7x$ *in* $f(x)$ *and reducing modulo* $\hat{I}_0$ *(i.e., replacing* x_0^4*,* x_0^3 *and* x_0^2 *by* x_0*) we get* $f_{I_0}(x_0, x) := f(x_0 + 7x) = 7 \cdot [7^3x^4 + 7^2x^3(4x_0 - 8) + 7x^2(-18x_0 + 7) - (6x_0)x] \bmod \hat{I}_0 + \langle 7^5\rangle$*. Then* $(\hat{I}_0, f_{I_0})$ *is pushed into Stack S.*

First iteration of the while loop*: The algorithm pops* $(\hat{I}_0, f_{I_0})$ *and at Step-7, gets* $\alpha = 1$ *and* $g(x_0, x) = 7^3x^4 + 7^2x^3(4x_0 - 8) + 7x^2(-18x_0 + 7) - (6x_0)x$ *modulo* $\hat{I}_0 := \langle x_0^2 - x_0\rangle$*. At Step-8 we get* $\tilde{g} = (-6x_0)x \bmod I_0$ *and* $g_1 = -6x_0 \bmod I_0$ *where* $I_0 := \hat{I}_0 \bmod 7$*.*

Step-9 computes that g_1 *is a zero-divisor and this splits* $\hat{h}_0 = x_0^2 - x_0$ *into* $\hat{h}_{0,1} = x_0$ *and* $\hat{h}_{0,2} = x_0 - 1$ *(at Step-10). Then Step-10 sends the control to Step-20.*

At Step-20, the algorithm discards the current tuple $(\hat{I}_0, f_{I_0})$ *and push back into* S *two new tuples* $(\hat{I}_{0,1}, f_{I_{0,1}})$ *and* $(\hat{I}_{0,2}, f_{I_{0,2}})$ *where* $\hat{I}_{0,1} := \langle \hat{h}_{0,1} \rangle = \langle x_0 \rangle$ *and* $\hat{I}_{0,2} := \langle \hat{h}_{0,2} \rangle = \langle x_0 - 1 \rangle$ *and* $f_{I_{0,1}}$, $f_{I_{0,2}}$ *are as follows:*

$f_{I_{0,1}} = f_{I_0} \bmod \hat{I}_{0,1} + \langle 7^5 \rangle = 7^3 \cdot [7x^4 - 8x^3 + x^2] \bmod \hat{I}_{0,1} + \langle 7^5 \rangle$ *(substituting* $x_0 = 0$*) and,*

$f_{I_{0,2}} = f_{I_0} \bmod \hat{I}_{0,2} + \langle 7^5 \rangle = 7 \cdot [7^3 x^4 - 4 \cdot 7^2 x^3 - 11 \cdot 7x^2 - 6x] \bmod \hat{I}_{0,1} + \langle 7^5 \rangle$ *(substituting* $x_0 = 1$*).*

Second iteration of the while loop*: The algorithm pops* $(\hat{I}_{0,1}, f_{I_{0,1}})$ *and at Step-7 we get* $\alpha = 3$ *and* $g = 7x^4 - 8x^3 + x^2 \bmod \hat{I}_{0,1}$*. Step-8 gives* $\tilde{g} = x^2 - x^3$ *and* $g_1 = -1$ *(not a zero-divisor).*

So Step-12 recomputes $\tilde{g} = x^3 - x^2 = x^2(x-1)$ *and takes gcd of* $\tilde{g}$ *with* $x^7 - x$ *which is* $x(x-1) = x^2 - x$ *(non-trivial gcd).*

This leads us to Step-18 where we get $\hat{h}_1(x_0, x_1) = x_1^2 - x_1$ *and* $\hat{I}_1 := \hat{I}_{0,1} + \langle \hat{h}_1 \rangle = \langle x_0, x_1^2 - x_1 \rangle$*. Step-18 then push* $(\hat{I}_1, f_{I_1})$ *into* S *where* $f_{I_1} = f_{I_{0,1}}(x_0, x_1 + 7x) \bmod \hat{I}_1 + \langle 7^5 \rangle$*. Expanding* $f_{I_{0,1}}(x_0, x_1 + 7x)$*, replacing* x_1^4*,* x_1^3 *and* x_1^2 *by* x_1 *(reducing mod* $\hat{I}_1$*), and reducing modulo* $\langle 7^5 \rangle$ *we get,*

$f_{I_1}(\bar{x}_1, x) = 7^4 \cdot (-22x_1)x \bmod \hat{I}_1 + \langle 7^5 \rangle$*. The control now moves to Step-4.*

Third iteration of the while loop*: The algorithm pops* $(\hat{I}_1, f_{I_1})$ *where (at Step-7) we have* $\alpha = 4$ *and* $g = (-22x_1)x$*. Thus at Step-8,* $\tilde{g} = -x_1 \bmod I_1$ *and* $g_1 = -1$ *where* $I_1 := \hat{I}_1 \bmod 7$*.*

Step-9 tests g_1 *to be a zero-divisor and Step-10 sends control to Step-20 after splitting* $\hat{h}_1 = x_1(x_1 - 1)$ *into* $\hat{h}_{1,1} := x_1$ *and* $\hat{h}_{1,2} := x_1 - 1$*.*

At Step-20, the algorithm splits $\hat{I}_1$ *into* $\hat{I}_{1,1} := \hat{I}_{0,1} + \langle \hat{h}_{1,1} \rangle = \langle x_0, x_1 \rangle$ *and* $\hat{I}_{1,2} := \hat{I}_{0,1} + \langle \hat{h}_{1,2} \rangle = \langle x_0, x_1 - 1 \rangle$ *and computes* $f_{I_{1,1}} = f_{I_1} \bmod \hat{I}_{1,1} + \langle 7^5 \rangle = 0$ *(substituting* $x_1 = 0$*) and* $f_{I_{1,2}} = f_{I_1} \bmod \hat{I}_{1,2} + \langle 7^5 \rangle = 7^4(-22)x$ *(substituting* $x_1 = 1$*). It discards the current tuple* $(\hat{I}_1, f_{I_1})$ *and push tuples* $(\hat{I}_{1,1}, f_{I_{1,1}})$ *and* $(\hat{I}_{1,2}, f_{I_{1,2}})$ *into* S*.*

Fourth iteration of the while loop*: The algorithm pops* $(\hat{I}_{1,1}, f_{I_{1,1}})$ *and adds* $\hat{I}_{1,1}$ *to List* $\mathcal{L}$ *(Step-6) as* $f_{I_{1,1}}(\bar{x}_1, x)$ *is identically zero modulo* $\hat{I}_{1,1} + \langle 7^5 \rangle$ *(*$\hat{I}_{1,1}$ *is a maximal split ideal). Go to Step-4 now.*

***Fifth iteration of the while loop**: The algorithm pops $(\hat{I}_{1,2}, f_{I_{1,2}})$ and gets (at Step-7) $\alpha = 4$, $g = -22x$ and (at Step-8) $\tilde{g} = -x$ and $g_1 = -1$. After recomputing $\tilde{g}$ at Step-12, we have $\tilde{g} = x$. Thus at Step-18, we get $gcd(\tilde{g}, x^7 - x) = x$. This yields $\hat{h}_{2,1} := x_2$ and the split ideal $\hat{I}_{2,1} := \hat{I}_{1,2} + \langle \hat{h}_{2,1} \rangle = \langle x_0, x_1 - 1, x_2 \rangle$. The algorithm (Step-18) then push $(\hat{I}_{2,1}, f_{I_{2,1}})$ into S where $f_{I_{2,1}} := f_{I_{1,2}}(\bar{x}_1, x_2 + 7x) \equiv 0 \bmod \hat{I}_{2,1} + \langle 7^5 \rangle$ (obtained by substituting $x_2 = 0$ and reducing mod 7^5). The control now moves to Step-4.*

***Sixth iteration of the while loop**: The algorithm pops $(\hat{I}_{2,1}, f_{I_{2,1}})$ and adds $\hat{I}_{2,1}$ to List $\mathcal{L}$ (Step-6) as $f_{I_{2,1}}(\bar{x}_2, x)$ is identically zero modulo $\hat{I}_{2,1} + \langle 7^5 \rangle$ ($\hat{I}_{2,1}$ is a maximal split ideal). Go to Step-4 now.*

***Seventh iteration of the while loop**: The algorithm pops $(\hat{I}_{0,2}, f_{I_{0,2}})$ and gets (at Step-7) $\alpha = 1$, $g = 7^3x^4 - 4 \cdot 7^2x^3 - 11 \cdot 7x^2 - 6x$ and (at Step-8) $\tilde{g} = -6x$ and $g_1 = -6$. After recomputing $\tilde{g}$ at Step-12, we have $\tilde{g} = x$. Thus at Step-18, we get $gcd(\tilde{g}, x^7 - x) = x$. This yields $\hat{h}_{1,3} := x_1$ and the split ideal $\hat{I}_{1,3} := \hat{I}_{0,2} + \langle \hat{h}_{1,3} \rangle = \langle x_0 - 1, x_1 \rangle$. The algorithm (Step-18) then push $(\hat{I}_{1,3}, f_{I_{1,3}})$ into S where $f_{I_{1,3}} := f_{I_{0,2}}(x_0, x_1 + 7x) \equiv 7^2 \cdot [-11 \cdot 7^2x^2 - 6x] \bmod \hat{I}_{1,3} + \langle 7^5 \rangle$ (obtained by substituting $x_1 = 0$ and reducing mod 7^5). The control now moves to Step-4.*

As the last few iterations of the loop are similar to seventh iteration we omit the details. In eighth iteration we get $\hat{I}_{2,2} := \hat{I}_{1,3} + \langle \hat{h}_{2,2} := x_2 \rangle$. In ninth iteration we get $\hat{I}_3 := \hat{I}_{2,2} + \langle \hat{h}_3 := x_3 \rangle$ and then in tenth iteration we get $\hat{I}_4 := \hat{I}_3 + \langle \hat{h}_4 := x_4 \rangle$ which is added to $\mathcal{L}$ as it is a maximal split ideal.

Algorithm 5 returns $\mathcal{L} = \{\hat{I}_{1,1}, \hat{I}_{2,1}, \hat{I}_4\}$ where $\hat{I}_{1,1} = \langle x_0, x_1 \rangle$, $\hat{I}_{2,1} = \langle x_0, x_1 - 1, x_2 \rangle$ and $\hat{I}_4 = \langle x_0 - 1, x_1, x_2, x_3, x_4 \rangle$. The degree of each of these ideals (Definition 3.2.1) is one so each of them have exactly one zero in their zero-set over $R = \mathbb{Z}/\langle 7^5 \rangle$.

We have $(0, 0) \in \mathcal{Z}_R(\hat{I}_{1,1})$, so roots represented by $\hat{I}_{1,1}$ are of the form $0 + 0 \cdot 7 + 7^2x_2 + 7^3x_3 + 7^4x_4 = 7^2x_2 + 7^3x_3 + 7^4x_4$ for $x_2, x_3, x_4 \in \{0, \ldots, 6\}$. Thus it represents 7^3 roots of $f \bmod 7^5$ (also by Lemma 3.2.8).

We have $(0, 1, 0) \in \mathcal{Z}_R(\hat{I}_{2,1})$, so roots represented by $\hat{I}_{2,1}$ are of the form $0 + 1 \cdot 7 + 0 \cdot 7^2 + 7^3x_3 + 7^4x_4 = 7 + 7^3x_3 + 7^4x_4$ for $x_3, x_4 \in \{0, \ldots, 6\}$. Thus it represents 7^2 roots of $f \bmod 7^5$ (also by Lemma 3.2.8).

We have $(1, 0, 0, 0, 0) \in \mathcal{Z}_R(\hat{I}_4)$, so $\hat{I}_4$ represents exactly one root of $f \bmod 7^5$ which is

$1 + 0 \cdot 7 + 0 \cdot 7^2 + 0 \cdot 7^3 + 0 \cdot 7^4 = 1$.

Thus root count of $f = x^4 - 8x^3 + 7x^2 \bmod 7^5$ *is* $7^3 + 7^2 + 1 = 393$.

One can verify the roots computed in Example 9 using the randomized root counting algorithm of Chapter 2. We refer the reader to Section 4.1.3 to see the pictorial view of working of Algorithm 5 on Example 9.

4.1.2 Correctness of the Algorithm

Our main goal is to prove the following result about partitioning of root-set.

Theorem 4.1.3 (Algorithm 5 partitions $\mathcal{Z}_R(f)$)**.** *Algorithm 5 yields the structure of the root-set* $\mathcal{Z}_R(f)$ *through a list data structure* $\mathcal{L}$ *(a collection of maximal split ideals* $\hat{I}_1, \ldots, \hat{I}_n$*) which partitions the zero-set* $\mathcal{Z}_R(f) =: \bigsqcup_{j\in[n]} S_j$*, where* S_j *is the set of roots of* $f \bmod p^k$ *represented by* $\mathcal{Z}_R(\hat{I}_j)$.

The maximum number n of partitions, given by Theorem 4.1.3, is bounded by degree d (in Example 9, $d = 4$ and $n = 3$). This will be proved in Section 4.1.3.

We defer the proof of Theorem 4.1.3 to the end of this subsection. For now, let us see the properties of our algorithm which go in proving Theorem 4.1.3.

Given a polynomial $g(\bar{x}_l) \in \mathbb{F}_p[\bar{x}_l]$ and an element $\bar{a} \in \mathbb{F}_p^l$, consider the *projection* $g_{\bar{a}}(x_l) := g(\bar{a}, x_l)$. Chinese remaindering (on decomposition given by Lemma 3.2.5) gives us the following gcd property under projections: GCD taken at Step-18 of Algorithm 5 is equivalent to univariate gcd over $\mathbb{F}_p$ under projections.

Lemma 4.1.4. *Let* $w(\bar{x}_l), z(\bar{x}_l) \in \mathbb{F}_p[\bar{x}_l]$ *and* $I_{l-1} \subseteq \mathbb{F}_p[\bar{x}_{l-1}]$ *be the reduction modulo* p *of a split ideal* $\hat{I} \subseteq R[\bar{x}_{l-1}]$*. Define* $h(\bar{x}_l) := \mathrm{GCD}(w(\bar{x}_l), z(\bar{x}_l), I_{l-1})$ *such that* h *is monic, with respect to* x_l*, and there exist* $u, v \in (\mathbb{F}_p[\bar{x}_{l-1}]/I_{l-1})[x_l]$ *such that* $h = uw + vz \bmod I_{l-1}$*. Then, for all* $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$*:* $h_{\bar{a}}(x_l) \in \mathbb{F}_p[x_l]$ *equals* $\gcd(w_{\bar{a}}(x_l), z_{\bar{a}}(x_l)) \in \mathbb{F}_p[x_l]$ *up to a unit multiple (in* $\mathbb{F}_p^*$*).*

Proof. $h(\bar{x}_l)$ is a monic polynomial mod I_{l-1}, s.t., $h|w$ and $h|z$ (mod I_{l-1}). Fix $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$. Since $h_{\bar{a}}(x_l) \not\equiv 0 \bmod p$ ($\because$ h is monic), restricting $\bar{x}_{l-1}$ to $\bar{a}$ gives $h_{\bar{a}}|w_{\bar{a}}$ and $h_{\bar{a}}|z_{\bar{a}}$, showing $h_{\bar{a}}|\gcd(w_{\bar{a}}, z_{\bar{a}})$, in $\mathbb{F}_p[x_l]$.

Since there exists $u, v \in (\mathbb{F}_p[\bar{x}_{l-1}]/I_{l-1})[x_l]$, such that, $h = uw + vz$. Restricting first l co-ordinates to $\bar{a}$, we get $h_{\bar{a}} = u_{\bar{a}} w_{\bar{a}} + v_{\bar{a}} z_{\bar{a}}$. This equation implies $\gcd(w_{\bar{a}}, z_{\bar{a}}) | h_{\bar{a}}$. Thus we get $h_{\bar{a}}(x_l) = \gcd(w_{\bar{a}}(x_l), z_{\bar{a}}(x_l))$ up to a unit multiple. □

Algorithm 5 calls Algorithm 4 to compute gcd of $\tilde{g}(\bar{x}_l, x)$ and $\tilde{h}_{l+1}(\bar{x}_l, x)$ (with respect to x) modulo $I \subseteq \mathbb{F}_p[\bar{x}_l]$ which in turn gives us a gcd satisfying the conditions of Lemma 4.1.4. Thus the conclusion of Lemma 4.1.4 holds for Step-18 of Algorithm 5.

Prefix-free: Let $\hat{I} \subseteq R[\bar{x}_i], \hat{J} \subseteq R[\bar{x}_j]$ be two split ideals (say $i \leq j$). $\hat{I}$ and $\hat{J}$ are called *prefix-free* iff $\nexists\ \bar{a} = (a_0, a_1, \ldots, a_i) \in \mathcal{Z}_R(\hat{I}),\ \bar{b} = (b_0, b_1, \ldots, b_j) \in \mathcal{Z}_R(\hat{J}) : a_l = b_l\ \forall l \leq i$. (Note that it may still happen that $(a_0, \ldots, a_{i-1}) = (b_0, \ldots, b_{i-1})$.)

Our next lemma shows an invariant about Algorithm 5. Essentially it says that Algorithm 5 push and pop only split ideals in and out of Stack S and all the ideals in S remain disjoint (with respect to roots) all the time.

Lemma 4.1.5 (Stack contents)**.** *Stack S in Algorithm 5 satisfies following conditions at every point:*

1) Let $(\hat{I}_l, f_{I_l}) \in S$ such that $p^\alpha || f_{I_l} \bmod \hat{I}_l$. Then the length of $\hat{I}_l$, $l + 1 \leq \min(\alpha, k)$.

2) All ideals in S are split ideals.

3) Any two ideals in S are prefix-free.

Proof. Let us assume that ideals in S are split ideals (proved later), to show that their lengths are at most $\min(\alpha, k)$. Step 7 defines g via f_{I_l} as, $f_{I_l} =: p^\alpha g(\bar{x}_l, x) \bmod \hat{I}_l$. Looking at the f_{I_l} analogues pushed in Steps 3, 18, 20, one can easily deduce $f(\sum_{0 \leq i \leq l} x_i p^i + x p^{l+1}) \equiv f_{I_l}(\bar{x}_l, x) \bmod \hat{I}_l$. We also have for all $(a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I}_l)$, $f(a_0 + \cdots + p^l a_l) \equiv 0 \bmod p^{l+1}$ (condition 3 of Definition 3.2.1). Thus applying inverse chinese remaindering (on the decomposition given by Lemma 3.2.5) we get $f(\sum_{0 \leq i \leq l} x_i p^i) \equiv 0 \bmod \hat{I}_l + \langle p^{l+1} \rangle$.

Thus $f\left(\sum_{0 \leq i \leq l} x_i p^i\right) \equiv p^\alpha g(\bar{x}_l, x) \equiv 0 \bmod \hat{I} + \langle p^{l+1} \rangle$. Since, $p \nmid g \bmod \hat{I}$, we deduce $\alpha \geq l + 1$. Moreover, by Step 6 we know that $l < k$ throughout the algorithm.

Now we will prove invariants 2 and 3. There are three ways in which a new ideal is added to stack S. We show below that the invariant is maintained in all three cases.

(Step 3) S is initialized with the ideal $\hat{I}_0 = \langle \hat{h}_0(x_0) \rangle \subseteq R[x_0]$ such that $h_0 \equiv \hat{h}_0 \bmod$

$p := gcd(f(x_0) \bmod p, x_0^p - x_0)$. The triangular ideal $\hat{I}_0$ is a split ideal, because $|\mathcal{Z}_{\mathbb{F}_p}(I_0)| = \deg_{x_0}(h_0)$ (where $I_0 := \hat{I}_0 \bmod p$) and its root are all the distinct roots of $f(x_0) \bmod p$.

(Step 20) Ideal $\hat{I}_l$ is popped from S, and some generator $\hat{h}_i$ of $\hat{I}_l$ splits. In this case, we update S with the corresponding factors of any $(\hat{I}, f_I) \in S$, whenever currently $\hat{I}$ has $\hat{h}_i$. Corollary 3.2.6 shows that the factors of $\hat{I}$ are split ideals themselves, and their root-sets partition that of $\hat{I}$ (by Lemma 3.2.5). Thus these root-sets are prefix-free among themselves. Moreover, they are prefix-free with any other ideal $\hat{J}$ appearing in S, because $\hat{I}$ was prefix-free with $\hat{J}$.

(Step 18) Ideal $\hat{I}_l$ is popped and it grows to $\hat{I}_{l+1}$ as follows: Let $f_{I_l} =: p^\alpha g(\bar{x}_l, x) \bmod \hat{I}_l$, where $\alpha \geq l+1$ (Invariant 1 on split ideal $\hat{I}_l$), and $\tilde{g} = g \bmod p$. Algorithm 5 computes $h_{l+1} := gcd(\tilde{g}, x^p - x) \bmod I_l$ where $I_l := \hat{I}_l \bmod p$. Then it defines $\hat{I}_{l+1} := \hat{I}_l + \langle \hat{h}_{l+1}(\bar{x}_l, x_{l+1}) \rangle$ where $\hat{h}_{l+1}$ is a 'monic' R-lift of h_{l+1}. Then $(\hat{I}_{l+1}, f_{I_{l+1}})$ is added to S where $f_{I_{l+1}} := f_{I_l}(\bar{x}_l, x_{l+1} + px) \bmod \hat{I}_{l+1}$.

Clearly $\hat{I}_{l+1}$ is a triangular ideal with monic generators: it follows first part of Definition 3.2.1. Since $\tilde{g}(\bar{x}_l, x_{l+1}) \equiv 0 \bmod I_l + \langle h_{l+1} \rangle$, we have $p | g(\bar{x}_l, x_{l+1} + px) \bmod \hat{I}_{l+1} + \langle \hat{h}_{l+1} \rangle$. Thus $f_{I_{l+1}} = p^\alpha g(\bar{x}_l, x_{l+1} + px) =: p^{\alpha'} g' \bmod \hat{I}_{l+1}$ where $\alpha' \geq \alpha + 1 \geq l + 2$. Substituting $f_{I_{l+1}} = f(\sum_{0 \leq i \leq l+1} x_i p^i + x p^{l+2})$ we have $f(\sum_{0 \leq i \leq l+1} x_i p^i) \equiv 0 \bmod \hat{I}_{l+1} + \langle p^{l+2} \rangle$. Thus $\hat{I}_{l+1}$ follows third condition of Definition 3.2.1.

For the second condition for $\hat{I}_{l+1}$ being a split ideal, fix a particular root $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$. Using Lemma 4.1.4, the projection $h_{l+1,\bar{a}}(x)$ equals $\gcd(\tilde{g}_{\bar{a}}(x), x^p - x)$ (up to a unit multiple). By Lemma 3.4.2, h_{l+1} is monic mod I_l; giving $\deg(h_{l+1,\bar{a}}) = \deg_{x_{l+1}}(h_{l+1})$. Since $h_{l+1,\bar{a}} | (x^p - x)$, there are exactly $\deg_x(h_{l+1})$-many $a_{l+1} \in \mathbb{F}_p$, such that $h_{l+1,\bar{a}}(a_{l+1}) \equiv 0 \bmod p$. So, every root $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$ can be extended to $\deg_x(h_{l+1})$-many roots; giving $|\mathcal{Z}_{\mathbb{F}_p}(I_{l+1})| = \deg_x(h_{l+1}) \cdot \prod_{i=0}^{l} \deg_{x_i}(h_i)$. This makes $\hat{I}_{l+1}$ a split ideal.

$\hat{I}_{l+1}$ remains prefix-free with any other ideal $\hat{J}$ of S, because roots of $\hat{I}_{l+1}$ are extension of roots of $\hat{I}_l$ (recall: $\hat{I}_l$ was prefix-free with $\hat{J}$ and it was popped out of S).

This proves all the invariants for the stack S. □

Using the invariant, we prove that Algorithm 5 terminates on any input.

Lemma 4.1.6 (Algorithm 5 always terminates)**.** *Algorithm 5 finishes in finite number of steps for any $f \in \mathbb{Z}[x]$ and a prime power p^k.*

Proof. We show that the number of iterations in Algorithm 5 are finite. Assume that all the ideals which result in a dead-end are moved to a list D; say C is the disjoint union of all ideals in S, $\mathcal{L}$ and D. Whenever a split ideal $\hat{I}$ from S is moved to $\mathcal{L}$ or D, the underlying roots (of $\hat{I}$) stop extending to the next precision. Togetherwith Lemma 4.1.5, we deduce that in fact all the ideals in C are prefix-free and split ideals. Now by Step 18, and the rate of growth of split ideals up to length $l+1 \leq k$, we get a lazy estimate of $|C| \leq \min(d^k, p^k)$.

Let $\text{len}(\hat{I})$ denote the length of an ideal $\hat{I}$, it is bounded by k (Lemma 4.1.5). Notice that factoring/growing an ideal increases $\sum_{\hat{I}\in C} \text{len}(\hat{I})$; and getting a maximal split ideal/ dead-end increases $|\mathcal{L}| + |D|$. Thus every iteration of the algorithm strictly increases the quantity $(\sum_{\hat{I}\in C} \text{len}(\hat{I})) + |\mathcal{L}| + |D|$. By the estimate on $|C|$, all the terms in this quantity are bounded; thus the number of iterations are finite. □

The following lemma shows that if there is an ideal $\hat{I}_l \in S$ which represents a root $r \in \mathcal{Z}_R(f)$ modulo p^{l+1} then at a later point of time there will be $\hat{I}_{l+1} \in S$ representing r modulo p^{l+2} (higher precision).

Lemma 4.1.7. *Let Algorithm 5 pops an ideal $\hat{I}_l$ of length $l+1$, not a maximal split ideal, which represents a root $r \in \mathcal{Z}_R(f)$ modulo p^{l+1} i.e., there is an $\bar{a}_l := (a_0, \ldots, a_l) \in \mathcal{Z}_R(\hat{I}_l)$ such that $r \equiv a_0 + pa_1 + \cdots + p^l a_l \bmod p^{l+1}$. Then at a later point of time Algorithm 5 pops an ideal $\hat{I}_{l+1}$ (of length $l+2$) representing r modulo p^{l+2} i.e., there is a unique $a_{l+1} \in R$ such that $(\bar{a}_l, a_{l+1}) \in \mathcal{Z}_R(\hat{I}_{l+1})$ and $r \equiv a_0 + \cdots + a_{l+1}p^{l+1} \bmod p^{l+2}$.*

Let us verify the claim of Lemma 4.1.5 on Example 9.

Example 10. *Recall Example 9: $f = x^4 - 8x^3 + 7x^2$ and $R = \mathbb{Z}/\langle 7^5\rangle$. Consider a root $r = 350 = 0 + 7\cdot 1 + 7^2 \cdot 0 + 7^3 \cdot 1 + 7^4 \cdot 0$ of $f \bmod 7^5$.*
At first iteration: $\hat{I}_0 = \langle x_0^2 - x_0\rangle$ is popped and $(0) \in \mathcal{Z}_R(\hat{I}_0)$ represents r as $350 \equiv 0 \bmod 7$.
At third iteration: $\hat{I}_1 = \langle x_0, x_1^2 - x_1\rangle$ is popped and $(0,1) \in \mathcal{Z}_R(\hat{I}_1)$ represents r as $350 \equiv 0 + 7\cdot 1 \bmod 7^2$.

At sixth iteration: $\hat{I}_{2,1} = \langle x_0, x_1 - 1, x_2 \rangle$ *is popped and* $(0,1,0) \in \mathcal{Z}_R(\hat{I}_{2,1})$ *represents* r *as* $350 \equiv 0 + 7 \cdot 1 + 7^2 \cdot 0 \bmod 7^3$. *Now* $\hat{I}_{2,1}$ *becomes a maximal split ideal so it represents* r.

Proof of Lemma 4.1.7. We again consider three possible situations.

(Step 18) The ideal $\hat{I}_l$ grows to another split ideal $\hat{I}_{l+1}$ as follows: Let $f_{I_l} =: p^\alpha g(\bar{x}_l, x) \bmod \hat{I}_l$, where $p \nmid g$ and $\alpha \geq l+1$ (Lemma 4.1.5), and $\tilde{g} = g \bmod p$. Algorithm 5 computes $h_{l+1} := gcd(\tilde{g}, x^p - x) \bmod I_l$ where $I_l := \hat{I}_l \bmod p$. Then it defines $\hat{I}_{l+1} := \hat{I}_l + \langle \hat{h}_{l+1}(\bar{x}_l, x_{l+1}) \rangle$ where $\hat{h}_{l+1}$ is a 'monic' R-lift of h_{l+1}.

Looking at the f_I analogues pushed in Steps $3, 18, 20$, one can deduce the invariant: $f\left(\sum_{0 \leq i \leq l} x_i p^i + x p^{l+1}\right) \equiv f_I(\bar{x}_l, x) \bmod \hat{I}$. Substituting $\bar{x}_l = \bar{a}_l$ we get $f(a_0 + \cdots + p^l a_l + p^{l+1} x) \equiv p^\alpha g(\bar{a}_l, x) \bmod p^k$. Notice that $p^\alpha g(\bar{a}_l, x) \not\equiv 0 \bmod p^k$ as $\alpha < k$ ($\because \hat{I}_l$ is not maximal) and $p \nmid g(\bar{a}_l, x)$. This is because leading coefficient of $g \bmod p$ (with respect to x) is a unit (due to check at Step-9).

Write $r \in \mathcal{Z}_R(f)$ in the unique representation $r = r_0 + p r_1 + \cdots + p^{k-1} r_{k-1}$ where each $r_i \in \{0, \ldots, p-1\}$. We know that $\sum_{0 \leq i \leq l} r_i p^i \equiv \sum_{0 \leq i \leq l} a_i p^i \bmod p^{l+1}$, thus $(\sum_{0 \leq i \leq l} a_i p^i - \sum_{0 \leq i \leq l} r_i p^i) = b_{l+1} p^{l+1}$ for some $b_{l+1} \in R$.
So $f\left(\sum_{0 \leq i \leq l} a_i p^i + x p^{l+1}\right) \equiv f\left(\sum_{0 \leq i \leq l} r_i p^i + (b_{l+1} + x) p^{l+1}\right) \equiv p^\alpha g(\bar{a}_l, x) \bmod p^k$.

Let $\tilde{a}_{l+1} \in \mathbb{F}_p$ such that $b_{l+1} + \tilde{a}_{l+1} = r_{l+1} \bmod p$ then $\tilde{g}(\bar{a}_l, \tilde{a}_{l+1}) = 0 \bmod p$ (thus $\tilde{g}(\bar{a}_l, x)$ has degree at least 1). It implies $h_{l+1}(\bar{a}_l, \tilde{a}_{l+1}) = 0 \bmod p$ (due to Lemma 4.1.4). Since $\tilde{a}_{l+1}$ is a root of $h_{l+1}(\bar{a}_l, x)$ of multiplicity 1, hence there is a unique $a_{l+1} \in R$ such that $\hat{h}_{l+1}(\bar{a}_l, a_{l+1}) = 0 \bmod p^k$ and $\tilde{a}_{l+1} \equiv a_{l+1} \bmod p$ (Hensel Lemma 2.2.2). Hence $(\bar{a}_l, a_{l+1}) \in \hat{I}_{l+1}$.

(Step 16) Proof of the previous case shows that $h_{l+1}(\bar{a}, x)$ has degree at least 1, so $\hat{I}_l$ could not result in a *dead-end*.

(Step 20) Ideal $\hat{I}_l$ factors into (smaller) split ideals. In this case, $\bar{a}_l$ will be included in exactly one of those ideals (Lemma 4.1.5). This ideal will be handled later in the algorithm and will give an ideal $\hat{I}_{l+1}$ with $(\bar{a}_l, a_{l+1})$ as root. □

Now, we can prove Theorem 4.1.3.

Proof of Theorem 4.1.3. Lemma 4.1.5 states that at any point, Stack S only contains split ideals which have disjoint root sets. Lemma 4.1.6 assures that Algorithm 5 terminates on every input. So both these lemmas and Definition 3.2.7 of maximal split ideal makes it clear that Algorithm 5 returns a list $\mathcal{L}$ containing maximal split ideals $\hat{I}_1, \ldots, \hat{I}_n$, for $n \in \mathbb{N}$. Further, we show:

1) The root-set of $\hat{I}_j$ $(1 \leq j \leq n)$ yields a subset S_j of $\mathcal{Z}_R(f)$, and they are pairwise disjoint.

2) Given a root $r \in \mathcal{Z}_R(f)$, there exists j such that r is represented by a root in $\mathcal{Z}_R(\hat{I}_j)$.

For the first part, root-sets for different maximal split ideals $\hat{I}_j$ are pairwise disjoint because of Lemma 4.1.5. Each of these root-set yields a subset of the zero-set of $f \bmod p^k$ (follows from the definition of maximal split ideal).

For the second part, let r be a root in $\mathcal{Z}_R(f)$ which has unique representation $r_0 + \cdots + r_{k-1}p^{k-1}$ where $r_i \in \{0, \ldots, p-1\}$. Stack S was initialized by the R-lift $\hat{I}_0$ of the ideal $I_0 = \langle h_0 := \gcd(f(x_0) \bmod p, x_0^p - x_0)\rangle$; so $r_0 \in \mathcal{Z}_{\mathbb{F}_p}(I_0)$, as $f(r_0) \equiv f(r) \equiv 0 \bmod p$. Thus there must be a unique $\hat{r}_0 \in \mathcal{Z}_R(\hat{I}_0)$ such that $\hat{r}_0 \equiv r_0 \bmod p$ (Hensel Lemma 2.2.2).

Assume that $\hat{I}_0$ is not a maximal split ideal (otherwise we are done). Applying Lemma 4.1.7, there must exist an $\hat{I}_1$ whose root-set contains $(\hat{r}_0, \hat{r}_1)$ such that $r_0 + pr_1 \equiv \hat{r}_0 + p\hat{r}_1 \bmod p^2$. Repeated applications of Lemma 4.1.7 show that we will keep getting split ideals of larger lengths, partially representing r; finally, reaching a maximal split ideal (say $\hat{I}_j$) fully representing r. In other words, if length of $\hat{I}_j$ is $l+1$ then $r \equiv \hat{r}_0 + \cdots + \hat{r}_l p^l \bmod p^{l+1}$ and since for all $\hat{r}_{l+1} \in R$, $\hat{r}_0 + \cdots + \hat{r}_l p^l + \hat{r}_{l+1}p^{l+1} \in \mathcal{Z}_R(f)$ so we have a unique $\hat{r}_{l+1} \in R$ such that $r = \hat{r}_0 + \cdots + \hat{r}_l p^l + \hat{r}_{l+1}p^{l+1}$.

We showed that each root r of $f \bmod p^k$ is represented by a *unique* maximal split ideal $\hat{I}$, given by Algorithm 5, and they collectively represent exactly the roots of f modulo p^k. Hence, root-sets of ideals in $\mathcal{L}$ partition the zero-set $\mathcal{Z}_R(f)$. □

4.1.3 Time Complexity Analysis: Introducing the Roots-Tree *RT*

We know that Algorithm 5 takes finite amount of time and terminates (Lemma 4.1.6). To show that it is efficient, note that the time complexity of the algorithm can be divided into two parts.

1) Number of iterations taken by Algorithm 5, which is clearly bounded by the number of updates on Stack S in the algorithm.

2) Time taken by the various algebraic operations in one iteration of the algorithm: reduction by a triangular ideal, valuation computation modulo a split ideal, testing if some polynomial is a zerodivisor modulo a split ideal, performing repeated squaring modulo a triangular ideal and computing gcd of two multi-variates modulo a split ideal.

For the purpose of bounding iterations, we define a data structure we call a *roots-tree (RT)* which essentially keeps track of the updates on Stack S. We will map an element $(\hat{I}, f_I)$ in stack S to a node $N_I := (\hat{I}, f_I)$ in the roots-tree. Each *push* will create a new node in RT. The nodes are *never* deleted from RT.

Construction of roots-tree (RT):

1. Denote the root of RT by $N_{\langle 0\rangle} := (\langle 0\rangle, f_{\langle 0\rangle} := f(x))$. Add a child node N_{I_0} to the root corresponding to the initialization of Stack S (Steps 1-3) by $(\hat{I}_0, f_{I_0})$, where $\hat{I}_0 := \langle \hat{h}_0(x_0)\rangle$ (label the edge by $\hat{h}_0$ in RT).

2. If, at some time t, the algorithm pops $(\hat{I}_{l-1}, f_{I_{l-1}})$ from S then the *current node* in RT will be the leaf node $N_{I_{l-1}} = (\hat{I}_{l-1}, f_{I_{l-1}})$. For the next three points, assume the current node is $N_{I_{l-1}} = (\hat{I}_{l-1}, f_{I_{l-1}})$.

3. If the current ideal $\hat{I}_{l-1}$ *grows* to $\hat{I}_l := \hat{I}_{l-1} + \langle \hat{h}_l\rangle$ (Step 18) and $(\hat{I}_l, f_{I_l})$ is pushed in S, then create a child of the current node $N_{I_{l-1}}$ in RT using an edge labelled $\hat{h}_l$ (label the node $N_{I_l} := (\hat{I}_l, f_{I_l})$).

4. (Step 16) If the algorithm reached *dead-end* (no update in stack S or list $\mathcal{L}$), then add a child labelled $\mathcal{D}$ to the current node $N_{I_{l-1}}$. It indicates a dead-end at the current branch. Analogously, if the algorithm finds a *maximal split ideal* (Step 6), we add a child labelled $\mathcal{M}$ to the current node $N_{I_{l-1}}$ (indicating $\hat{I}_{l-1}$ is a maximal split ideal).

5. (Step 20) Suppose, processing of the current split ideal $\hat{I}_{l-1}$ of length-l results in factoring each ideal $\hat{U}$ in S, containing $\hat{h}_i$, to m split ideals. We describe the *duplication process* for a particular $\hat{U}$ (repeat it for each split ideal containing $\hat{h}_i$).

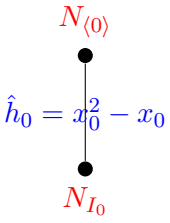
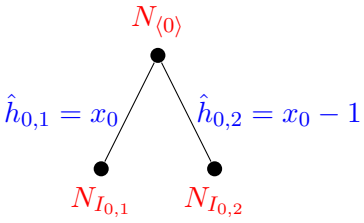
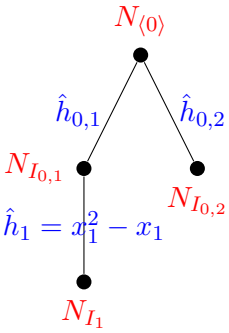


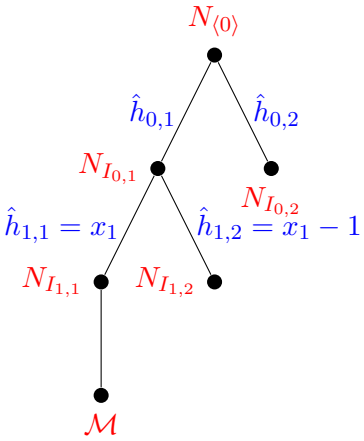
Figure 4.1: Initialization



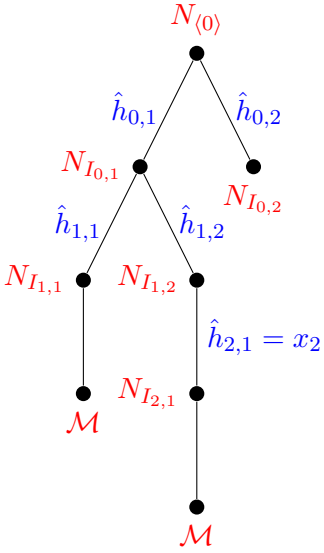
After first iteration



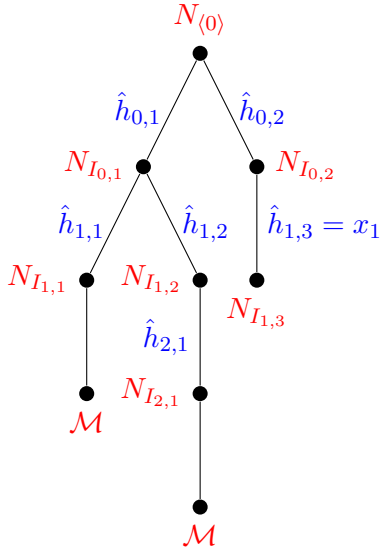
After second iteration



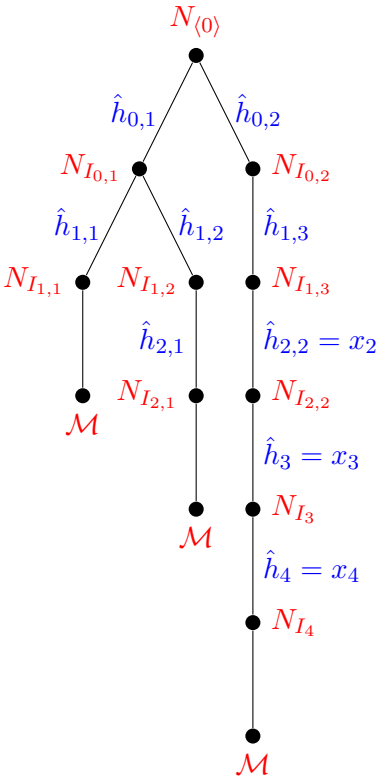
After fourth iteration



After sixth iteration



After seventh iteration



Final roots-tree RT

Figure 4.2: Construction of roots-tree RT for Example 9

Let $\hat{U}_{i-1}$ be the length-i restriction of $\hat{U}$. First, we move to the ancestor node $N_{U_{i-1}} := (\hat{U}_{i-1}, f_{U_{i-1}})$ of N_U. Make m copies of the sub-tree at Node $N_{U_{i-1}}$, each of them attached to $N_{U_{i-1}}$ by edges labelled with $\hat{h}_{i,1}, \ldots, \hat{h}_{i,m}$ respectively. The copy of each old node $N = (\hat{V}, f_V)$, in sub-tree corresponding to $\hat{h}_{i,j}$, will be relabelled with $(\hat{V}_j, f_{V_j})$ corresponding to the factor split ideal $\hat{V}_j$ of $\hat{V}$ and the newly computed f_{V_j}.

This step does not increase the height of the tree, though it increases the size.

For the rest of this section, RT denotes the final roots-tree created at the end of the above process.

Example 11. *Figure 4.1, 4.2 explains the construction of RT for Example 9. Figure 4.2 gives the final roots-tree RT constructed at the end of Algorithm 5 on Example 9.*

Properties of RT: We state some easy properties of RT, which will help us in analyzing the time complexity. These can be easily verified on Example 9 and Figure 4.2.

1) By construction, size of the roots-tree increases at every iteration. We never delete a node or an edge (though relabelling might be done). So, the size of RT bounds the number of iterations taken by Algorithm 5.

2) Consider a node $N_I =: (\hat{I}, f_I)$ in RT such that its child is not $\mathcal{M}$ or $\mathcal{D}$. Here $f_I(\bar{x}_l, x) \in R[\bar{x}_l, x]$, and let $g_I \in R[\bar{x}_l, x]$ be defined as in Algorithm 5 (Step-7), $f_I(\bar{x}_l, x) =: p^\alpha g_I \bmod \hat{I}$, where $p^\alpha \mid\mid f_I \mod \hat{I}$. Then $g_I \bmod I$ $(I := \hat{I} + \langle p \rangle)$ is a non-constant polynomial, with respect to x, over $\mathbb{F}_p$.

3) For each node $N_I =: (\hat{I}, f_I(\bar{x}_l, x))$ and its child $N_J =: (\hat{J}, f_J(\bar{x}_{l+1}, x))$, we have the relation, $f_J = f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$.

Bounding $|RT|$**:** To bound the size of RT, we define a parameter for a node N of RT, called the *degree* of the node N and denoted by $[N]$. Intuitively, degree of a node $N_I := (\hat{I}, f_I(\bar{x}_l, x))$ is the modulo p degree of the 'effective' part (part free from p-powers) of f_I times the number of zeroes in $\mathcal{Z}_R(\hat{I})$ (which is $\deg(I)$).

Definition 4.1.8 (Degree of a node in RT)**.** *The degree of a node in RT is defined as follows:*

1. *The degree of root node* $N_{\langle 0 \rangle}$ *is* $[N_{\langle 0 \rangle}] := d$ $(= \deg(f) \bmod p)$*. The degree of leaves* $\mathcal{D}$ *and* $\mathcal{M}$ *is defined to be* 1.

2. *If a node* $N_I = (\hat{I}, f_I)$ *is parent of a leaf* $\mathcal{D}$ *or* $\mathcal{M}$ *then* $[N_I] := \deg(\hat{I})$.

3. *For any other node* $N_I =: (\hat{I}, f_I)$, *where* $\hat{I} \subseteq R[\bar{x}_l]$ *and* $f_I \in R[\bar{x}_l, x]$, *let* $f_I =: p^\alpha g_I(\bar{x}_l, x) \bmod \hat{I}$ *such that* $p^\alpha \mid\mid f_I \bmod \hat{I}$. *Then* $[N_I] := \deg_x(g_I \bmod I) \times \deg(\hat{I})$ *where* $I := \hat{I} \bmod p$.

We show that the degree of a parent node bounds the sum of the degree of its children.

Lemma 4.1.9 (Degree distributes in *RT*)**.** *Let* N *be a node in roots-tree* RT *and* $des(N)$ *denote the set of all children of* N. *Then* $[N] \geq \sum_{C \in des(N)} [C]$.

So, the sum of the degrees of all nodes, at any level l, *is at least the sum of the degrees of all nodes at level* $l+1$.

Following example illustrates Definition 4.1.8 and Lemma 4.1.9.

Example 12. *Recall Example 9 and the associated final roots-tree* RT *in Figure 4.2. The degree of root node* $N_{\langle 0 \rangle}$ *is* 4 *as the degree of* $f = x^4 - x^3 - 7x^3 + 7x^2$ *is* 4 *modulo* 7. *Its two children are* $N_{I_{0,1}} = (\hat{I}_{0,1}, f_{I_{0,1}})$ *and* $N_{I_{0,2}} = (\hat{I}_{0,2}, f_{I_{0,2}})$. *The degree of both the ideals* $\hat{I}_{0,1} = \langle x_0 \rangle$ *and* $\hat{I}_{0,2} = \langle x_0 - 1 \rangle$ *is clearly* 1.

Since $f_{I_{0,1}} = 7^\alpha \cdot g_{I_{0,1}}(x_0, x) = 7^3 \cdot [7x^4 - 8x^3 + x^2] \bmod \hat{I}_{0,1} + \langle 7^5 \rangle$, *we have* $[N_{I_{0,1}}] = \deg(\hat{I}_{0,1}) \times \deg_x(g_{I_{0,1}} \bmod \hat{I}_{0,1} + \langle 7 \rangle) = 3$. *Similarly since* $f_{I_{0,2}} = 7 \cdot [7^3 x^4 - 4 \cdot 7^2 x^3 - 11 \cdot 7x^2 - 6x] \bmod \hat{I}_{0,2} + \langle 7^5 \rangle$ *so* $[N_{I_{0,2}}] = 1 \times 1 = 1$. *Hence,* $[N_{\langle 0 \rangle}] = [N_{I_{0,1}}] + [N_{I_{0,2}}]$.

Further, $N_{I_{0,1}}$ *has two child* $N_{I_{1,1}}$ *and* $N_{I_{1,2}}$. *The degree of both the ideals* $\hat{I}_{1,1} = \langle x_0, x_1 \rangle$ *and* $\hat{I}_{1,2} = \langle x_0, x_1 - 1 \rangle$ *is clearly* 1. *So by Definition 4.1.8 (point 2),* $[N_{I_{1,1}}] = 1$. *Since* $f_{I_{1,2}} = 7^4 \cdot (-22)x \bmod \hat{I}_{1,2} + \langle 7^5 \rangle$, *we have* $[N_{I_{1,2}}] = 1$. *Thus* $[N_{I_{0,1}}] = 3 > 2 = [N_{I_{1,1}}] + [N_{I_{1,2}}]$.

Similarly, we can verify the claim of Lemma 4.1.9 for the degree of other nodes.

Proof of Lemma 4.1.9. Let $N = (\hat{I}, f_I)$, where $\hat{I} = \langle \hat{h}_0, \ldots, \hat{h}_l \rangle$ and $f_I(\bar{x}_l, x) \in R[\bar{x}_l, x]$ such that $f_I = p^\alpha g(\bar{x}_l, x) \bmod \hat{I}$. We also have $I := \langle h_0, \ldots, h_l \rangle$ where $h_i := \hat{h}_i \bmod p$. Define $\tilde{g}_I \in \mathbb{F}_p[\bar{x}_l, x]$ as $\tilde{g}_I := g_I(\bar{x}_l, x) \bmod I$. Assume $\alpha < k$, otherwise we are done. So, $\tilde{g}_I$ is non-trivial with respect to x; by Step 9 (failure) leading coefficient with respect to x of $\tilde{g}_I$ is a unit modulo I so we get,

$$\forall \bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I) : \ \deg_x(\tilde{g}_I \bmod I) \ = \ \deg_x(\tilde{g}_I(\bar{a}, x)) \ . \tag{4.1}$$

Recall $h_{l+1}(\bar{x}_l, x) := \gcd(\tilde{g}_I(\bar{x}_l, x), x^p - x)$. Let C be a child node of N in RT such that $C =: (\hat{J}_C, f_{J_C})$, where $\hat{J}_C =: \hat{I} + \langle \hat{h}_{l,C}(\bar{x}_{l+1}) \rangle$ and $f_{J_C}(\bar{x}_{l+1}, x) := f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}_C$ (by property 3 of RT). We also have $h_{l,C} := \hat{h}_{l,C} \bmod p$ and $J_C = I + \langle h_{l,C} \rangle$. This gives us the factorization $h_{l+1}(\bar{x}_l, x) = \prod_{C \in \mathrm{des}(N)} h_{l,C}(\bar{x}_l, x) \bmod I$ (Step 20, and 'duplication step' when we constructed RT). Again,

$$\forall \bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(J_C) : \ \deg_x(\tilde{g}_{J_C} \bmod J_C) = \deg_x(\tilde{g}_{J_C}(\bar{b}, x)) \ . \tag{4.2}$$

If $g_{J_C} =: f_{J_C}/p^{v'} \bmod \hat{J}_C$ for some $v' \in \mathbb{N}$, by property 3 of RT, we have $g_{J_C} = f_I(\bar{x}_l, x_{l+1} + px)/p^{v'} \bmod \hat{J}_C$.

By definition, $[N] = \deg(I) \cdot \deg_x(\tilde{g}_I)$ and $[C] = \deg(J_C) \cdot \deg_x(\tilde{g}_{J_C})$ ($\because \deg(\hat{I}) = \deg(I)$). Since $\deg(J_C) = \deg(I) \cdot \deg_x(h_{l,C}(\bar{x}_l, x))$, the lemma statement is equivalent to showing,

$$\deg_x(\tilde{g}_I) \geq \sum_{C \in \mathrm{des}(N)} \deg_x(h_{l,C}(\bar{x}_l, x)) \cdot \deg_x(\tilde{g}_{J_C}) \ . \tag{4.3}$$

Continuing with the notation of a particular child C, fix an $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I)$. Since $\hat{J}_C$ is a split ideal so by Corollary 3.2.3, $h_{l,C}(\bar{a}, x)$ (of degree d'_C) can be written as $\prod_{i=1}^{d'_C}(x - c_i)$, where each $c_i \in \mathbb{F}_p$ and are distinct. Then each c_i is also a root of $\tilde{g}_I(\bar{a}, x)$, say with multiplicity $m_i \in \mathbb{N}$. So, there exists $G(x) \in \mathbb{F}_p[x]$ (coprime to $x - c_i$), such that, $\tilde{g}_I(\bar{a}, x) \equiv (x - c_i)^{m_i} \cdot G(x) \bmod p$. Lifting this equation mod p^k, there exists $G_1(x) \in R[x]$, of degree less than m_i, and a unique 'monic' lift $G_2(x) \in R[x]$ of $G(x)$ (Hensel lemma (2.2.2)) : $g_I(\bar{a}, x) \equiv ((x - c_i)^{m_i} + pG_1(x)) \cdot G_2(x) \bmod p^k$. Substituting $x \to c_i + px$, we get, $g_I(\bar{a}, c_i + px) \equiv ((px)^{m_i} + pG_1(c_i + px)) \cdot G_2(c_i + px) \bmod p^k$.

Let $\bar{b}_i = (\bar{a}, c_i) \in \mathcal{Z}_{\mathbb{F}_p}(J_C)$. We know that $\tilde{g}_{J_C}(\bar{b}_i, x) = f_I(\bar{a}, c_i + px)/p^{v'} \bmod p$ is non-trivial (otherwise $[C] = 1$ as it will be parent of $\mathcal{D}$ or $\mathcal{M}$). This implies that, $((px)^{m_i} + pG_1(c_i + px))/p^{v'} \bmod p$ is a nonzero polynomial of degree at most m_i ($\because G_2(c_i + px) \equiv G_2(c_i) \not\equiv 0 \bmod p$ so $p \nmid G_2(c_i)$).

Since $G_2(c_i + px) \not\equiv 0 \bmod p$ is a unit, $\deg_x(\tilde{g}_{J_C}(\bar{b}_i, x)) = \deg_x(\tilde{g}_{J_C}) \leq m_i$ (Eqn. 4.2). Summing up over all the roots c_i of $\tilde{g}_I(\bar{a}, x)$,

$$\sum_{i=1}^{d'_C} \deg_x(\tilde{g}_{J_C}(\bar{b}_i, x)) = d'_C \cdot \deg_x(\tilde{g}_{J_C}) \leq \sum_{i=1}^{d'_C} m_i =: d_C(g_I) \ .$$

Summing over all children $C \in \mathrm{des}(N)$ (using Eqn. 4.1, factorization of h_{l+1} & distinctness of $\mathbb{F}_p$-roots), we deduce,

$$\sum_{C \in \mathrm{des}(N)} \deg_x(h_{l,C}) \deg_x(\tilde{g}_{J_C}) \;\leq\; \sum_{C} d_C(g_I) \;\leq\; \deg_x(\tilde{g}_I(\bar{a}, x)) \;=\; \deg_x(\tilde{g}_I)\,.$$

This proves Eqn. 4.3, and hence the lemma. □

Define the degree of list $\mathcal{L}$ as, $\deg(\mathcal{L}) := \Sigma_{\hat{I} \in \mathcal{L}} \deg(\hat{I})$.

Lemma 4.1.10 (Bounding $|RT|$, $\deg(\hat{I})$, $\deg(\mathcal{L})$, $|\mathcal{L}|$)**.** *Let RT be the roots-tree constructed from the execution of Algorithm 5. The number of leaves of RT, respectively $\deg(\mathcal{L})$, is at most $d = \deg(f(x))$. Also, the size $|RT|$ of the roots-tree (hence, the number of iterations by Algorithm 5) is bounded by dk.*

Proof. Applying Lemma 4.1.9 inductively, sum of the degrees of nodes at any level is bounded by the degree d of the root node. In particular,

1) We can extend every leaf to bring it to the last level (create a chain of nodes of same degree) without changing the degree distribution property. So, $\deg(\mathcal{L}) = \Sigma_{\hat{I} \in \mathcal{L}} \deg(\hat{I}) \leq d$. Since the number of leaves is $\geq |\mathcal{L}|$ (leaves are either $\mathcal{D}$ or $\mathcal{M}$), we get $|\mathcal{L}| \leq d$.

2) For any split ideal $\hat{I}$ in stack S, $\deg \hat{I} \leq d$.

3) Since the depth of the roots-tree is at most k, $|RT| \leq kd$. □

Lemma 4.1.11 (Computation cost at a node)**.** *Computation cost at each node of RT (time taken by Algorithm 5 in every iteration of the while loop) is bounded by $\mathsf{poly}(d, k \log p)$.*

Proof. During an iteration, the major computations performed by the algorithm are— testing for zerodivisors (Step 9), computing modular gcd (Step 13), computing reduced f_I (Steps 3, 18), performing reduction for repeated squaring (Step 12), and factoring ideals (Step 20).

These operations are described by Lemmas 3.3.2, 3.3.3, 4.1.1, 3.4.1 and 3.4.2. All of them take time $\mathsf{poly}(d, k \log p, \deg(\hat{I}))$, where $\hat{I}$ is the concerned triangular ideal.

For any split ideal $\hat{I}$ (or its reduction I), we know that $\deg(\hat{I}) \leq d$ (Lemma 4.1.10). So, Steps 3, 9, 13, 18, 20 take time $\mathsf{poly}(d, k \log p)$. Step 12 to compute repeated squaring modulo

$I + \langle \tilde{g} \rangle$ takes time $\mathsf{poly}(\deg_x(\tilde{g}), \deg(I),\ k\ \log p)$ (using Lemma 3.3.2). Since $\deg(I) \leq d$, and degree of $\tilde{g}$ is at most d, so Step 12 also takes $\mathsf{poly}(d, k \log p)$ time.

Hence the computation cost at each node is $\mathsf{poly}(d, k \log p)$. □

Proof of Theorem 4.0.1. The definition of roots-tree shows that the number of leaves upper bound the number of all maximal split ideals in $\mathcal{L}$. Lemmas 4.1.10 and 4.1.11 show that the time complexity of Algorithm 5 is bounded by $\mathsf{poly}(d, k \log p)$ (by bounding both number of iterations and the cost of computation at each iteration). Using Lemma 3.2.8 on the output of Algorithm 5, we get the exact count on the number of roots of $f \bmod p^k$ in time $\mathsf{poly}(d, k \log p)$. □

4.2 Summary

We saw the de-randomization of the algorithm of [BLQ13] via the algebraic tool split ideals. The algorithm for deterministic root-counting goes similar to randomized root-counting with extra work on the implementation with the help of ideals. These ideals will further help in solving more problems deterministically in later chapters.

Chapter 5

Counting Unramified Factors modulo Prime Powers

In this chapter, we extend the ideas for counting roots to count all the basic-irreducible factors of a given $f \in \mathbb{Z}[x]$ modulo p^k. Recall that a basic-irreducible factor of $f \bmod p^k$ is an irreducible factor that remains irreducible modulo p. This can be seen as the first significant move towards deterministically counting all the irreducible factors of $f \bmod p^k$.

Theorem 5.0.1 (Factor count)**.** *Let p be a prime, $k \in \mathbb{N}$ and $f(x) \in \mathbb{Z}[x]$. Then all the basic-irreducible factors of $f \bmod p^k$ can be counted in deterministic poly($\deg f, k \log p$)-time.*

We achieve this by extending the idea of counting roots to more general p-adic integers. Essentially, we efficiently count all the roots of $f(x)$ in $\mathcal{O}_K/\langle p^k \rangle$, where $\mathcal{O}_K$ is the ring of integers of a p-adic *unramified* extension $K/\mathbb{Q}_p$.

Corollary 5.0.2. *Let $K/\mathbb{Q}_p$ be an unramified p-adic extension of degree Δ and $\mathcal{O}_K$ be its ring of integers. Given $f(x) \in \mathbb{Z}[x]$, Δ, and prime power p^k (in binary) as input, we can count all the roots of f, in $\mathcal{O}_K/\langle p^k \rangle$, in deterministic poly($\deg(f), k \log p, \Delta$)-time.*

5.1 Counting Factors with Strong Irreducibility

A polynomial f can be factored mod p^k if it has two basic-irreducible factors of different degree (using distinct degree factorization [vzGP01] and Hensel Lemma 2.2.2). Further, if

two basic-irreducible factors appear with different exponents/multiplicities, then again f can be factored (using formal derivatives [vzGP01] and Hensel Lemma 2.2.2). So, for factoring $f \bmod p^k$, we can assume $f \equiv (\varphi_1 \dots \varphi_t)^e + ph \bmod p^k$, where every $\varphi_i \in (\mathbb{Z}/\langle p^k \rangle)[x]$ is a basic-irreducible polynomial of a fixed degree b. Also, $d := \deg(f) = bte$. Let us fix this assumption for this section, unless stated explicitly.

A basic-irreducible factor of $f \bmod p^k$ has the form $\varphi_i + pw_i(x) \bmod p^k$, for $i \in [t]$ (Lemma 2.2.3). If $b = 1$, counting basic-irreducible factors of f is equivalent to counting roots of f. When $b > 1$, we prove a simple generalization of this idea; it is enough to count all the roots of f in the ring extension $\mathbb{Z}[y]/\langle p^k, \varphi(y) \rangle$, where $\varphi(y)$ is an irreducible mod p of degree b. Recall that these rings are called *Galois rings* denoted by $\mathbb{G}(p^k, b)$ (unique, for fixed k and b, up to isomorphism).

5.1.1 Reduction to Root Counting in $\mathbb{G}(p^k, b)$

By Lemma 2.2.3, any basic-irreducible factor of $f \bmod p^k$ is a factor of a unique $({\varphi_i}^e + pw_i(x))$; and φ_i are coprime mod p. So in this subsection, for simplicity of exposition, we will assume that $f(x)$ equals $\varphi^e \bmod p$ (φ is a monic degree-b irreducible mod p).

Define $\mathbb{G} := \mathbb{G}(p^k, b)$. Let $y_0, y_1, \dots, y_{b-1}$ be the roots of $\varphi(x)$ in $\mathbb{G}$ (Proposition 1). Without loss of generality, take $y := y_0$, $y_i \equiv y^{p^i} \bmod p$, for all $i \in \{0, \dots, b-1\}$ (Frobenius conjugates in $\mathbb{F}_p$).

The lemma below associates a root of f, in $\mathbb{G}$, to a unique basic-irreducible factor of f in $(\mathbb{Z}/\langle p^k \rangle)[x]$.

Lemma 5.1.1 (Root to factor)**.** *Let $r(y) \in \mathbb{G}$ be a root of $f(x)$. Then*

$$h(x) := \prod_{i=0}^{b-1} (x - r(y_i))$$

is the unique basic-irreducible factor of f having root $r(y)$. We say that $h(x)$ is the basic-irreducible factor associated to root $r(y)$.

Proof. The coefficients of h are symmetric polynomials in $r(y_i)$ (over $0 \le i < b$). Since the automorphism $\psi_1 : y \to y_1$ of $\mathbb{G}$ (as defined in Proposition 2) permutes $r(y_i)$'s ($\because$ it permutes

y_i's), it fixes all the coefficients of h. From Proposition 2, all these coefficients are then in $\mathbb{Z}/\langle p^k\rangle$. Hence, $h \in (\mathbb{Z}/\langle p^k\rangle)[x]$.

If $r(y)$ is a root of another polynomial h' in $(\mathbb{Z}/\langle p^k\rangle)[x]$, then $r(y_i)$'s are also roots of h' (applying automorphisms ψ_i of $\mathbb{G}$). Since these roots are coprime mod p, we actually get: $h|h'$. Thus h is the unique monic irreducible factor of f containing $r(y)$.

Looking mod p, the $r(y_i)$'s are a permutation of the roots of $\varphi(x)$, so $h(x) \equiv \varphi(x) \bmod p$. Hence, $h(x)$ is the unique monic basic-irreducible factor of f having root $r(y)$. □

Following is the reduction to counting all roots of f in $\mathbb{G}$.

Theorem 5.1.2 (Factor to root)**.** *Any degree-b basic-irreducible factor of $f \bmod p^k$ has exactly b roots in $\mathbb{G}$. Conversely, if f has a root $r(y) \in \mathbb{G}$, then it must be a root of a unique degree-b basic-irreducible factor of $f \bmod p^k$.*

So, the number of degree-b basic-irreducible factors of $f \bmod p^k$ is exactly the number of roots, of f in $\mathbb{G}$, divided by the degree b.

Proof. By Lemma 5.1.1 (and uniqueness of Galois rings), for every root $r(y) \in \mathbb{G}$ of f, we can associate a unique basic-irreducible factor of $f(x)$.

Conversely, let $h(x) =: \varphi(x) + pw(x)$ be a basic-irreducible factor of $f(x)$. It splits completely in $\mathbb{G}$ (as, $h(x) \equiv \varphi \bmod p$; first factor in $\mathbb{G}/\langle p\rangle$ and then Hensel lift to $\mathbb{G}$). So, h has exactly b roots in $\mathbb{G}$, each of them is also a root of f in $\mathbb{G}$.

Hence the theorem statement follows. □

Remark. This 'irreducible factor vs root' correspondence, for $f \bmod p^k$, breaks down if $\mathbb{G}$ is *not* a Galois ring.

5.1.2 Counting Roots in $\mathbb{G}(p^k, b)$

In this section, we show how to count the roots of $f \equiv (\varphi_1\varphi_2\ldots\varphi_t)^e + ph(x) \bmod p^k$ in $\mathbb{G}(p^k, b)$. Since $\mathbb{G} := \mathbb{G}(p^k, b)$ is a Galois ring, so $\mathbb{G}/\langle p\rangle = \mathbb{F}_{p^b} =: \mathbb{F}_q$. (Recall: $R = \mathbb{Z}/\langle p^k\rangle$.)

Split ideals and zero-sets in the Galois ring: First, we will modify the definition of zero-sets (Section 3.1) to include zeros of f in $\mathbb{G}$. A $\mathbb{G}$-*zero-set* of $f(x) \in R[x]$ will be defined

as $\mathcal{Z}_{\mathbb{G}}(f) := \{r \in \mathbb{G} \mid f(r) \equiv 0 \bmod p^k\}$. Similarly, for an ideal $I \subseteq \mathbb{F}_p[\bar{x}_l]$, its $\mathbb{F}_q$-zero-set is defined as $\mathcal{Z}_{\mathbb{F}_q}(I) := \{\bar{a} = (a_0, \ldots, a_l) \in (\mathbb{F}_q)^{l+1} \mid g(\bar{a}) \equiv 0 \bmod p^k, \forall g \in I\}$. Similarly, for $\hat{I} \subseteq R[\bar{x}_l]$, $\mathcal{Z}_{\mathbb{G}}(\hat{I}) := \{\bar{a} = (\hat{a}_0, \ldots, \hat{a}_l) \in (\mathbb{G})^{l+1} \mid g(\bar{a}) \equiv 0 \bmod p^k, \forall g \in \hat{I}\}$.

The definition of triangular ideals, split ideals and maximal split ideals will remain exactly same (generators defined over R, Section 3.2), except that in the third condition for split ideals, zero-set will be over $\mathbb{G}$ instead of R. They can now be seen as storing potential roots of $f(x)$ in $\mathbb{G}$ (or, storing potential basic irreducible factors of $f \bmod p^k$). The reason is, a root $r(y) \in \mathbb{G}$ of $f \bmod p^k$ can be viewed 'uniquely' as, $r(y) = r_0(y) + pr_1(y) + p^2 r_2(y) + \ldots + p^{k-1} r_{k-1}(y)$, where each $r_i(y) \in \mathbb{G}$ such that $\deg_y(r_i) < \deg_y(\varphi(y))$ and coefficients with respect to y are in $\{0, \ldots, p-1\}$. So, the decomposition of formal variable $x =: x_0 + px_1 + p^2 x_2 + \ldots + p^{k-1} x_{k-1}$, now represents candidates for r_0, r_1, and so on, over $\mathbb{G}$.

A split ideal $\hat{I}_l \subseteq R[\bar{x}_l]$, defined as $\hat{I}_l := \langle \hat{h}_0(x_0), \ldots, \hat{h}_l(\bar{x}_l) \rangle$, now implicitly stores the candidates as follows: for (r_0) there exists $(\hat{r}_0) \in \mathcal{Z}_R(\hat{h}_0)$: $r_0 \equiv \hat{r}_0 \bmod p$, for (r_0, r_1) there exists $(\hat{r}_0, \hat{r}_1) \in \mathcal{Z}_R(\hat{h}_1)$: $r_0 + pr_1 \equiv \hat{r}_0 + p\hat{r}_1 \bmod p^2$, and so on. These, in turn, give candidates for basic-irreducible factors of $f \bmod p^{l'}$ (some $l' \leq k$).

In particular, when $\hat{I}_l$ is a maximal split ideal, an $(\hat{r}_0, \ldots, \hat{r}_l)$ implicitly denote a basic-irreducible factor of $f \bmod p^k$. The number of such factors is $\deg(\hat{I}_l) \cdot q^{k-l-1}/b$ (thanks to Theorem 5.1.2 and Lemma 3.2.8).

Split ideals follow all the properties given in Section 3.2, just by replacing the fact that roots belong to $\mathbb{G}$ and not R.

Description of the modified algorithm: Algorithm 5, to count roots in R, extends directly to count roots in $\mathbb{G}$. The algorithm is exactly same except one change: to compute GCD (Steps 3 and 13), we now use the Frobenius polynomial $x^q - x$ instead of the prior $x^p - x$ (GCD computation implicitly stores the candidate roots truncated modulo p, they are in $\mathbb{F}_q$ now).

So the algorithm works as follows:

1. It gets $f(x) \equiv (\varphi_1 \ldots \varphi_t)^e + pw(x) \bmod p^k$ as input, computes gcd $h_0(x) := \gcd(f(x), x^q - x)$ over $\mathbb{F}_p$. Since $x^q - x$, over $\mathbb{F}_p$, is the product of all irreducible factors of degree dividing b, we deduce: $h_0(x) = \varphi_1 \ldots \varphi_t \bmod p$; and define the first split ideal $\hat{I}_0 := \langle \hat{h}_0 \rangle$

where $\hat{h}_0$ is a 'monic' lift of h_0 over R. (Note: We do not have access to φ_i's themselves.)

Remark. The length 1 split ideal stores all the roots of f in $\mathbb{G}/\langle p \rangle$, or all the basic irreducible factors of $f \bmod p$; as $h_0(x) = \varphi_1 \dots \varphi_t$. Also, its degree is tb, which when divided by b, gives the count of the basic-irreducible factors of $f \bmod p$.

2. The algorithm then successively looks for the next precision candidates. It computes h_l by taking gcd with $x^q - x$, and adds its monic lift $\hat{h}_l \subseteq R[\bar{x}_l]$ to the previous ideal $\hat{I}_{l-1} \subseteq R[\bar{x}_l]$ like before.

3. All the supporting algebraic algorithms and lemmas (given in Sections 3.3, 3.4 and 4.1.1) work the same as before; since they are being passed the same parameters— a split ideal, or a triangular ideal, or a polynomial over R.

Thus a similar proof of correctness and time complexity can be given as before.

Proof of Theorem 5.0.1. Consider a univariate $f(x) \bmod p^k$. As discussed in the beginning of this section, $f \bmod p^k$ can be efficiently factorized as $f \equiv \prod_{i=1}^{m} f_i \bmod p^k$, where each $f_i(x)$ is a power of a product of degree-b_i irreducible polynomials mod p (i.e., of the form $\equiv (\varphi_1 \varphi_2 \dots \varphi_t)^e + ph(x)$, where φ_j is a degree-b_i irreducible mod p).

On each such $f_i \bmod p^k$, we use Algorithm 5 with the new Frobenius polynomial $(x^{q_i} - x)$ $(q_i = p^{b_i})$, in Steps 3 and 18, as discussed above. Let the final list output, for $f_i \bmod p^k$, be $\mathcal{L}_i =: \{\hat{I}_1(l_1, D_1), \dots, \hat{I}_n(l_n, D_n)\}$. Thus we get the count on the $\mathbb{G}(p^k, b_i)$-roots of $f_i \bmod p^k$ as $\Sigma_{j=1}^{n} D_j q_i^{k-l_j}$ (Lemma 3.2.8). Using Theorem 5.1.2, the number of the degree-b_i basic-irreducible factors of $f \bmod p^k$ is $B_k(f_i) := (1/b_i) \times \Sigma_{j=1}^{n} D_j q_i^{k-l_j}$.

Using Lemma 2.2.3, we get the count on the basic-irreducible factors of $f \bmod p^k$ as, $B_k(f) = \Sigma_{i=1}^{m} B_k(f_i)$.

For the time complexity, only difference is the repeated-squaring to compute the reduced form of polynomial $x^{q_i} - x$ (Steps 3, 12), it will take $b_i \log p$ operations instead of $\log p$ operations. But $b_i \leq d$, so the algorithm runs in time $\mathsf{poly}(d, k \log p)$ (and remains deterministic).

□

Now, we can prove Corollary 5.0.2.

Proof of Corollary 5.0.2. The proof of Corollary 5.0.2 comes with a slight change in the proof of Theorem 5.0.1.

We are given an integral univariate $f(x) \in \mathbb{Z}[x]$, a degree of extension Δ and a prime power p^k (in binary) and we are required to count all the roots of f in $\mathcal{O}_K/\langle p^k \rangle$ for some unknown p-adic extension $K/\mathbb{Q}_p$ of degree Δ.

As in the proof of Theorem 5.0.1, assume $f \equiv \prod_{i=1}^{m} f_i \bmod p^k$, where each $f_i(x)$ is a power of a product of degree-b_i irreducible polynomials mod p (i.e., of the form $\equiv (\varphi_1\varphi_2 \dots \varphi_t)^e + ph(x)$, where φ_j is a degree-b_i irreducible mod p).

Note that, for an unramified extension $K/\mathbb{Q}_p$, $\mathcal{O}_K/\langle p^k \rangle \cong \mathbb{G}(p^k, \Delta)$ and so $\mathcal{O}_K/\langle p \rangle \cong \mathbb{F}_q$ where $q = p^\Delta$. Hence any root of $f(x)$ in $\mathcal{O}_K/\langle p^k \rangle$ corresponds to a unique f_i (using Lemma 2.2.3) and so finding roots of each f_i in $\mathcal{O}_K/\langle p^k \rangle$ individually is enough.

Any f_i can have a root in $\mathbb{G}(p^k, \Delta)$ only if $f_i \equiv (\varphi_1\varphi_2 \dots \varphi_t)^e \bmod p$ has a root in $\mathbb{F}_q$. Since $x^q - x$ is product of all irreducible factors (over $\mathbb{F}_p$) of degrees dividing Δ hence, f_i has a root in $\mathcal{O}_K/\langle p^k \rangle$ only when $b_i | \Delta$.

So, we can discard in advance those f_is for which $b_i \nmid \Delta$. To find roots of f_i in $\mathbb{G}(p^k, \Delta)$ we follow the root-counting algorithm as in proof of Theorem 5.0.1 (Sec. 5.1.2) with just one difference: each time GCD is taken with forbenius polynomial $x^q - x$ instead of $x^{q_i} - x$ (recall $q = p^\Delta$). This is the main difference with the proof of Theorem 5.0.1 where root-counting was done in $\mathbb{G}(p^k, b_i)$ for f_i opposite to root-counting in same Galois-ring $\mathbb{G}(p^k, \Delta)$ here for all f_i (s.t. $b_i \nmid \Delta$). Collectively the sum of the count on the roots of all f_is, for $i \in [m]$, gives exactly the count on the roots of f in $\mathcal{O}_K/\langle p^k \rangle$.

For the time complexity, only difference with the proof of Theorem 5.0.1 is the repeated-squaring to compute the reduced form of polynomial $x^q - x$, it will take $\log q = \Delta \log p$ operations, so the algorithm runs in time $\mathsf{poly}(d, k \log p, \Delta)$ (and remains deterministic).

□

5.2 Discussions

In this chapter we saw the application of split ideals in efficient deterministic counting of (1) basic irreducible factors of $f(x)$ mod p^k and, (2) roots of $f(x)$ over unramified extension of p-adic integers mod p^k. The definition of split ideals naturally extends to count roots in higher extensions. We will further see in next chapter that the parameters of split ideals like their length and degree and even their structural evolution during their construction quite naturally provide us crucial information to compute rational form of an important generating function in arithmetic geometry– Igusa local zeta function. Similar to the techniques of extended Hensel's lemma, we will connect the roots of $f \in \mathbb{Z}[x]$ modulo p^k for large enough k to p-adic roots of f to give an elementary proof of convergence of Igusa zeta function for univariate polynomials and to get them in rational function form.

Chapter 6

Computing Igusa's Local Zeta Function and p-adic Applications

In this chapter, we will see two applications of split ideals– (1) Computing Igusa's local zeta function associated to univariate polynomials and, (2) counting all the p-adic ($\mathbb{Z}_p$) roots with *multiplicity* of a univariate polynomial. The former has appeared in [DS20] and the latter application is a direct consequence of the new methods used in [DS20] (unmentioned there).

We are interested in computing rational function form of Igusa's local zeta function associated to a univariate polynomial $f \in \mathbb{Z}[x]$ and prime p. Essentially, the function encodes root count of f modulo prime powers as evident from the definition of the equivalent Poincaré series $P(t) := \sum_{i=0}^{\infty} N_i(f) \cdot (p^{-n}t)^i$ where $N_i(f)$ is the root count of $f \bmod p^i$ ($N_0(f) := 1$) and $t \in \mathbb{C}$ with $|t| < 1$. We will compute the Igusa zeta function for f by finding polynomials $A(t)$ and $B(t)$ such that $P(t) =: A(t)/B(t)$.

Theorem 6.0.1. *We are given a univariate integral polynomial $f(x) \in \mathbb{Z}[x]$ of degree d, with coefficients magnitude bounded by $C \in \mathbb{N}$, and a prime p. Then, we compute Poincaré series $P(t) = A(t)/B(t)$, associated with f and p, in deterministic poly$(d, \log C + \log p)$ time.*

The degree of the integral polynomial $A(t)$ is $\tilde{O}(d^2 \log C)$ and that of $B(t)$ is $O(d)$.

We achieve the rational form of $P(t)$ by getting an explicit formula for the number of zeros $N_k(f)$, of $f \bmod p^k$, which sheds new light on the properties of the function $N_k(\cdot)$.

Eventually, it gives an elementary proof of the rationality of the Poincaré series $P(t)$.

Corollary 6.0.2. *Let k be large enough, namely, $k \geq k_0 := O(d^2(\log C + \log d))$. Then, we give a closed form expression for $N_k(f)$ (in Theorem 6.4.1).*

Interestingly, if f has non-zero discriminant, then $N_k(f)$ is constant (independent of k) for all $k \geq k_0$.

The new insight on the properties of the function $N_k(\cdot)$ has more consequences. As a corollary of Theorem 6.0.1 we give the first deterministic polynomial-time algorithm to count *all* the $\mathbb{Z}_p$-roots of f.

Corollary 6.0.3. *We are given a univariate integral polynomial $f(x) \in \mathbb{Z}[x]$ of degree d, with coefficients magnitude bounded by $C \in \mathbb{N}$, and a prime p. Then, we can count all the p-adic integral roots (in $\mathbb{Z}_p$) of f in deterministic poly$(d, \log C + \log p)$-time.*

6.1 Preliminaries

Recall the definition of representatives and representative roots from Chapter 2. Throughout the chapter, we denote them using bold small letters, like $\mathbf{a}, \mathbf{b}$ etc. We denote the *length* of a representative $\mathbf{a}$ by $|\mathbf{a}|$, so if $\mathbf{a} := a_0 + pa_1 + \ldots + p^{l-1}a_{l-1} + p^l *$ then its length is $|\mathbf{a}| = l$.

Also recall the definition of a maximal split ideal I from Chapter 3. Essentially, I is encoding some representative-roots of f mod p^k in the form of common roots of its generators. If $(a_0, \ldots, a_l)$ is a common zero of the generators then $a_0 + pa_1 + \ldots + p^l a_l + p^{l+1} *$ follows all the conditions to be a representative-root. So we restate some results of the previous chapters in the following Lemma 6.1.1 and Theorem 6.1.2 in slightly modified form to be used in later sections of this chapter.

Lemma 6.1.1 (Lemma 3.2.5 & 3.2.8)**.** *The length of an MSI I is the length of each representative-root encoded by it and the degree of I is the count on them. Thus, we get the count on the roots of f mod p^k encoded by I as $\prod_{i=0}^{l} \deg_{x_i}(h_i) \times p^{k-l-1}$.*

We restate the result of which returns all the representative-roots, in MSI form, in deterministic polynomial time.

Theorem 6.1.2. (Compute $N_k(f)$) *In deterministic poly($|f|, k \log p$)-time one gets the maximal split ideals which collectively contain exactly the representative-roots of a univariate polynomial $f(x) \in \mathbb{Z}[x]$ modulo prime-power p^k.*

Moreover, using Lemma 6.1.1 we can count them, and all the roots of $f \bmod p^k$, in deterministic poly-time.

6.1.1 Some Definitions and Notations related to f

We are given an integral univariate polynomial $f(x) \in \mathbb{Z}[x]$ of degree d with coefficients magnitude at most $C \in \mathbb{N}$, and a prime p. Then, f can also be thought of as an element of $\mathbb{Z}_p[x]$ (as $\mathbb{Z} \subseteq \mathbb{Z}_p$).

Now we define the factors of f in $\mathbb{Z}_p[x]$ as follows (note: we do not require f to be monic).

Definition 6.1.3. *Let the p-adic integral factorization of f, into coprime irreducible factors, be*

$$f(x) \;=:\; \prod_{i\in[n]} (x-\alpha_i)^{e_i} \cdot \prod_{j=1}^{m} g_j(x)^{t_j}$$

where each α_i is a $\mathbb{Z}_p$-root of f with multiplicity *e_i. Each $g_j(x) \in \mathbb{Z}_p[x]$ has multiplicity t_j; it is irreducible over $\mathbb{Z}_p$ and has no $\mathbb{Z}_p$-root.*

For example, over $\mathbb{Z}_2$, $f = 2x^2 + 3x + 1 = (x+1)\cdot(2x+1)$ has $n = m = 1$.

Definition 6.1.4. *For each $i \in [n]$, we define $f_i(x) \in \mathbb{Z}_p[x]$, called* α_i-free part of f, *as $f_i(x) := f(x)/(x-\alpha_i)^{e_i}$. We denote* valuation *$v_p(f_i(\alpha_i))$ as ν_i, for all $i \in [n]$.*

Definition 6.1.5. *Define $\mathsf{Rad}(f) := (\prod_{i=1}^{n}(x-\alpha_i)) \cdot (\prod_{j=1}^{m} g_j(x))$. Analogously, the radical of f_i, for each $i \in [n]$, is defined as $\mathsf{Rad}(f_i) := \mathsf{Rad}(f)/(x-\alpha_i)$.*

Definition 6.1.6. *We denote by Δ, the valuation with respect to p of the discriminant of radical of f, i.e, $\Delta := v_p(D(\mathsf{Rad}(f)))$.*

We see that Δ must be finite, since roots of $\mathsf{Rad}(f)$ are distinct. The following fact is easily established by the definition of discriminant and the fact that $\alpha_1, \ldots, \alpha_n$ are also roots of $\mathsf{Rad}(f)$.

Fact 1. *For $i \neq j \in [n]$, we have $v_p(\alpha_i - \alpha_j) \leq \Delta/2 < \infty$.*

For our algorithm, Δ will be crucial in informing us about the behavior of the roots of $f \bmod p^k$.

Properties of Discriminant:

1. Over $\mathbb{Z}_p$, if $u(x)|w(x)$ then $D(u) \mid D(w)$ and $v_p(D(u)) \leq v_p(D(w))$.
2. Discriminant of a linear polynomial is defined to be 1.
3. If $w(x) = (x-a) \cdot u(x)$ then by the definition of discriminant, it is clear that $D(w) = D(u) \cdot u(a)^2$.
4. Discriminant $D(h)$ of a degree-l univariate polynomial $h(x) := h_l x^l + \ldots + h_1 x + h_0$, over $\mathbb{Z}_p$, is also a multivariate polynomial over $\mathbb{Z}_p$ in the coefficients $h_0, \ldots, h_l$ (see [LN94, Chapter 1]). Moreover, it is computable in time polynomial in size of given h using determinant of a Sylvester matrix [vzGG13, Chapter 11]).

6.2 Interplay of $\mathbb{Z}_p$-Roots and $\left(\mathbb{Z}/\langle p^k \rangle\right)$-Roots

In this section we will establish a connection between $\left(\mathbb{Z}/\langle p^k \rangle\right)$-roots and $\mathbb{Z}_p$-roots of the given f, when k is sufficiently large i.e, $k > d\Delta$ (see Section 6.1.1 for the related notation).

Recall that $\alpha_1, \ldots, \alpha_n$ are the distinct $\mathbb{Z}_p$-roots of f (Definition 6.1.3). The following claim establishes a notion of 'closeness' of any $\bar{\alpha} \in \mathbb{Z}_p$ to an α_j. Later we will apply this to a representative-root $\bar{\alpha}$.

Claim 6.2.1 (Close to a root)**.** *For some $j \in [n], \bar{\alpha} \in \mathbb{Z}_p$, if $v_p(\bar{\alpha} - \alpha_j) > \Delta/2$, then $v_p(\bar{\alpha} - \alpha_i) = v_p(\alpha_j - \alpha_i) \leq \Delta/2$, for all $i \neq j, i \in [n]$.*

Proof. $v_p(\bar{\alpha} - \alpha_i) = v_p(\bar{\alpha} - \alpha_j + \alpha_j - \alpha_i)$. Since $v_p(\bar{\alpha} - \alpha_j) > \Delta/2$ and $v_p(\alpha_j - \alpha_i) \leq \Delta/2$ (by Fact 1), we deduce, $v_p(\bar{\alpha} - \alpha_i) = \min\{v_p(\bar{\alpha} - \alpha_j), v_p(\alpha_j - \alpha_i)\} = v_p(\alpha_j - \alpha_i) \leq \Delta/2$. □

The following lemma says that an irreducible can not take values with ever increasing valuation.

Lemma 6.2.2 (Valuation of an irreducible)**.** *Let $h(x) \in \mathbb{Z}_p[x]$ be a polynomial with no $\mathbb{Z}_p$-root, and discriminant $D(h) \neq 0$. Then, for any $\bar{\alpha} \in \mathbb{Z}_p$, $v_p(h(\bar{\alpha})) \leq v_p(D(h))$.*

Proof. We give the proof by contradiction i.e, we show that if $v_p(h(\bar{\alpha})) > v_p(D(h))$, then $h(x)$ has a root in $\mathbb{Z}_p$.

Define $v_p(D(h)) =: d(h)$. Let $\bar{\alpha} \in \mathbb{Z}_p$ such that $h(\bar{\alpha}) \equiv 0 \bmod p^\delta$, for $\delta > d(h)$. Then we write, $h(x) = (x - \bar{\alpha}) \cdot h_1(x) \ + \ p^\delta \cdot h_2(x)$. The two things to note here are:

(1). $D(h) \equiv D(h \bmod p^\delta) \bmod p^\delta$ by discriminant's Property (4) in Section 6.1.1. Also, $D(h) \neq 0$ is given.

(2). Let $h'(x)$ be the first derivative of $h(x)$ and let $i := v_p(h'(\bar{\alpha}))$. Then, we claim that $\delta > d(h) \geq 2i$.

Consider $h'(x) = h_1(x) + (x - \bar{\alpha})h_1'(x) + p^\delta h'_2(x)$. So, $h'(\bar{\alpha}) \equiv h_1(\bar{\alpha}) \bmod p^\delta$. By Property (3) (Section 6.1.1) of discriminants, $D(h) \equiv D((x - \bar{\alpha}) \cdot h_1(x)) \equiv D(h_1) \cdot h_1(\bar{\alpha})^2 \equiv D(h_1) \cdot h'(\bar{\alpha})^2 \bmod p^\delta$. Togetherwith $D(h) \neq 0 \bmod p^\delta$, we deduce, $2i \leq d(h) < \delta$.

Now, we show that the root $\bar{\alpha}$ of $h \bmod p^\delta$ *lifts* to roots of $h \bmod p^{\delta+j}$, for all $j \in \mathbb{Z}^+$. This is due to Hensel's Lemma (see [vzGG13, Chapter 15]); for completeness we give the proof.

By Taylor expansion, we have $h(\bar{\alpha} + p^{\delta-i}x) = h(\bar{\alpha}) + h'(\bar{\alpha}) \cdot p^{\delta-i}x + h''(\bar{\alpha}) \cdot p^{2(\delta-i)}x^2/2! + \cdots$.

Note that there exists a unique solution $x_0 \equiv (-h(\bar{\alpha})/h'(\bar{\alpha})p^{\delta-i}) \bmod p : h(\bar{\alpha} + p^{\delta-i}x_0) \equiv 0 \bmod p^{\delta+1}$. This follows from the Taylor expansion and since $2(\delta - i) > \delta$.

So, $\bar{\alpha} - p^{\delta-i}(h(\bar{\alpha})/h'(\bar{\alpha})p^{\delta-i}) \bmod p^{\delta+1}$ is a lift, of $\bar{\alpha} \bmod p^\delta$. By a similar reasoning, it can be lifted further to arbitrarily high powers $p^{\delta+j}$. Thus, proving that $h(x)$ has a $\mathbb{Z}_p$-root; which is a contradiction. □

The following lemma is perhaps the most important one. It associates every root $\bar{\alpha}$ of $f(x) \bmod p^k$ to a unique $\mathbb{Z}_p$-root of f. Recall the notation from Section 6.1.1.

Lemma 6.2.3 (Unique association)**.** *Let $k > d(\Delta + 1)$ and $\bar{\alpha} \in \mathbb{Z}_p$ be a root of $f(x) \bmod p^k$. There exists a unique α_i such that $v_p(\bar{\alpha} - \alpha_i) > \Delta + 1$ and thus, $v_p(\bar{\alpha} - \alpha_i) > v_p(\alpha_i - \alpha_j)$, for all $j \neq i, j \in [n]$.*

Proof. Let us first prove that there exists some $i \in [n]$, given $\bar{\alpha}$, such that $v_p(\bar{\alpha} - \alpha_i) > \Delta + 1$. For the sake of contradiction, assume that $v_p(\bar{\alpha} - \alpha_i) \leq \Delta + 1$ for all $i \in [n]$. Then, by

Definition 6.1.3, $v_p(f(\bar{\alpha})) = \sum_{i=1}^{n} e_i \cdot v_p(\bar{\alpha} - \alpha_i) + \sum_{j=1}^{m} t_j \cdot v_p(g_j(\bar{\alpha})) \leq (\Delta + 1) \cdot \sum_{i=1}^{n} e_i + \sum_{j=1}^{m} t_j \cdot v_p(g_j(\bar{\alpha}))$.

Since g_j has no $\mathbb{Z}_p$-root, for all $j \in [m]$, by Lemma 6.2.2, $v_p(g_j(\bar{\alpha})) \leq v_p(D(g_j))$. By the properties given in Section 6.1.1 we get: $v_p(D(g_j)) \leq v_p(D(\mathsf{Rad}(f))) = \Delta$; proving that $v_p(g_j(\bar{\alpha})) \leq \Delta$.

Going back, $v_p(f(\bar{\alpha})) \leq (\Delta + 1) \cdot (\sum_{i=1}^{n} e_i + \sum_{j=1}^{m} t_j) \leq d(\Delta + 1) < k$. It implies that $f(\bar{\alpha}) \not\equiv 0 \bmod p^k$; which contradicts the hypothesis that $\bar{\alpha}$ is a root of $f \bmod p^k$.

Thus, $\exists i \in [n]$, $v_p(\bar{\alpha} - \alpha_i) > \Delta + 1$. The uniqueness of i follows from Claim 6.2.1. □

Having seen that every root $\bar{\alpha}$, of $f \bmod p^k$, is associated (or close) to a unique $\mathbb{Z}_p$-root α_i, the following lemma tells us that the valuation of α_i-free part of f (respectively factors of f with no $\mathbb{Z}_p$-root) is the *same* on any $\bar{\alpha}$ close to α_i. This unique valuation is important in getting an expression for $N_k(f)$.

Lemma 6.2.4 (Unique valuation)**.** *Fix $i \in [n]$. Fix $\bar{\alpha} \in \mathbb{Z}_p$ such that $v_p(\bar{\alpha} - \alpha_i) > \Delta$. Recall $g_j(x), f_i$ from Section 6.1.1. Then,*

1. *$v_p(g_j(\bar{\alpha})) = v_p(g_j(\alpha_i))$, for all $j \in [m]$,*

2. *$v_p(f_i(\bar{\alpha})) = v_p(f_i(\alpha_i))$.*

In other words, valuation with respect to p of $f_i = f(x)/(x - \alpha_i)^{e_i}$, on $x \mapsto \bar{\alpha}$, is fixed uniquely to $\nu_i := v_p(f_i(\alpha_i))$, for any 'close' approximation $\bar{\alpha} \in \mathbb{Z}_p$ of α_i.

Proof. Since $g_j \mid \mathsf{Rad}(f_i)$ and $\mathsf{Rad}(f_i) \mid \mathsf{Rad}(f)$, we have by the properties of discriminants (Section 6.1.1): $v_p(g_j(\alpha_i)) \leq v_p(\mathsf{Rad}(f_i)(\alpha_i)) \leq \Delta$, for all $j \in [m]$.

Since $v_p(\bar{\alpha} - \alpha_i) > \Delta$, we deduce $v_p(g_j(\bar{\alpha}) - g_j(\alpha_i)) > \Delta$. Furthermore, $v_p(g_j(\alpha_i)) \leq \Delta$ implies: $v_p(g_j(\bar{\alpha})) = v_p(g_j(\alpha_i))$. This proves the first part.

By Claim 6.2.1, $v_p(\bar{\alpha} - \alpha_u) = v_p(\alpha_i - \alpha_u)$, for all $u \neq i, u \in [n]$. Also, by the first part, $v_p(g_w(\bar{\alpha})) = v_p(g_w(\alpha_i))$, for all $w \in [m]$. Consequently, $v_p(f_i(\bar{\alpha})) = \sum_{u=1, u\neq i}^{n} e_u \cdot v_p(\alpha_i - \alpha_u) + \sum_{w=1}^{m} t_w \cdot v_p(g_w(\alpha_i)) = v_p(f_i(\alpha_i))$. This proves the second part. □

6.3 Representative Roots versus Neighborhoods

We now connect the $\mathbb{Z}_p$-roots of f to the representative-roots of $f \bmod p^k$. Later we characterize each representative-root as a 'neighborhood' in Theorem 6.3.3.

Lemma 6.3.1 (Perturb a root). *Let $k > d(\Delta+1)$ and $\bar{\alpha}$ be a root of $f(x) \bmod p^k$ with $l := v_p(\alpha_i - \bar{\alpha}) > \Delta + 1$, for some $i \in [n]$ (as in Lemma 6.2.3). Then, every $\bar{\beta} \in \bar{\alpha} + p^l*$ is also a root of $f(x) \bmod p^k$.*

Proof. Since $f(\bar{\alpha}) \equiv 0 \bmod p^k$, we have $v_p(f(\bar{\alpha})) \geq k$. Using Lemma 6.2.4 we have $v_p(f_i(\bar{\alpha})) = v_p(f_i(\alpha_i)) = \nu_i$. Thus, $v_p(f(\bar{\alpha})) = v_p(\alpha_i - \bar{\alpha}) \cdot e_i + v_p(f_i(\bar{\alpha})) = v_p(\alpha_i - \bar{\alpha}) \cdot e_i + \nu_i \geq k$.

Similarly, $v_p(f(\bar{\beta})) = v_p(\alpha_i - \bar{\beta}) \cdot e_i + v_p(f_i(\bar{\beta})) = v_p(\alpha_i - \bar{\beta}) \cdot e_i + \nu_i \geq v_p(\alpha_i - \bar{\alpha}) \cdot e_i + \nu_i$. Last inequality follows from $v_p(\alpha_i - \bar{\beta}) \geq l = v_p(\alpha_i - \bar{\alpha})$.

From the above two paragraphs we get, $v_p(f(\bar{\beta})) \geq k$. Hence, $f(\bar{\beta}) \equiv 0 \bmod p^k$. □

Now we define a notion of 'neighborhood' of a $\mathbb{Z}_p$-root of f.

Definition 6.3.2 (Neighborhood). *For $i \in [n]$, $k > d(\Delta+1)$, we define* neighborhood $S_{k,i}$ *of α_i mod p^k to be the set of all those roots of $f \bmod p^k$, which are close to the $\mathbb{Z}_p$-root α_i of f. Formally,*

$$S_{k,i} := \{\bar{\alpha} \in \mathbb{Z}/\langle p^k \rangle \mid v_p(\bar{\alpha} - \alpha_i) > \Delta + 1, f(\bar{\alpha}) \equiv 0 \bmod p^k\} .$$

The notion of representative-root was first given in [DMS21]. Below we discover its new properties which will lead us to the understanding of *length* of a representative-root; which in turn will give us the size of a neighborhood contributing to $N_k(f)$.

Theorem 6.3.3 (Rep.root is a neighborhood). *Let $k > d(\Delta+1)$ and $\mathbf{a} := a_0 + pa_1 + p^2a_2 + \ldots + p^{l-1}a_{l-1} + p^l*$ be a representative-root of $f(x) \bmod p^k$. Define the $\mathbb{Z}_p$-root reduction $\bar{\alpha}_i := \alpha_i \bmod p^k$, for all $i \in [n]$. Fix an $i \in [n]$, then,*

1. *Length of* $\mathbf{a}$ *is* large. *Formally, $l > \Delta + 1$.*
2. *If $\bar{\alpha}_i \in \mathbf{a}$, then $\bar{\alpha}_j \notin \mathbf{a}$ for all $j \neq i, j \in [n]$. (This means with Lemma 6.2.3: $\mathbf{a}$ has a* uniquely associated $\mathbb{Z}_p$*-root.)*

3. *If* $\mathbf{a}$ *contains* $\bar{\alpha}_i$ *then it also contains the respective* neighborhood. *In fact, if* $\bar{\alpha}_i \in \mathbf{a}$, *then* $S_{k,i} = \mathbf{a}$.

Proof. 1. Consider $\bar{\alpha} := a_0 + pa_1 + \ldots + p^{l-1}a_{l-1}$. By Lemma 6.2.3, there is a unique $s \in [n]$: $v_p(\bar{\alpha} - \alpha_s) > \Delta + 1$. Suppose $l \leq \Delta + 1$. Then, $v_p(\bar{\alpha} + p^{\Delta+1} - \alpha_s) = \Delta + 1$. As, $\bar{\alpha}' := (\bar{\alpha} + p^{\Delta+1})$ is also in $\mathbf{a}$, it again has to be close to a unique α_t, $s \neq t \in [n]$: $v_p(\bar{\alpha}' - \alpha_t) > \Delta + 1$. In other words, $\alpha_s + p^{\Delta+1} \equiv \bar{\alpha} + p^{\Delta+1} \equiv \alpha_t \bmod p^{\Delta+2}$. Thus, $v_p(\alpha_s - \alpha_t) = \Delta + 1 > \Delta/2$; contradicting Fact 1. This proves $l > \Delta + 1$.

2. Consider distinct $\bar{\alpha}_i, \bar{\alpha}_j \in \mathbf{a}$. Then, by the definition of $\mathbf{a}$, we have $v_p(\bar{\alpha}_i - \bar{\alpha}_j) \geq l > \Delta + 1 > \Delta/2$; contradicting Fact 1. Thus, there is a unique i.

3. Suppose there exists a neighborhood element $\bar{\beta} \notin \mathbf{a}$, satisfying the conditions $v_p(\alpha_i - \bar{\beta}) > \Delta + 1$ and $f(\bar{\beta}) \equiv 0 \bmod p^k$. Let j be the index of the first coordinate where $\bar{\beta}$ and $\mathbf{a}$ differ; so, $j < l$ since $\bar{\beta} \notin \mathbf{a}$. Clearly, $j > \Delta + 1$; otherwise, since $\bar{\alpha}_i \in \mathbf{a}$ and $\bar{\beta} \notin \mathbf{a}$, we deduce $v_p(\alpha_i - \bar{\beta}) = j \leq \Delta + 1$; which is a contradiction.

   By $v_p(\alpha_i - \bar{\beta}) = j > \Delta + 1$ and Lemma 6.3.1, we get: every element in $\bar{\beta} + p^j *$ is a root of $f(x) \bmod p^k$. Consequently, each element in $a_0 + pa_1 + p^2a_2 + \ldots + p^{j-1}a_{j-1} + p^j *$ is a root of $f(x) \bmod p^k$; which contradicts that $\mathbf{a}$ is a representative-root ($\because j < l$, see Section 2.3.1 in Chapter 2). Thus, $\bar{\beta} \in \mathbf{a}$; implying $S_{k,i} \subseteq \mathbf{a}$.

   Conversely, consider $\bar{\alpha} \in \mathbf{a}$. Then, as before, $v_p(\bar{\alpha}_i - \bar{\alpha}) \geq l > \Delta + 1$; implying that $\bar{\alpha} \in S_{k,i}$. Thus, $S_{k,i} \supseteq \mathbf{a}$..

□

Next, we get the expression for the length of a representative-root.

Theorem 6.3.4. *For* $k > d(\Delta + 1)$, *the representative-roots, of* $f(x) \bmod p^k$, *are in a one-to-one correspondence with* $\mathbb{Z}_p$*-roots of* f. *Moreover, the length of the representative-root* $\mathbf{a}$, *corresponding to* α_i, *is* $l_{i,k} := \lceil (k - \nu_i)/e_i \rceil$.

Proof. Every root of $f \bmod p^k$ is in exactly one of the representative-roots. So each reduced $\mathbb{Z}_p$-root $\bar{\alpha}_i := \alpha_i \bmod p^k$ is in a unique representative-root. Thus, by Theorem 6.3.3 parts (2) & (3), we get the one-to-one correspondence as claimed.

Consider a p-adic integer $\bar{\alpha}$ with $v_p(\bar{\alpha}-\alpha_i) =: l_{\bar{\alpha}} > \Delta$. We have the following equivalences: $\bar{\alpha} \in \mathbf{a}$ iff $v_p(f(\bar{\alpha})) \geq k$ iff $v_p((\bar{\alpha}-\alpha_i)^{e_i} \cdot f_i(\bar{\alpha})) \geq k$ iff $e_i l_{\bar{\alpha}} + \nu_i \geq k$ (by Lemma 6.2.4) iff $l_{\bar{\alpha}} \geq \lceil (k-\nu_i)/e_i \rceil = l_{i,k}$.

Write the representative-root corresponding to α_i as $\mathbf{a} =: a_0 + pa_1 + p^2 a_2 + \ldots + p^{l-1} a_{l-1} + p^l *$. Clearly, $l = \min\{l_{\bar{\alpha}} \mid \bar{\alpha} \in \mathbf{a}\} \geq l_{i,k}$. Note that if $l > l_{i,k}$ then by the equivalences we could reduce the length l of the representative-root $\mathbf{a}$; which is a contradiction. Thus, $l = l_{i,k}$. □

6.4 Formula for $N_k(f)$

For large enough k, the earlier section gives us an easy way to count the roots. In fact, we have the following simple formula for $N_k(f)$.

Theorem 6.4.1 (Roots mod p^k)**.** *For $k > d(\Delta+1)$, $N_k(f) = \sum_{i\in[n]} p^{k-\lceil (k-\nu_i)/e_i \rceil}$, where clearly ν_i, e_i and n (as in Sec.6.1.1) are independent of k.*

Proof. Fix $i \in [n]$ and $k > d(\Delta+1)$. By Theorem 6.3.4 we get that in the unique representative-root $\mathbf{a}$, corresponding to α_i mod p^k, the $(k - \lceil (k-\nu_i)/e_i \rceil)$-many higher-precision coordinates could be set arbitrarily from $[0 \ldots p-1]$ (while the rest lower-precision ones are fixed). That gives us the count via contribution for each $i \in [n]$. Moreover, the sum over neighborhoods, for each $i \in [n]$, gives us exactly $N_k(f)$.

Also, note that if $n = 0$ then the count $N_k(f) = 0$. □

Proof of Corollary 6.0.2. Theorem 6.4.1 gives a closed form expression for $N_k(f)$, when $k \geq k_0 := d(\Delta+1)+1 \leq d(2d-1)(\log_p C + \log_p d) + 1$.

For the other part, note that the discriminant $D(f) \neq 0$ iff f is squarefree. In the squarefree case $e_i = 1$, for all $i \in [n]$. By Theorem 6.4.1, $N_k(f) = \sum_{i\in[n]} p^{\nu_i}$; which is independent of k. □

6.5 Computing Poincaré Series

Building up on the ideas of the previous sections, we will show how to deterministically compute Poincaré series $P(t) = \sum_{k=0}^{\infty} N_k(f)(p^{-1}t)^k$, associated to the input $f(x)$, efficiently; thereby proving Theorem 6.0.1. Before that, we need few notation as follows.

Set $k_0 := d(\Delta+1)+1$ so we know by Theorem 6.4.1 that for $k \geq k_0$, $N_k(f) = \sum_{i=1}^{n} N_{k,i}(f)$, where $N_{k,i}(f) := p^{k-\lceil (k-\nu_i)/e_i \rceil}$. For each $i \in [n]$, define k_i to be the least integer such that $k_i \geq k_0$ and $(k_i - \nu_i)/e_i$ is an integer. Then, Poincaré series $P(t)$ can be partitioned into finite and *infinite* sums as,

$$P(t) \;=\; P_0(t) + \sum_{i=1}^{n} P_i(t)$$

where $P_0(t) := \left(\sum_{k=0}^{k_0-1} N_k(f)\cdot(p^{-1}t)^k\right) + \sum_{i=1}^{n}\sum_{k=k_0}^{k_i-1} N_{k,i}(f)\cdot(p^{-1}t)^k$ and $P_i(t) := \sum_{k=k_i}^{\infty} N_{k,i}(f)\cdot(p^{-1}t)^k$.

We now compute the multiplicity e_i by viewing it as the *step* that increments the length, of the representative-root associated to α_i, as k keeps growing above k_0.

Lemma 6.5.1 (Compute e_i)**.** *We can compute the number of $\mathbb{Z}_p$-roots n of f as well as k_i, ν_i and e_i, for each $i \in [n]$, in deterministic poly$(d, \log C + \log p)$ time.*

Proof. By Theorem 6.1.2, we get all the representative-roots of $f \bmod p^k$ implicitly in the form of maximal split ideals (in short we call split ideals). By Lemma 6.1.1, the length of a split ideal is also the length of all the representative-roots represented by it and the degree is the number of representative roots represented by it. Since by Theorem 6.3.4, n is also the number of representative roots of $f \bmod p^k$ for $k \geq k_0$ hence we run the algorithm of Theorem 6.1.2 for $k = k_0$ and sum up the degree of all the split ideals obtained, to get n.

Suppose the split ideal I we find contains a representative-root $\mathbf{a}$ of $f \bmod p^k$ corresponding to α_i, with k_i as defined before. How do we compute k_i? By Theorem 6.3.4 length of $\mathbf{a}$, when $k = k_i$, is $l_{i,k_i} = (k_i - \nu_i)/e_i$. Now, for all $k = k_i+1, k_i+2, \ldots, k_i+e_i$, the length $l_{i,k}$ remains equal to $l_{i,k_i}+1$; while for the next $k = k_i + e_i + 1$, $l_{i,k}$ increments by one.

So we run the algorithm of Theorem 6.1.2, for several $k \geq k_0$. When we find the length incrementing by one, namely, at the two values $k = k_i + 1$ and $k = k_i' := k_i + 1 + e_i$, then we

have found e_i (and k_i). From the equation, $k_i - \nu_i = e_i \cdot l_{i,k_i}$, we also find ν_i.

Suppose the split ideal I we find contains *two* representative-roots $\mathbf{a}$ and $\mathbf{b}$ mod p^k, corresponding to $\mathbb{Z}_p$-roots α_i and α_j respectively, such that $e_i \neq e_j$ (wlog say $e_i < e_j$). In this case, even if $\mathbf{a}$ and $\mathbf{b}$ have the same length, when $k = k_i$, they will evolve to different length representative-roots when we go to a 'higher-precision' arithmetic mod $p^{k_i+1+e_i}$ (by formula in Theorem 6.3.4). So $\mathbf{a}, \mathbf{b}$ must lie in different length split ideals, say I_a and I_b respectively.

Now, for another representative-root $\mathbf{c}$ in I_a, say corresponding to α_s, we have $e_i = e_s$ and hence $\nu_i = \nu_s$. By computing e_i and ν_i as before, now using the length of I and I_a, we compute e_s and ν_s (and k_s) for every $\mathbf{c}$ in I_a. Since, by Lemma 6.1.1, the degree of I_a is the number of such representative-roots in I_a, we could compute n; moreover, we get k_i, ν_i, e_i for all $i \in [n]$.

Clearly, we need to run the algorithm of Theorem 6.1.2 at most $2\max_{i\in[n]}\{e_i\} = O(d)$ times, to study the evolution of split ideals (implicitly, that of the underlying representative-roots). Also Δ is logarithm (to base p) of the determinant of a Sylvester matrix which gives $\Delta = O(d \cdot (\log_p C + \log_p d))$. So, the algorithm is poly-time as claimed. □

Now we prove Corollary 6.0.3.

Proof of Corollary 6.0.3. Recall n is the number of distinct $\mathbb{Z}_p$-roots of f and e_is are the multiplicities of the $\mathbb{Z}_p$-roots α_i for $i \in [n]$. Lemma 6.5.1 computes both of them in required time. Hence the proof.

□

Now we prove that the infinite sums $P_i(t)$ are formally equal to rational functions of $t = p^{-s}$.

Lemma 6.5.2 (Infinite sums are rational)**.** *For each $i \in [n]$, the series $P_i(t)$ is a rational function of t as,*

$$P_i(t) \;=\; \frac{t^{k_i} \cdot (p - t(p-1) - t^{e_i})}{p^{(k_i-\nu_i)/e_i} \cdot (1-t) \cdot (p - t^{e_i})}.$$

Proof. Recall that $P_i(t) = \sum_{k=k_i}^{\infty} N_{k,i}(f) \cdot (p^{-1}t)^k$. For simplicity write $T := p^{-1}t$ and define

integer $\delta_i := k_i - (k_i - \nu_i)/e_i$. Now P_i can be rewritten using residues mod e_i as,

$$P_i(t) \;=\; \sum_{l=k_i}^{k_i+e_i-1} \sum_{k=0}^{\infty} N_{l+ke_i,i}(f) \cdot T^{l+ke_i} \;.$$

For simplicity take $l = k_i$ and consider the sum, $\sum_{k=0}^{\infty} N_{k_i+ke_i,i}(f) \cdot T^{k_i+ke_i}$. We find that $N_{k_i,i}(f) = p^{\delta_i}, N_{k_i+e_i,i}(f) = p^{\delta_i+e_i-1}, N_{k_i+2e_i,i}(f) = p^{\delta_i+2(e_i-1)}$, and so on. Hence,

$$\sum_{k=0}^{\infty} N_{k_i+ke_i,i}(f) \cdot T^{k_i+ke_i} \;=\; p^{\delta_i} T^{k_i} \cdot [1 + p^{e_i-1}T^{e_i} + (p^{e_i-1}T^{e_i})^2 + \ldots] = \; p^{\delta_i} \cdot T^{k_i} / \left(1 - p^{e_i-1}T^{e_i}\right).$$

$$\begin{aligned}
\text{So, } P_i(t) \;&=\; \frac{p^{\delta_i}T^{k_i}}{1-p^{e_i-1}T^{e_i}} + \frac{p^{\delta_i}T^{k_i+1}}{1-p^{e_i-1}T^{e_i}} + \frac{p^{\delta_i+1}T^{k_i+2}}{1-p^{e_i-1}T^{e_i}} + \ldots + \frac{p^{\delta_i+e_i-2}T^{k_i+e_i-1}}{1-p^{e_i-1}T^{e_i}} \\
&=\; \frac{p^{\delta_i}T^{k_i}}{1-p^{e_i-1}T^{e_i}} + \frac{p^{\delta_i}T^{k_i+1}}{1-p^{e_i-1}T^{e_i}} \cdot \left(1 + pT + (pT)^2 + \ldots + (pT)^{e_i-2}\right) \\
&=\; \frac{p^{\delta_i}T^{k_i}}{1-p^{e_i-1}T^{e_i}} \cdot \left(1 + T \cdot \frac{1-(pT)^{e_i-1}}{1-pT}\right) \;.
\end{aligned}$$

Putting $T = t/p$ and $\delta_i = k_i - (k_i - \nu_i)/e_i$ we get,

$$P_i(t) = \frac{t^{k_i}(p - t(p-1) - t^{e_i})}{p^{(k_i-\nu_i)/e_i}(1-t)(p-t^{e_i})}.$$

□

Now we are in a position to prove our main theorem.

Proof of Theorem 6.0.1. Recall $P(t) = P_0(t) + \sum_{i=1}^{n} P_i(t)$. We first compute $P_0(t)$ which is the sum of two polynomials in t, namely, $Q_1(t) := \sum_{j=0}^{k_0-1} N_j(f)(p^{-1}t)^j$ of degree $O(d\Delta)$, and $Q_2(t) = \sum_{i=1}^{n} \sum_{l=k_0}^{k_i-1} N_{l,i}(f)(p^{-1}t)^l$ also of degree $O(d\Delta)$. By a standard determinant/Sylvester matrix calculation one shows: $d\Delta \le O\left(d^2 \cdot (\log_p C + \log_p d)\right)$.

We can compute the polynomial $Q_1(t)$ in deterministic $\mathrm{poly}(d, \log C + \log p)$-time by calling the root-counting algorithm (Theorem 6.1.2) $k_0 - 1$ times, getting each $N_j(f)$, for $j = 1, \ldots, k_0 - 1$ (note: $N_0(f) := 1$).

Polynomial $Q_2(t)$ is a sum of $n \le d$ polynomials, each with $k_i - k_0 \le d$ many simple terms. Using Lemma 6.5.1, we can compute each ν_i, e_i, hence, $N_{l,i}(f)$. So, computation of Q_2 again takes time $\mathrm{poly}(d, \log C + \log p)$.

Lemma 6.5.2 gives us the rational form expression for $P_i(t)$, for each $i \in [n]$. So, using

Lemma 6.5.1 we can compute the Poincaré series

$$P(t) = P_0(t) + \sum_{i=1}^{n} \frac{t^{k_i}(p - t(p-1) - t^{e_i})}{p^{(k_i-\nu_i)/e_i}(1-t)(p-t^{e_i})}$$

in deterministic $\text{poly}(d, \log C + \log p)$ time.

By inspecting the above expression, the degree of denominator $B(t)$ is $1+\sum_{i=1}^{n} e_i = O(d)$. The degree of numerator $A(t)$ is $\leq k_0 + 2d \leq O\left(d^2 \cdot (\log_p C + \log_p d)\right)$. □

6.6 Summary

We presented the first complete solution to the problem of computing Igusa's local zeta function for any given integral univariate polynomial and a prime p. Indeed, our methods work for given $f \in \mathbb{Z}_p[x]$ (with f having computable representation) as our proof for integral f goes by considering its factorization over $\mathbb{Z}_p$.

We also found explicit closed-form expression for $N_k(f)$ and efficiently computed the explicit parameters involved therein, which could be used to compute Greenberg's constants associated with a univariate f and a prime p. Greenberg's constants appear in a classical theorem of Greenberg [Gre66, Theorem 1] which is a generalization of Hensel's lemma to several n-variate polynomials. We hope that our methods for one-variable case could be generalized to compute Greenberg's constants for n-variable case to give an effective version of Greenberg's theorem.

We also hope that our methods extend computing Igusa's local zeta function from characteristic zero ($\mathbb{Z}_p$) to positive characteristic ($\mathbb{F}_p[[T]]$) at least if some standard restrictions are imposed on the characteristic for e.g. p is 'large-enough'. This is supported by the fact that the root counting algorithm of [DMS19] (Chapter 4) also extends to $\mathbb{F}[[T]]$ for a field $\mathbb{F}$.

Part II

Random Sampling via Ideals

Chapter 7

Reduction of Factoring to Root Finding and Factoring modulo p^4

In this part of the thesis, we study two important problems over Galois rings of characteristic p^k: (1) Factoring an integral univariate polynomial $f(x)$ and, (2) Solving a system of multivariate polynomial equations (Search Hilbert's Nullstellensatz SHN).

For these problems, only inefficient brute force algorithms are known even when $k = 3$. We will establish a connection between these two seemingly different problems in the *constant* k regime. Firstly in the present chapter, we will reduce univariate factoring modulo prime powers to univariate root finding modulo a bi-generated ideal. This partial reduction helps us to make first progress in factoring $f \bmod p^k$ for $k = 3, 4$. Then in Chapter 8, we extend the reduction further and reduce *low*-degree univariate factoring to *low*-variate polynomial system solving when k is constant. Finally, we utilize the method of ideals which allowed efficient random sampling for the latter, thereby making significant progress on both the problems. Our first result is,

Theorem 7.0.1. *Let p be prime, $k \leq 4$ and $f \in \mathbb{Z}[x]$ be a univariate integral polynomial. Then, $f \bmod p^k$ can be factored (find a non-trivial factor or report irreducible) in randomized poly($\deg f, \log p$) time.*

The procedure to factor $f \bmod p^4$ also factors mod p^3 and mod p^2 (and tests for irre-

ducibility) in randomized poly$(\deg f, \log p)$ time. We also give a refinement of Hensel lifting to get all the lifts of a factor of f mod p to mod p^k when $k \leq 4$.

Theorem 7.0.2. *Let p be prime, $k \leq 4$ and $f \in \mathbb{Z}[x]$ be a univariate integral polynomial such that f mod p is a power of an irreducible polynomial. Let $\tilde{g}$ be a given factor of f mod p. Then, in randomized poly$(\deg f, \log p)$ time, we can compactly describe (and count) all possible factors of f mod p^k which are lifts of $\tilde{g}$ (or report that there are none).*

We give some assumptions here, on the given input polynomial $f \in \mathbb{Z}[x]$, which will be followed in further sections unless explicitly stated otherwise.

Preprocessing: Our task is to non-trivially factor a univariate integral polynomial $f \in \mathbb{Z}[x]$ of degree d modulo a prime power p^k. Without loss of generality, we can assume that $f \not\equiv 0 \bmod p$. Otherwise, we can efficiently divide f by the highest power of p possible, say p^ℓ, such that $f(x) \equiv p^\ell \tilde{f}(x) \bmod p^k$ and $\tilde{f}(x) \not\equiv 0 \bmod p$. In this case, it is equivalent to factor $\tilde{f}$ instead of f.

To simplify the input further, write f mod p (uniquely) as a product of powers of coprime irreducible polynomials (Theorem 2.2.1). If there are two coprime factors of f, using Hensel lemma (Lemma 2.2.2), we get a non-trivial factorization of f mod p^k. So we can assume that f is a power of a monic irreducible polynomial $\varphi \in \mathbb{Z}[x]$ modulo p. In other words, we can efficiently write

$$f \equiv \varphi^e + p\ell \bmod p^k$$

for a polynomial ℓ in $(\mathbb{Z}/\langle p^k \rangle)[x]$. We have $e \cdot \deg \varphi \leq \deg f$, for the integral polynomials f and φ.

Organization: Factoring a univariate modulo p goes through root finding in an extension field of $\mathbb{F}_p$. Our factoring method passes through a similar stage. In Section 7.2, we reduce factoring f mod p^k to root finding of $E \in (\mathbb{Z}[x])[y]$ modulo the bi-generated ideal $\langle p^k, \varphi^{ak} \rangle$ for some $a < e$. In Section 7.1, we give some useful tools to work with these bi-generated ideals.

Section 7.3 proves our main Theorems- 7.0.1 and 7.0.2. Section 7.3.1 shows how to find (and count) roots of E in simpler case of $k = 3$. In rest of Section 7.3 we generalize the idea used for $k = 3$ to find (and count) roots of E for $k = 4$. Finally, we conclude in Section 7.5.

7.1 Preliminaries

Let $R[x]$ be the ring of polynomials over $R = \mathbb{Z}/\langle p^k \rangle$. The following lemma about zero divisors in $R[x]$ will be helpful.

Lemma 7.1.1. *A polynomial $f \in R[x]$ is a zero divisor iff $f \equiv 0 \bmod p$. Consequently, for any polynomials $f, g_1, g_2 \in R[x]$ and $f \not\equiv 0 \bmod p$, $fg_1 = fg_2$ implies $g_1 = g_2$.*

Proof. If $f \equiv 0 \bmod p$ then $f \cdot p^{k-1}$ is zero, and f is a zero divisor.

For the other direction, let $f \not\equiv 0 \bmod p$ and assume $fg = 0$ for some non-zero $g \in R[x]$. Let

- i be the biggest integer such that the coefficient of x^i in f is non-zero modulo p,
- and j be the biggest integer such that the coefficient of x^j in g has minimum valuation with respect to p.

Then, the coefficient of x^{i+j} in $f \cdot g$ has same valuation as the coefficient of x^j in g, implying that the coefficient is nonzero. This contradicts the assumption $f \cdot g = 0$.

The consequence follows because $f \not\equiv 0 \bmod p$ implies that f cannot be a zero divisor. □

We can make the following observations about quotient ideals and bi-generated ideals.

Claim 7.1.2 (Cancellation)**.** *Suppose I is an ideal of ring R and a, b, c are three elements in R. By definition of quotient ideals, $ca \equiv cb \bmod I$ iff $a \equiv b \bmod I : \langle c \rangle$.*

Claim 7.1.3. *Let p be a prime and $\varphi \in (\mathbb{Z}/\langle p^k \rangle)[x]$ be such that $\varphi \not\equiv 0 \bmod p$. Given an ideal $I := \langle p^\ell, \varphi^m \rangle$ of $\mathbb{Z}[x]$,*

1. *$I : \langle p^i \rangle = \langle p^{\ell-i}, \varphi^m \rangle$, for $i \leq \ell$, and*
2. *$I : \langle \varphi^j \rangle = \langle p^\ell, \varphi^{m-j} \rangle$, for $j \leq m$.*

Proof. We will only prove part (1), as the proof of part (2) is similar. If $c \in \langle p^{\ell-i}, \varphi^m \rangle$ then there exists $c_1, c_2 \in \mathbb{Z}[x]$, such that, $c = c_1 p^{\ell-i} + c_2 \varphi^m$. Multiplying by p^i,

$$p^i c = c_1 p^\ell + c_2 p^i \varphi^m \in I \Rightarrow c \in I : \langle p^i \rangle.$$

To prove the reverse direction, if $c \in I : \langle p^i \rangle$ then there exists $c_1, c_2 \in \mathbb{Z}[x]$, such that, $p^i c = c_1 p^\ell + c_2 \varphi^m$. Since $i \leq \ell$ and $p \nmid \varphi$, we know $p^i | c_2$. So, $c = c_1 p^{\ell - i} + (c_2/p^i)\varphi^m \Rightarrow c \in \langle p^{\ell-i}, \varphi^m \rangle$. □

Lemma 7.1.4 (Compute quotient)**.** *Given a polynomial $\varphi \in \mathbb{Z}[x]$ not divisible by p, define I to be the ideal $\langle p^\ell, \varphi^m \rangle$ of $\mathbb{Z}[x]$. If $g(y) \in (\mathbb{Z}[x])[y]$ is a polynomial such that $g \equiv 0 \bmod \langle p, \varphi^m \rangle$, then $p|g \bmod I$ and $g/p \bmod I : \langle p \rangle$ is efficiently computable.*

Proof. The equation $g \equiv 0 \bmod \langle p, \varphi^m \rangle$ implies $g = pc_1 + \varphi^m c_2$ for some polynomials $c_1, c_2 \in \mathbb{Z}[x][y]$. Going modulo I, $g \equiv pc_1 \bmod I$. Hence, $p|g \bmod I$ and $g/p \equiv c_1 \bmod I : \langle p \rangle$ (Claim 7.1.2).

If we write g in the reduced form modulo I, then the polynomial g/p can be obtained by dividing each coefficient of $g \bmod I$ by p. □

7.2 Factoring to Root Finding

In this section we give a general framework to work on the problem of factoring $f \bmod p^k$– we reduce factoring $f \bmod p^k$ to root finding in a more general ring. This is inspired from characteristic p fields ($k = 1$) where efficient factoring is shown to be equivalent to finding roots efficiently. The reduction seems quite natural: It helps in factoring for $k \leq 4$ in this chapter and it extends further in Chapter 8 to give factoring results for constant k.

Following the preprocessing, it is enough to factor $f \in \mathbb{Z}[x]$ such that

$$f \equiv \varphi^e + p\ell \bmod p^k,$$

where $\varphi \in \mathbb{Z}[x]$ is an irreducible polynomial modulo p. Up to multiplication by units, any non-trivial factor h of f has the form $h \equiv \varphi^a - py$, as $h \bmod p$ is a factor of $f \equiv \varphi^e \bmod p$, where $a < e$ and y is a polynomial in $(\mathbb{Z}/\langle p^k \rangle)[x]$.

Let us denote the ring $\mathbb{Z}[x]/\langle p^k, \varphi^{ak} \rangle$ by R. Also, denote the ring $\mathbb{Z}[x]/\langle p, \varphi^{ak} \rangle$ by R_0. We define an auxiliary polynomial $E \in R[y]$ via

$$E := f \cdot (\varphi^{a(k-1)} + \varphi^{a(k-2)}(py) + \cdots + \varphi^a (py)^{k-2} + (py)^{k-1}).$$

Theorem 7.2.1 reduces the problem of factoring $f \bmod p^k$ to the problem of finding roots of the univariate polynomial E in ring R. Thus, we convert the problem of finding factors of $f \in \mathbb{Z}[x]$ modulo a *principal ideal* $\langle p^k \rangle$ to root finding of a polynomial $E \in (\mathbb{Z}[x])[y]$ modulo a *bi-generated ideal* $\langle p^k, \varphi^{ak} \rangle$.

Theorem 7.2.1 (Reduction theorem)**.** *Given a prime power p^k; let $f, h \in \mathbb{Z}[x]$ satisfy $f \equiv \varphi^e + p\ell \bmod p^k$ and $h \equiv \varphi^a - py \bmod p^k$, with $\ell, y \in (\mathbb{Z}/\langle p^k \rangle)[x]$ and $a \leq e$. Then, h divides f modulo p^k if and only if*

$$E \;=\; f \cdot (\varphi^{a(k-1)} + \varphi^{a(k-2)}(py) + \cdots + \varphi^{a}(py)^{k-2} + (py)^{k-1}) \;\equiv\; 0 \bmod \langle p^k, \varphi^{ak} \rangle.$$

Proof. Let Q denote the *ring of fractions* of the ring $(\mathbb{Z}/\langle p^k \rangle)[x]$. Since φ is not a zero divisor, $(E(y)/\varphi^{ak}) \in Q$.

We first prove the reverse direction. If $E \equiv 0 \bmod \langle p^k, \varphi^{ak} \rangle$, then (E/φ^{ak}) is a polynomial over $(\mathbb{Z}/\langle p^k \rangle)[x]$. Multiplying h with $(E/\varphi^{ak}) \bmod p^k$, we write,

$$(\varphi^a - py)((f/\varphi^{ak})\Sigma_{i=0}^{k-1}\varphi^{a(k-1-i)}(py)^i) \;\equiv\; (f/\varphi^{ak})(\varphi^{ak} - (py)^k) \equiv f \cdot \varphi^{ak}/\varphi^{ak} \equiv f \bmod p^k.$$

The first equality comes via geometric series. Hence, h divides f modulo p^k.

For the forward direction, assume that there exists some $g \in (\mathbb{Z}/\langle p^k \rangle)[x]$, such that, $f(x) \equiv h(x)g(x) \bmod p^k$. We get two factorizations of f in Q,

$$f = h \cdot g \quad \text{and} \quad f = h \cdot (E/\varphi^{ak}).$$

Subtracting the first equation from the second one,

$$h \cdot \Big(g - (E/\varphi^{ak})\Big) = 0.$$

Notice that h is not a zero divisor in $(\mathbb{Z}/\langle p^k \rangle)[x]$ (by Lemma 7.1.1) and is thus invertible in Q. So, $E/\varphi^{ak} = g$ in Q. Since g is in $(\mathbb{Z}/\langle p^k \rangle)[x]$, we deduce the equivalent divisibility statement: $E(y) \equiv 0 \bmod \langle p^k, \varphi^{ak} \rangle$. □

Following the reduction in this section, we move on to find roots of $E(y)$, when $k \leq 4$, in the next section (Section 7.3).

7.3 Factoring and Lifting modulo Prime Power p^4

In this section we will prove Theorems 7.0.1 and 7.0.2. We want to find (and count) all the factors $h \in (\mathbb{Z}/\langle p^k \rangle)[x]$ of the given degree d polynomial $f \in \mathbb{Z}[x]$ modulo p^k for $k \leq 4$, where $f \equiv \varphi^e + p\ell \bmod p^k$ and $h = \varphi^a - py$.

We also recall the definitions from Section 7.2. We have $R := \mathbb{Z}[x]/\langle p^k, \varphi^{ak} \rangle$ and $R_0 = \mathbb{Z}[x]/\langle p, \varphi^{ak} \rangle$. For a factor h of f, define $E \in R[y]$ as

$$E := f \cdot (\varphi^{a(k-1)} + \varphi^{a(k-2)}(py) + \cdots + \varphi^a (py)^{k-2} + (py)^{k-1}).$$

The following two observations simplify our task of finding roots y of polynomial $E(y)$.

(1) First, due to symmetry, it is enough to find factors $h \equiv \varphi^a \bmod p$ with $a \leq e/2$. The assertion follows because $f \equiv hg \bmod p^k$ implies, at least one of the factor (say h) must be of the form $\varphi^a \bmod p$ for $a \leq e/2$. By Lemma 7.1.1, for a fixed $h \equiv \varphi^a - py \bmod p^k$, there is a unique $g \equiv \varphi^{e-a} - py' \bmod p^k$ such that $f \equiv hg \bmod p^k$. So, to find g, it is enough to find h.

(2) Second, observe that any root $y \in R$ (of $E \in R[y]$) can be seen as $y = y_0 + py_1 + p^2 y_2 + \cdots + p^{k-1} y_{k-1}$, where each $y_i \in R_0$ for all i in $\{0, \ldots, k-1\}$. The following lemma decreases the required precision of a root y.

Lemma 7.3.1. *Let $y = y_0 + py_1 + p^2 y_2 + \cdots + p^{k-1} y_{k-1}$ be a root of E, where $k \geq 2$ and $a \leq e/2$. Then, all elements of the set $y = y_0 + py_1 + p^2 y_2 + \cdots + p^{k-3} y_{k-3} + p^{k-2} *$ are also roots of E.*

Proof. Notice that the variable y is multiplied with p in $E(y)$, implying y_{k-1} is irrelevant. A similar argument is applicable for the coefficient y_{k-2} in any term involving $(py)^i$ for $i \geq 2$. The only surviving term containing y_{k-2} is $f\varphi^{a(k-2)}(py)$. The coefficient of y_{k-2} in this term is $\varphi^{a(k-2)} f p^{k-1}$, it also vanishes because

$$\varphi^{a(k-2)} f \equiv \varphi^{a(k-2)} \varphi^e \equiv \varphi^{ak} \varphi^{e-2a} \equiv 0 \bmod \langle p, \varphi^{ak} \rangle .$$ □

Root-finding modulo a Principal Ideal. In next few sections we will see that finding roots of E in R goes through finding roots of intermediate polynomials in $R_0 = \mathbb{F}_p[x]/\langle \varphi^{ak} \rangle$ (i.e, modulo a principal ideal). In the following, we give slightly modified version of Theorem 2.3.1 which says that all the roots of a polynomial $g \in R_0[y]$ can be efficiently described

in form of analogously defined representative roots. Recall Section 2.3 in Chapter 2 for the notion of representatives, representative roots and representative pairs. A representative in R_0 looks like $y := y_0 + \varphi y_1 + \cdots + \varphi^i y_i + \varphi^{i+1}*$, with y_is seen as elements of $R_0/\langle\varphi\rangle$, and it is a representative root if not all elements of $y' := y_0 + \varphi y_1 + \cdots + \varphi^{i-1} y_{i-1} + \varphi^i *$ are the roots. The representative pair for y is written as $(v, i+1)$ where $v := y_0 + \varphi y_1 + \cdots + \varphi^i y_i$ in R_0.

Theorem 7.3.2 (Modified Theorem 2.3.1)**.** *Given a bivariate $g \in R_0[y]$ where $R_0 = \mathbb{Z}[x]/\langle p, \varphi^i\rangle$, let $Z \subseteq R_0$ be the root set of $g(y)$. Then Z can be expressed as the disjoint union of at most $\deg_y(g)$ many representative pairs (a_0, i_0) ($a_0 \in R_0$ and $i_0 \in \mathbb{N}$).*

These representative pairs can be found in randomized poly($\deg_y(g), \log p, ak \deg \varphi$) time.

7.3.1 Finding All the Factors modulo p^k for $k < 4$

In this section we partially prove Theorems 7.0.1 and 7.0.2, i.e., we efficiently find all the factors of $f \bmod p^2$ and $f \bmod p^3$. Although the case of $k = 2$ is already solved [Săl05], the case of $k = 3$ was left open in [Sir17]. The ideas in this section (for $k \leq 3$) will be generalized to solve the case $k = 4$.

Factoring f mod p^2. The reduction theorem (Theorem 7.2.1) and Lemma 7.3.1 make factoring mod p^2 easy: They imply that any root of E is independent of coordinates y_0 and y_1. So, either $h = \varphi^a - py$ can not be a factor of $f \bmod p^2$ or it is a factor for every value of $y \in R_0$. Substituting $y = 0$, we get that $h \equiv \varphi^a - py \bmod p^2$ is a factor of f if and only if $\varphi^a | f$ modulo p^2. In fact, we get a simple irreducibility criteria— *f mod p^2 factors if and only if $\varphi | f$ mod p^2* (first discovered by [Săl05]).

Factoring f mod p^3. Theorem 7.3.3 below solves the factoring problem modulo p^3.

Theorem 7.3.3. *Given $f \in \mathbb{Z}[x]$, a univariate polynomial of degree d and a prime $p \in \mathbb{N}$, we give (and count) all the distinct factors of $f \bmod p^3$ of degree at most d in randomized poly($d, \log p$) time.*

Note: We will assume that the leading coefficient of f is 1. Also, we will not distinguish two factors if they are same up to multiplication by a unit.

Proof of Theorem 7.3.3. By Theorem 2.2.1, a general f can be written as:

$$f(x) \equiv \prod_{i=1}^{n} f_i(x) \equiv \prod_{i=1}^{n} (\varphi_i^{e_i} + ph_i) \mod p^3, \tag{7.1}$$

where $f_i(x) \equiv (\varphi_i^{e_i} + ph_i) \bmod p^3$ with $\varphi_i \bmod p^3$ being monic and irreducible mod p, $e_i \in \mathbb{N}$, and $h_i(x) \bmod p^3$ of degree $< e_i \deg(\varphi_i)$, for all $i \in [n]$.

Using Lemma 2.2.2, it is sufficient to consider the case $f \equiv \varphi^e + ph$.

By Reduction theorem (Theorem 7.2.1) finding factors of the form $\varphi^a - py \bmod p^3$ of $f \equiv \varphi^e + ph \bmod p^3$, for $a \leq e/2$, is equivalent to finding all roots of the equation

$$E \equiv f \cdot (\varphi^{2a} + \varphi^a(py) + (py)^2) \equiv 0 \mod \langle p^3, \varphi^{3a} \rangle.$$

Consider $R := \mathbb{Z}[x]/\langle p^3, \varphi^{3a} \rangle$ and $R_0 := \mathbb{Z}[x]/\langle p, \varphi^{3a} \rangle$ (analogous to Section 3.2).

Using Lemma 7.3.1, we know that all solutions of the equation $E \equiv 0 \bmod \langle p^3, \varphi^{3a} \rangle$ will be of the form $y = y_0 + p* \in R$, for a $y_0 \in R_0$. Substituting, we get

$$E \equiv ph\varphi^{2a} + (p^2 h\varphi^a)y_0 + (p^2\varphi^e)y_0^2 \equiv 0 \mod \langle p^3, \varphi^{3a} \rangle.$$

Looking at this equation mod $\langle p^2, \varphi^{3a} \rangle$, we get that $h \equiv 0 \bmod \langle p, \varphi^a \rangle$ is a necessary condition for a root y_0 to exist. Define $h := \varphi^a g_1 + pg_2$ for unique $g_1, g_2 \in \mathbb{F}_p[x]$, the equation becomes

$$E \equiv p^2 g_2 \varphi^{2a} + (p^2 g_1 \varphi^{2a})y_0 + (p^2\varphi^e)y_0^2 \equiv 0 \mod \langle p^3, \varphi^{3a} \rangle,$$

This equation is already divisible by p^2 as well as φ^{2a}. Using Claim 7.1.3, finding factors of the form $\varphi^a - py \bmod p^3$ is equivalent to finding all roots of the equation

$$g_2 + g_1 y_0 + \varphi^{e-2a} y_0^2 \equiv 0 \mod \langle p, \varphi^a \rangle .$$

These roots can be obtained using one call to ROOT-FIND in randomized $\text{poly}(d, \log p)$ time. Note that any root y_0 given by ROOT-FIND is an element of $\mathbb{F}_p[x]/\langle \varphi^a \rangle$, implying its degree in x is $< a \deg(\varphi)$. This yields *monic* factors of $f \bmod p^3$ (with $0 \leq a \leq e/2$).

For $e \geq a > e/2$, we can replace a by $b := e - a$ in the above steps. Once we get a factor $\varphi^b - py \bmod p^3$, we output the cofactor $f/(\varphi^b - py) = (f/\varphi^{ak})(\varphi^{2a} + \varphi^a(py) + (py)^2)$ (which remains monic).

Since Theorem 7.3.2 gives the numbers of roots from ROOT-FIND, we also get a count on total number of factors in poly-time.

For a general f (Equation 7.1), if N_i is the number of factors of $f_i \bmod p^3$, then $\prod_{i=1}^{n} N_i$ is the count on the number of distinct monic factors of $f \bmod p^3$. □

Let us illustrate the steps in the proof of Theorem 7.3.3 by an example.

Example 13. *Let* $f = x^4 + 18x^3 + 33x^2 + 54x + 9$ *be an integral polynomial and pick* $p = 3, k = 3$.

We need to find factors of $f \equiv x^4 + 18x^3 + 6x^2 + 9 \bmod 27$*; since* $f \equiv x^4 \bmod 3$*, fix* $\varphi := x \in \mathbb{Z}[x]$*. Say, we want to find quadratic factors of* $f \bmod 27$*, so fix* $a = 2$.

Recall the reduction theorem (Theorem 7.2.1), $(\varphi^a - py)$ *is a factor of* $f \bmod p^3$ *iff*

$$E := f \cdot (\varphi^{2a} + \varphi^a(py) + (py)^2) \equiv 0 \bmod \langle p^3, \varphi^{3a} \rangle.$$

$\Leftrightarrow (x^4 + 18x^3 + 6x^2 + 9)[x^4 + x^2(3y_0) + 9y_0^2] \equiv 0 \bmod \langle 27, x^6 \rangle$. *($y = y_0$ [Lemma 7.3.1])*

$\Leftrightarrow 9x^4y_0^2 + 18x^4y_0 + 9x^4 \equiv 0 \bmod \langle 27, x^6 \rangle$.

$\Leftrightarrow y_0^2 + 2y_0 + 1 \equiv 0 \bmod \langle 3, x^2 \rangle$.

$\Leftrightarrow (y_0 + 1)^2 \equiv 0 \bmod \langle 3, x^2 \rangle$.

Applying [Pan95, BLQ13] (Theorem 7.3.2) on last equation, we get exactly one representative root $y_0 = 2 + x*$.

Choosing $y_1 = 0$ *and* $y_0 = 2 + 0$*, we have a corresponding factor* $(x^2 - 3(2 + 0)) \equiv (x^2 + 21) \bmod 27$*. The co-factor of this is* $(x^2 + 18x + 12)$*, giving*

$$f \equiv x^4 + 18x^3 + 6x^2 + 9 \equiv (x^2 + 21)(x^2 + 18x + 12) \bmod 27.$$

Remark. Observe that the core idea for p^3 was to first reduce root finding of $E \bmod \langle p^3, \varphi^{3a} \rangle$ to root finding modulo a principal ideal $\langle p, \varphi^a \rangle$. It was then solved by just one application of Theorem 7.3.2. For p^3, we only needed to deal with a univariate polynomial in y_0.

The approach for $k = 4$ is similar, though it requires several applications of Theorem 7.3.2 to go to principal ideal $\langle p, \varphi^{4a} \rangle$ (Sec. 7.3.2). Even after that, we are required to solve a bivariate equation modulo the principal ideal (as opposed to a univariate in the case $k = 3$).

We will fix $k = 4$ for the rest of Section 7.3.

7.3.2 Reduction to Root Finding modulo a Principal Ideal of $\mathbb{F}_p[x]$

In this subsection, the task to find roots of E modulo the bi-generated ideal $\langle p^4, \varphi^{4a}\rangle$ of $\mathbb{Z}[x]$ will be reduced to finding roots modulo the principal ideal $\langle \varphi^{4a}\rangle$ (of $\mathbb{F}_p[x]$).

Let us consider the equation $E \equiv 0 \bmod \langle p^4, \varphi^{4a}\rangle$. We have,

$$f(\varphi^{3a} + \varphi^{2a}(py) + \varphi^a(py)^2 + (py)^3) \equiv 0 \bmod \langle p^4, \varphi^{4a}\rangle\,. \tag{7.2}$$

Using Lemma 7.3.1, we can assume $y = y_0 + py_1$,

$$f(\varphi^{3a} + \varphi^{2a}p(y_0 + py_1) + \varphi^a p^2(y_0^2 + 2py_0y_1) + (py_0)^3) \equiv 0 \bmod \langle p^4, \varphi^{4a}\rangle\,. \tag{7.3}$$

The idea is to first solve this equation modulo $\langle p^3, \varphi^{4a}\rangle$. Since $f \equiv \varphi^e \bmod p$, $e \geq 2a$, variable y_1 is redundant while solving this equation modulo p^3. The following lemma finds all representative pairs (a_0, i_0) for y_0, such that, $E(a_0 + \varphi^{i_0}y_0 + py_1) \equiv 0 \bmod \langle p^3, \varphi^{4a}\rangle$ for all $y_0, y_1 \in R$. Alternatively, we can state this in the polynomial ring $R[y_0, y_1]$. Dividing by p^3, we will be left with an equation modulo the principal ideal $\langle \varphi^{4a}\rangle$ (of $\mathbb{F}_p[x]$).

Lemma 7.3.4 (Reduction to characteristic p)**.** *We efficiently compute a unique 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,*

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

for a polynomial $E'(y_0, y_1) \in R_0[y_0, y_1]$ (depending on (a_0, i_0)). Moreover,

1. *$|S_0| \leq 2$ and, If our algorithm fails to find E', then Eqn. 7.3 has no solution.*
2. *$E'(y_0, y_1) =: E_1(y_0) + E_2(y_0)y_1$, where $E_1 \in R_0[y_0]$ is cubic in y_0 and $E_2 \in R_0[y_0]$ is linear in y_0.*
3. *For every root $y \in R$ of E there exists $(a_0, i_0) \in S_0$ and $(a_1, a_2) \in R \times R$, such that $y = (a_0 + \varphi^{i_0}a_1) + pa_2$ and $E'(a_1, a_2) \equiv 0 \bmod \langle p, \varphi^{4a}\rangle$.*

We think of E' as the quotient $E((a_0 + \varphi^{i_0}y_0) + py_1)/p^3$ in the polynomial ring $R_0[y_0, y_1]$; and would work with it instead of E in the root-finding algorithm.

Proof. Looking at Eqn. 7.3 modulo p^2,

$$f\varphi^{2a}(\varphi^a + py_0) \equiv 0 \mod \langle p^2, \varphi^{4a}\rangle.$$

Substituting $f = \varphi^e + ph_1$, we get $(\varphi^e + ph_1)(\varphi^{3a} + \varphi^{2a}py_0) \equiv 0 \bmod \langle p^2, \varphi^{4a}\rangle$. Implying, $ph_1\varphi^{3a} \equiv 0 \bmod \langle p^2, \varphi^{4a}\rangle$. Using Claim 7.1.3 the above equation implies that,

$$h_1 \equiv 0 \mod \langle p, \varphi^a\rangle, \tag{7.4}$$

is a necessary condition for y_0 to exist.

We again look at Eqn. 7.3, but modulo p^3 now: $f(\varphi^{3a} + \varphi^{2a}py_0 + \varphi^a p^2 y_0^2) \equiv 0 \mod \langle p^3, \varphi^{4a}\rangle$.

Notice that y_1 is not present because of its coefficient: $p^2 f\varphi^{2a} \equiv 0 \bmod \langle p^3, \varphi^{4a}\rangle$. Substituting $f = \varphi^e + ph_1$, we get,

$$(\varphi^e + ph_1)(\varphi^{3a} + \varphi^{2a}py_0 + \varphi^a p^2 y_o^2) \equiv 0 \bmod \langle p^3, \varphi^{4a}\rangle.$$

Removing the coefficients of y_0 which vanish modulo $\langle p^3, \varphi^{4a}\rangle$,

$$\varphi^{e+a}p^2y_0^2 + \varphi^{3a}ph_1 + \varphi^{2a}p^2h_1y_0 \equiv 0 \mod \langle p^3, \varphi^{4a}\rangle.$$

From Eqn. 7.4, h_1 can be written as $ph_{1,1} + \varphi^a h_{1,2}$, so

$$p^2\left(\varphi^{e+a}y_0^2 + \varphi^{3a}h_{1,2}y_0 + \varphi^{3a}h_{1,1}\right) \equiv 0 \mod \langle p^3, \varphi^{4a}\rangle.$$

We can divide by $p^2\varphi^{3a}$ using Claim 7.1.3 to get an equation modulo φ^a in the ring $\mathbb{F}_p[x]$. This is a quadratic equation in y_0. Using Theorem 7.3.2, we find the solution set S_0 with at most two representative pairs: for $(a_0, i_0) \in S_0$, every $y \in a_0 + \varphi^{i_0} * + p*$ satisfies,

$$E \equiv 0 \mod \langle p^3, \varphi^{4a}\rangle.$$

In other words, upon substituting $y = a_0 + \varphi^{i_0}y_0 + py_1$ in $E(y)$, we get

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

for a "bivariate" polynomial $E'(y_0, y_1) \in R_0[y_0, y_1]$. This sets up the correspondence between

the roots of E and E'.

Substituting $(a_0+\varphi^{i_0}y_0+py_1)$ in Eqn. 7.3, we notice that $E'(y_0,y_1)$ has the form $E_1(y_0)+E_2(y_0)y_1$ for a linear E_2 and a cubic E_1.

Finally, this reduction is constructive, because of Lemma 7.1.4 and Theorem 7.3.2, giving a randomized poly-time algorithm. □

7.3.3 Finding Roots of a *Special* Bivariate $E'(y_0,y_1)$ modulo $\langle p,\varphi^{4a}\rangle$

The final obstacle is to find roots of $E'(y_0,y_1)$ modulo $\langle\varphi^{4a}\rangle$ in $\mathbb{F}_p[x]$. The polynomial $E'(y_0,y_1)=E_1(y_0)+E_2(y_0)y_1$ is *special* because $E_2\in R_0[y_0]$ is *linear* in y_0.

For a polynomial $u\in\mathbb{F}_p[x][\mathbf{y}]$ we define *valuation* $\mathrm{val}_\varphi(u)$ to be the largest r such that $\varphi^r|u$. Our strategy is to go over all possible valuations $0\le r\le 4a$ and find y_0, such that,

- $E_1(y_0)$ has valuation at least r.
- $E_2(y_0)$ has valuation exactly r.

From these y_0's, y_1 can be obtained by 'dividing' $E_1(y_0)$ by $E_2(y_0)$. The lemma below shows that this strategy captures all the solutions.

Lemma 7.3.5 (Bivariate solution)**.** *A pair $(u_0,u_1)\in R_0\times R_0$ satisfies an equation of the form $E_1(y_0)+E_2(y_0)y_1\equiv 0 \bmod \langle p,\varphi^{4a}\rangle$ if and only if $val_\varphi(E_1(u_0))\ge val_\varphi(E_2(u_0))$.*

Proof. Let r be $\mathrm{val}_\varphi(E_2(u_0))$, where r is in the set $\{0,1,\ldots,4a\}$. If $\mathrm{val}_\varphi(E_1(u_0))\ge\mathrm{val}_\varphi(E_2(u_0))$ then set $u_1\equiv -(E_1(u_0)/\varphi^r)/(E_2(u_0)/\varphi^r) \bmod \langle p,\varphi^{4a-r}\rangle$. The pair (u_0,u_1) satisfies the required equation. (Note: If $r=4a$ then we take $u_1=*$.)

Conversely, if $r':=\mathrm{val}_\varphi(E_1(u_0))<\mathrm{val}_\varphi(E_2(u_0))\le 4a$ then, for every u_1,

$\mathrm{val}_\varphi(E_1(u_0)+E_2(u_0)u_1)=r' \;\Rightarrow\; E_1(u_0)+E_2(u_0)u_1\not\equiv 0 \bmod \langle p,\varphi^{4a}\rangle$. □

We can efficiently find all representative pairs for y_0, at most three, such that $E_1(y_0)$ has valuation at least r (using Theorem 7.3.2). The next lemma shows that we can efficiently filter all y_0's, from these representative pairs, that give valuation *exactly* r for $E_2(y_0)$.

Lemma 7.3.6 (Reduce to a unit E_2)**.** *Given a linear polynomial $E_2(y_0) \in R_0[y_0]$ and an $r \in [4a-1]$, let (b,i) be a representative pair modulo $\langle p, \varphi^r \rangle$, i.e., $E_2(b+\varphi^i *) \equiv 0 \bmod \langle p, \varphi^r \rangle$. Consider the quotient $E_2'(y_0) := E_2(b+\varphi^i y_0)/\varphi^r$.*

If $E_2'(y_0)$ does not vanish identically modulo $\langle p, \varphi \rangle$, then there exists at most one $\theta \in R_0/\langle \varphi \rangle$ such that $E_2'(\theta) \equiv 0 \bmod \langle p, \varphi \rangle$, and this θ can be efficiently computed.

Proof. Suppose $E_2(b+\varphi^i y_0) \equiv u + v y_0 \equiv 0 \bmod \langle p, \varphi^r \rangle$. Since y_0 is formal, we get $\mathrm{val}_\varphi(u) \geq r$ and $\mathrm{val}_\varphi(v) \geq r$. We consider the three cases (with respect to these valuations),

1. $\mathrm{val}_\varphi(u) \geq r$ and $\mathrm{val}_\varphi(v) = r$: $E_2'(\theta) \not\equiv 0 \bmod \langle p, \varphi \rangle$, for all $\theta \in R_0/\langle \varphi \rangle$ except $\theta = (-u/\varphi^r)/(v/\varphi^r) \bmod \langle p, \varphi \rangle$.

2. $\mathrm{val}_\varphi(u) = r$ and $\mathrm{val}_\varphi(v) > r$: $E_2'(\theta) \not\equiv 0 \bmod \langle p, \varphi \rangle$, for all $\theta \in R_0/\langle \varphi \rangle$.

3. $\mathrm{val}_\varphi(u) > r$ and $\mathrm{val}_\varphi(v) > r$: $E_2'(y_0)$ vanishes identically modulo $\langle p, \varphi \rangle$, so this case is ruled out by the hypothesis.

There is an efficient algorithm to find θ, if it exists; because the above proof only requires calculating valuations which entails division operations in the ring. □

Before the algorithm let us illustrate the process on Example 13.

Example 14. *Consider the polynomial f from Example 13. We want to find factors of $f \equiv x^4 + 18x^3 + 33x^2 + 54x + 9 \bmod 81$. Fix $\varphi := x \in \mathbb{Z}[x]$ and $a = 2$.*

Let us apply the reduction theorem (Thm. 7.2.1): $(\varphi^a - py)$ is a factor of $f \bmod p^4$ iff

$$E := f \cdot (\varphi^{3a} + \varphi^{2a}(py) + \varphi^a (py)^2 + (py)^3) \equiv 0 \bmod \langle p^4, \varphi^{4a} \rangle.$$

Putting the values $\varphi = x, p = 3, a = 2$ and substituting $y = y_0 + 3y_1$ [Lemma 7.3.1] we have,

$(x^4+18x^3+33x^2+54x+9)[x^6+x^4 3(y_0+3y_1)+x^2 9(y_0+3y_1)^2+27(y_0+3y_1)^3] \equiv 0 \bmod \langle 81, x^8 \rangle$.

$\Leftrightarrow 9x^4[(6x^2 y_0 + 6x^2)y_1 + (3y_0^3 + y_0^2(x^2+6) + y_0(6x^3+2x^2+3) + 6x^3 + x^2)] \equiv 0 \bmod \langle 81, x^8 \rangle$.

Using Claim 7.1.3 to divide both sides by $9x^4$ we get the equation,

$$(6x^2 y_0 + 6x^2)y_1 + 3y_0^3 + y_0^2(x^2+6) + y_0(6x^3+2x^2+3) + 6x^3 + x^2 \equiv 0 \bmod \langle 9, x^4 \rangle. \quad (7.5)$$

Reducing the last equation mod $\langle 3, x^4\rangle$ *we have,*

$x^2y_0^2 + (2x^2)y_0 + x^2 \equiv 0 \bmod \langle 3, x^4\rangle$.

$\Leftrightarrow y_0^2 + 2y_0 + 1 \equiv 0 \bmod \langle 3, x^2\rangle$.

Notice that this is the same equation as for the case of $k = 3$ *in Example 13.*

Applying [Pan95, BLQ13] (Thm. 7.3.2) on last equation, we get exactly one representative root $y_0 = 2 + x*$.

We substitute $y_0 \to 2 + xy_0$ *in Equation 7.5 and simplify to get,*

$$(2xy_0)y_1 + xy_0^3 + 2y_0^2 + (2x)y_0 \equiv 0 \bmod \langle 3, x^2\rangle. \tag{7.6}$$

Equation 7.6 gives us $E_1(y_0) = xy_0^3 + 2y_0^2 + (2x)y_0 \bmod \langle 3, x^2\rangle$ *and* $E_2(y_0) = 2xy_0 \bmod \langle 3, x^2\rangle$.

We want the values of y_0*'s, such that,* $val_x(E_1(y_0))$ *is at least* $val_x(E_2(y_0))$. *Since* $val_x(E_2(y_0))$ *is* 1, *we are forced to have* $y_0 = 0 \bmod \langle 3, x\rangle$. *In that case, Equation 7.6 is identically zero, so* y_1 *is free to take any value mod* $\langle 3, x^2\rangle$.

Taking $y_1 = 0$ *and* $y_0 = 2 + 0$ *we have the corresponding factor* $(x^2 - 3(2 + 0)) \equiv (x^2 + 75) \bmod 81$. *The co-factor of this is* $(x^2 + 18x + 39)$, *giving*

$$f \equiv x^4 + 18x^3 + 33x^2 + 54x + 9 \equiv (x^2 + 75)(x^2 + 18x + 39) \bmod 81.$$

7.3.4 Algorithm to Find Roots of $E(y)$

We have all the ingredients to give the algorithm for finding roots of $E(y)$ modulo ideal $\langle p^4, \varphi^{4a}\rangle$ of $\mathbb{Z}[x]$.

Input: A polynomial $E \in R[y]$ defined as $E := f \cdot (\varphi^{3a} + \varphi^{2a}(py) + \varphi^a(py)^2 + (py)^3)$.

Output: A set $Z \subseteq R_0$ and a *bad* set $Z' \subseteq R_0$, such that, for each $y_0 \in Z - Z'$, there are (efficiently computable) $y_1 \in R_0$ (Theorem 7.3.7) satisfying $E(y_0 + py_1) \equiv 0 \bmod \langle p^4, \varphi^{4a}\rangle$. These are exactly the roots of E.

As we will show in Theorem 7.3.7 both the sets Z and Z' can be described by $O(a)$ many representatives. The representation is efficient as $a \leq d$. Hence, a $y_0 \in Z - Z'$ can be picked efficiently. We provide the correctness of Algorithm 6 in Theorem 7.3.7.

Algorithm 6 Finding all roots of $E(y)$ in R

1: Given $E(y_0 + py_1)$, using Lemma 7.3.4, 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 $p^3 | E((a_0 + \varphi^{i_0} y_0) + py_1) \bmod \langle p^4, \varphi^{4a} \rangle$.

2: Initialize sets $Z = \{\}$ and $Z' = \{\}$; seen as subsets of R_0.

3: **for** each $(a_0, i_0) \in S_0$ **do**

4: 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 7.3.4.

5: **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 $Z' \leftarrow Z' \cup (a_0 + \varphi^{i_0}(\theta + \varphi *))$.

6: **for** each possible valuation $r \in [4a]$ **do**

7: Initialize sets $Z_r = \{\}$ and $Z_r' = \{\}$.

8: 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 $E_1(a_1 + \varphi^{i_1} y_0) \equiv 0 \bmod \langle p, \varphi^r \rangle$.

9: **for** each $(a_1, i_1) \in S_1$ **do**

10: Analogously consider $E_2'(y_0) := E_2(a_1 + \varphi^{i_1} y_0) \bmod \langle p, \varphi^{4a} \rangle$.

11: 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 $E_2'(a_2 + \varphi^{i_2} y_0) \equiv 0 \bmod \langle p, \varphi^r \rangle$.

12: **if** $r = 4a$ **then**

13: Update $Z_r \leftarrow Z_r \cup (a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2} *))$ and $Z_r' \leftarrow Z_r' \cup \{\}$.

14: **else if** $E_2'(a_2 + \varphi^{i_2} y_0) \not\equiv 0 \bmod \langle p, \varphi^{r+1} \rangle$ **then**

15: Get a $\theta \in R_0/\langle \varphi \rangle$ (Lemma 7.3.6), 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 $Z_r' \leftarrow Z_r' \cup (a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2}(\theta + \varphi *)))$.

16: Update $Z_r \leftarrow Z_r \cup (a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2} *))$.

17: Update $Z \leftarrow Z \cup (a_0 + \varphi^{i_0} Z_r)$ and $Z' \leftarrow Z' \cup (a_0 + \varphi^{i_0} Z_r')$.

18: Return Z and Z'.

Theorem 7.3.7. *The output of Algorithm 6 (the set $Z - Z'$) 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 in R. We can compute the set of y_1 corresponding to a given $y_0 \in Z - Z'$ 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*, where* $y_0, y_1 \in R_0$ *are as above and* y_2 *can assume any value from* R*.*

Proof. The algorithm intends to output roots y of equation $E \equiv f \cdot (\varphi^{3a} + \varphi^{2a}(py) + \varphi^a(py)^2 + (py)^3) \equiv 0 \bmod \langle p^4, \varphi^{4a} \rangle$, where $y = y_0 + py_1 + p^2y_2$ with $y_0, y_1 \in R_0$ and $y_2 \in R$. From Lemma 7.3.1, any value of y_2 in $\mathbb{F}_p$ makes y a root, and we encode this by substituting the symbol $*$ for y_2.

Using Lemma 7.3.4, Algorithm 6 partially fixes y_0 from the set S_0 and reduces the problem to finding roots of an $E'(y_0, y_1) \bmod \langle p, \varphi^{4a} \rangle$. In other words, if we can find all roots (y_0, y_1) of $E'(y_0, y_1) \bmod \langle p, \varphi^{4a} \rangle$, then we can find (and count) all roots of $E(y) \bmod \langle p^4, \varphi^{4a} \rangle$. This is accomplished by Step 1. From Lemma 7.3.4, $|S_0| \leq 2$, so loop at Step 3 runs only for a constant number of times.

Using Lemma 7.3.4, $E'(y_0, y_1) \equiv E_1(y_0) + E_2(y_0)y_1 \bmod \langle p, \varphi^{4a} \rangle$ for a cubic polynomial $E_1 \in R_0[y_0]$ and a linear polynomial $E_2 \in R_0[y_0]$.

We find all solutions of $E'(y_0, y_1)$ by going over all possible valuations of $E_2(y_0)$ with respect to φ. The case of valuation 0 is handled in Step 5 and valuation $4a$ is handled in Step 12. For the remaining valuations $r \in [4a - 1]$, Lemma 7.3.5 shows that it is enough to find $(z_0, z_1) \in R_0 \times R_0$ such that $\varphi^r | E_1(z_0)$ and $\varphi^r || E_2(z_0)$.

Notice that the number of valuations is bounded by $4a = O(\deg f)$. At Step 6, the algorithm runs through the possible values of the valuation r of $E_2(y_0) \in R_0[y_0]$ and subsequent computation finds all representative roots $b + \varphi^i *$ efficiently (using Theorem 7.3.2), such that,

$$E_1(b + \varphi^i y_0) \equiv E_2(b + \varphi^i y_0) \equiv 0 \bmod \langle p, \varphi^r \rangle .$$

The representative root $b + \varphi^i *$ is denoted by $a_1 + \varphi^{i_1}(a_2 + \varphi^{i_2} *)$ in Steps 13 and 16 of Algorithm 6.

Finally, we need to filter out those y_0's for which $E_2(b + \varphi^i y_0) \equiv 0 \bmod \langle p, \varphi^{r+1} \rangle$. This can be done efficiently using Lemma 7.3.6, where we get a unique $\theta \in R_0/\langle \varphi \rangle$ for which,

$$E_2(b + \varphi^i(\theta + \varphi y_0)) \equiv 0 \bmod \langle p, \varphi^{r+1} \rangle.$$

We store partial roots in two sets Z_r and Z'_r, where Z'_r contains the bad values filtered

out by Lemma 7.3.6 as $b + \varphi^i(\theta + \varphi*)$ and Z_r contains all possible roots $b + \varphi^i*$. So, the set $Z_r - Z'_r$ contains exactly those elements z_0 for which there exists $z_1 \in R_0$, such that, the pair (z_0, z_1) is a root of $E'(y_0, y_1) \bmod \langle p, \varphi^{4a}\rangle$.

Note that size of each set S_1 obtained at Step 9 is bounded by three using Theorem 7.3.2 (E_1 is at most a cubic in y_0). Again using Theorem 7.3.2, we get at most one pair (a_2, i_2) at Step 11 for some $a_2 \in R_0$ and $i_2 \in \mathbb{N}$ (E'_2 is linear in y_0).

Now, for a fixed $z_0 \in Z_r - Z'_r$ we can calculate all z_1's by the equation

$$z_1 \equiv \tilde{z}_1 := -(C(y_0)/L(y_0)) \bmod \langle p, \varphi^{4a-r}\rangle.$$

Here $C(y_0) := E_1(z_0)/\varphi^r \bmod \langle p, \varphi^{4a-r}\rangle$ and $L(y_0) := E_2(z_0)/\varphi^r \bmod \langle p, \varphi^{4a-r}\rangle$. So, $z_1 \in R_0$ comes from the set $z_1 \in \tilde{z}_1 + \varphi^{4a-r}*$. This can be done in $\text{poly}(\deg f, \log p)$ time.

Finally, the sets $Z = a_0 + \varphi^{i_0} Z_r$ and $Z' = a_0 + \varphi^{i_0} Z'_r$, for $(a_0, i_0) \in S_0$ and corresponding valid $r \in \{0, \ldots, 4a-1\}$, returned by Algorithm 6, describe the y_0 for the roots of $E(y_0 + py_1) \bmod \langle p^4, \varphi^{4a}\rangle$. The number of representatives in each of these sets is $O(a)$, since $|S_0| \leq 2$ and sizes of Z_r and Z'_r are only constant.

Since we can efficiently describe these y_0's and corresponding y_1's, and we know their precision, we can count all roots $y = y_0 + py_1 + p^2* \subseteq R$ of $E(y) \bmod \langle p^4, \varphi^{4a}\rangle$. □

7.3.5 Proof of Main Results

We have all the ingredients available to prove our main results.

Proof of Theorem 7.0.1. We prove that given an arbitrary univariate $f \in \mathbb{Z}[x]$ and a prime p, a non-trivial factor of f modulo p^4 can be obtained in randomized $\text{poly}(\deg f, \log p)$ time (or the irreducibility of $f \bmod p^4$ gets certified).

If $f \equiv f_1 f_2 \bmod p$, where f_1, f_2 are two polynomials coprime in $F_p[x]$, then we can efficiently lift this factorization to the ring $(\mathbb{Z}/\langle p^4\rangle)[x]$, using Hensel lemma 2.2.2, to get non-trivial factors of $f \bmod p^4$.

For the remaining case, $f \equiv \varphi^e \bmod p$ for an irreducible polynomial $\varphi(x)$ modulo p. The question of factoring $f \bmod p^4$ then reduces to root finding of a polynomial $E(y) \bmod \langle p^4, \varphi^{4a}\rangle$ by Reduction theorem (Theorem 7.2.1). Using Theorem 7.3.7, we get all such roots and hence

a non-trivial factor of $f \bmod p^4$ is found. If there are no roots $y \in R$ of E, for all $a \leq e/2$, then the polynomial f is irreducible (by symmetry, if there is a factor for $a > e/2$ then there is a factor for $a \leq e/2$). □

Proof of Theorem 7.0.2. We are given a univariate $f \in \mathbb{Z}[x]$ of degree d and a prime p, such that, $f \bmod p$ is a power of an irreducible polynomial $\varphi(x)$. So, f is of the form $\varphi(x)^e + ph(x) \bmod p^4$, for an integer $e \in \mathbb{N}$ and a polynomial $h \in (\mathbb{Z}/\langle p^4\rangle)[x]$ of degree $\leq d$ (also, $\deg \varphi^e \leq d$). By unique factorization over the ring $\mathbb{F}_p[x]$, if $\tilde{g}$ is a factor of $f \bmod p$ then, $\tilde{g} \equiv \tilde{v}\varphi^a \bmod p$ for a unit $\tilde{v} \in \mathbb{F}_p$.

First, we show that it is enough to find all the lifts of $\tilde{g}$, such that, $\tilde{g} \equiv \varphi^a \bmod p$ for an $a \leq e$. If $\tilde{g} \equiv \tilde{v}\varphi^a \bmod p$, then any lift has the form $g(x) \equiv v(x)(\varphi^a - py) \bmod p^4$ for a unit $v(x) \in (\tilde{v} + p*) \subseteq (\mathbb{Z}/\langle p^4\rangle)[x]$. Any such $g(x)$ maps uniquely to a $g_1(x) := \tilde{v}^{-1}g(x) \bmod p^4$, which is a lift of $\varphi^a \bmod p$. So, it is enough to find all the lifts of $\varphi^a \bmod p$.

We know that any lift $g \in (\mathbb{Z}/\langle p^4\rangle)[x]$ of $\tilde{g}(x)$, which is a factor of f, must be of the form $\varphi^a - py \bmod p^4$ for a polynomial $y \in (\mathbb{Z}/\langle p^4\rangle)[x]$. By Reduction theorem (Theorem 7.2.1), we know that finding such a factor is equivalent to solving for y in the equation $E(y) \equiv 0 \bmod \langle p^4, \varphi^{4a}\rangle$. By Theorem 7.3.7, we can find all such roots y in randomized $\mathrm{poly}(d, \log p)$ time, for $a \leq e/2$.

If $a > e/2$ then we replace a by $b := e - a$, as $b \leq e/2$, and solve the equation $E(y) \equiv 0 \bmod \langle p^4, \varphi^{4b}\rangle$ using Theorem 7.3.7. This time the factor corresponding to y will be, $g \equiv f/(\varphi^b - py) \equiv E(y)/\varphi^{4b} \bmod p^4$, using Reduction theorem (Theorem 7.2.1).

The number of lifts of $\tilde{g}(x)$ which divide $f \bmod p^4$ is the count of y's that appear above. This is efficiently computable via Algorithm 6. □

7.4 Barriers to extension modulo higher powers p^k

The reader may wonder about polynomial factoring when k is greater than 4. In this section we will discuss the issues in applying our techniques to factor $f(x) \bmod p^5$.

Given $f \equiv \varphi^e \bmod p$, finding one of its factor $\varphi^a - py \bmod p^5$, for $a \leq e/2$ and $y \in$

$(\mathbb{Z}/\langle p^5\rangle)[x]$, is reduced to solving the equation

$$E \;:=\; f\cdot(\varphi^{4a}+\varphi^{3a}(py)+\varphi^{2a}(py)^2+\varphi^{a}(py)^3+(py)^4) \;\equiv\; 0 \mod \langle p^5,\varphi^{5a}\rangle \tag{7.7}$$

By Lemma 7.3.1, the roots of E mod $\langle p^5,\varphi^{5a}\rangle$ are of the form $y=y_0+py_1+p^2y_2+p^3*$ in R, where $y_0,y_1,y_2\in R_0$ need to be found.

First issue. The first hurdle comes when we try to reduce root-finding modulo the bi-generated ideal $\langle p^5,\varphi^{5a}\rangle\subseteq\mathbb{Z}[x]$ to root-finding modulo the principal ideal $\langle\varphi^{5a}\rangle\subseteq\mathbb{F}_p[x]$. In the case $k=4$, Lemma 7.3.4 guarantees that we need to solve at most two related equations of the form $E'(y_0,y_1)\equiv 0 \mod \langle p,\varphi^{4a}\rangle$ to find exactly the roots of E mod $\langle p^4,\varphi^{4a}\rangle$. Below, for $k=5$, we show that we have exponentially many candidates for $E'(y_0,y_1,y_2)\in R_0[y_0,y_1,y_2]$ and it is not clear if there is any compact efficient representation for them.

Putting $y=y_0+py_1+p^2y_2$ in Eqn. 7.7 we get,

$$E(y) \;=:\; E_1(y_0)+E_2(y_0)y_1+E_3(y_0)y_2+(f\varphi^{2a}p^4)y_1^2 \;\equiv\; 0 \mod \langle p^5,\varphi^{5a}\rangle, \tag{7.8}$$

where $E_1(y_0):=f\varphi^{4a}+f\varphi^{3a}py_0+f\varphi^{2a}p^2y_0^2+f\varphi^{a}p^3y_0^3+fp^4y_0^4$ is a quartic in $R[y_0]$, $E_2(y_0):=f\varphi^{3a}p^2+f\varphi^{2a}2p^3y_0+f\varphi^{a}3p^4y_0^2$ is a quadratic in $R[y_0]$ and $E_3(y_0):=f\varphi^{3a}p^3+f\varphi^{2a}2p^4y_0$ is linear in $R[y_0]$.

To divide Eqn. 7.8 by p^3, we go mod $\langle p^3,\varphi^{5a}\rangle$ obtaining

$$E(y) \;\equiv\; E_1(y_0)\equiv f\varphi^{4a}+f\varphi^{3a}py_0+f\varphi^{2a}p^2y_0^2 \;\equiv\; 0 \mod \langle p^3,\varphi^{5a}\rangle,$$

a univariate quadratic equation which requires the whole machinery used in the case $k=3$. We get this simplified equation since $E_3(y_0)\equiv 0 \mod\langle p^3,\varphi^{5a}\rangle$ and $E_2(y_0)\equiv f\varphi^{3a}p^2\equiv\varphi^{e-2a}\varphi^{2a+3a}p^2\equiv 0 \mod\langle p^3,\varphi^{5a}\rangle$.

But, to really reduce Eqn. 7.8 to a system modulo the principal ideal $\langle\varphi^{5a}\rangle\subseteq\mathbb{F}_p[x]$, we need to divide it by p^4. So, we go mod $\langle p^4,\varphi^{5a}\rangle$:

$$E(y) \;\equiv\; E_1'(y_0)+E_2'(y_0)y_1 \;\equiv\; 0 \mod \langle p^4,\varphi^{5a}\rangle$$

where $E_1'(y_0)\equiv E_1(y_0) \mod\langle p^4,\varphi^{5a}\rangle$ is a cubic in $R[y_0]$ and $E_2'(y_0)\equiv E_2(y_0) \mod\langle p^4,\varphi^{5a}\rangle$ is linear in $R[y_0]$. This requires us to solve a special bivariate equation which requires the

machinery used in the case $k = 4$.

Now, the problem reduces to computing a solution pair $(y_0, y_1) \in (R_0)^2$ of this bivariate equation. We can apply the idea used in Algorithm 6 to get all valid y_0 efficiently, but since y_1 is a function of y_0, we need to compute exponentially many y_1's. So, there seem to be exponentially many candidates for $E'(y_0, y_1, y_2)$, that behaves like $E(y)/p^4$ and lives in $(\mathbb{F}_p[x]/\langle\varphi^{5a}\rangle)[y_0, y_1, y_2]$. At this point, we are forced to compute all these E's, as we do not know which one will lead us to a solution of Eqn. 7.8.

Second issue. Even if we resolve the first issue and get a valid E', we are left with a trivariate equation to be solved mod $\langle p, \varphi^{5a}\rangle$ (Eqn. 7.8 after shifting y_0 and y_1 then dividing by p^4). We could do this when k was 4, because we could easily write y_1 as a function of y_0. Though, it is unclear how to solve this trivariate equation now as it is *nonlinear* in both y_0 and y_1.

For $k > 5$ the difficulty will only increase because of the recursive nature of Eqn. 7.7 with more and more unknowns (with higher degrees).

7.5 Conclusion

The study of [vzGH98, vzGH96] sheds some light on the behaviour of the factoring problem for integral polynomials modulo prime powers. It shows that for "large" k the problem is similar to the factorization over p-adic fields (already solved efficiently by [CG00]). But, for "small" k the problem seems hard to solve in polynomial time. We do not even know a practical algorithm.

This motivated us to study the case of constant k, with the hope that this will help us invent new tools. In this direction, we made significant progress by giving a unified method to factor f mod p^k for $k \leq 4$. We also refined Hensel lifting for $k \leq 4$, by giving all possible lifts of a factor of f mod p, in the classically hard case of f mod p being a power of an irreducible.

We gave a general framework (for any k) to work on, by reducing factoring in a big ring to root-finding in a smaller ring. In the next chapter, we will further extend this reduction to get more general factoring algorithm.

Chapter 8

Low Degree Factoring via Solving System of Polynomials

Efficiently factoring a univariate polynomial $f \in \mathbb{Z}[x]$ modulo a prime power p^k is a major open problem, when k is constant. Chapter 7 established that factoring $f \bmod p^k$ is efficient, when $k \leq 4$. In this chapter we present the first general factoring algorithm which is efficient for constant k but with one restriction– the degree of the factors we find is *constant*. In particular, if f is a constant degree univariate polynomial then we can factor it efficiently. We achieve this by extending the reduction given in Chapter 7 to get a new reduction which is efficient when k is constant. Essentially in the constant k regime, finding a constant degree factor of $f \bmod p^k$ is reduced to finding a common zero of a system of constant-variate polynomials of constant degree modulo p^k. We then use the method of ideals to efficiently solve a system of n-variate polynomial equations over a Galois ring of characteristic p^k when $n + k$ is constant.

8.1 Our Results

In the constant k regime, we efficiently compute a *constant-degree* factor of $f(x) \bmod p^k$ thereby giving the first randomized polynomial time algorithm to factor a *fixed* degree univariate polynomial into irreducibles.

Theorem 8.1.1 (Factoring). *Given a univariate polynomial $f \in \mathbb{Z}[x]$ and a prime-power p^k, in binary, with k fixed. We can find a constant-degree factor g of $f \bmod p^k$ in randomized $\mathsf{poly}(\deg(f), \log p)$-time; or decide that none exists.*

The difficult case in factoring is when $f \equiv \varphi^e \bmod p$ for a $\varphi \in \mathbb{Z}[x]$ which is irreducible mod p. We call e to be the *ramification-degree* of f. Our proof method provides factors of constant ramification degree which are more general factors.

Corollary 8.1.2 (Low ramification factors). *Given $f \in \mathbb{Z}[x]$ and prime-power p^k, with k constant. We can find a factor g of $f \bmod p^k$ in randomized $\mathsf{poly}(\deg(f), \log p)$-time, where the* ramification-degree *of g is at most a given constant; or decide that no such factor exists.*

The brute-force approach takes time $p^{\Omega(k\delta)}$; which is exponential even for fixed k and fixed ramification-degree δ. Our low degree factoring result is achieved by solving Search Hilbert's nullstellensatz SHN over Galois rings when k is constant.

Theorem 8.1.3 (SHN_{p^k}). *Given a system of n-variate polynomials $f_1, \ldots, f_m \in \mathbb{Z}[z, \mathbf{x}]/\langle p^k, \varphi(z)\rangle$ of degrees at most d, for a prime p; and an irreducible polynomial $\varphi(z) \in \mathbb{F}_p[z]$ defining the Galois ring $\mathbb{G} := \mathbb{Z}[z]/\langle p^k, \varphi(z)\rangle$. We can find a common root of the system in $\mathbb{G}$, in randomized $\mathsf{poly}(d^{c_{nk}}, m, \deg(\varphi)\log p)$-time; where $c_{nk} \leq (nk)^{O((nk)^2)}$.*

Theorem 8.1.3 is efficient when $n + k$ is constant. Recall that even if $k = 1$, decision version of SHN is NP-hard for unbounded n.

8.2 Finding Low Degree Factors modulo Small Prime Powers

In this section we will prove Theorem 8.1.1 and Corollary 8.1.2 i.e, we show how to efficiently find a 'low' ramification-degree factor of $f(x) \bmod p^k$ in randomized polynomial time. We achieve this via first reducing the problem, in Sections 8.2.1 and 8.2.2, to finding a common zero of a system of multivariate polynomial equations over a Galois ring of characteristic p^k.

Assume the input $f \in \mathbb{Z}[x]$ to be monic mod p^k (leading coefficient 1) as we can always remove the factors, which are units in the ring $(\mathbb{Z}/\langle p^k\rangle)[x]$, by division. Also assume $f \equiv \varphi(x)^e + p \cdot h(x) \bmod p^k$ (i.e, $f \equiv \varphi^e \bmod p$), where $\varphi \in \mathbb{Z}[x]$ is an irreducible polynomial

over $\mathbb{F}_p$. Otherwise, using coprime factorization mod p, we can efficiently find a non-trivial factor of $f \bmod p^k$ using Hensel's Lemma 2.2.2. Let $b := \deg(\varphi)$, with $\deg(f) = b \cdot e$ and $\deg(h) < \deg(f)$.

In the previous chapter, finding a ramification-degree δ factor is reduced to finding a root of an $E(y) \in \mathbb{Z}[x, y]$ modulo a bi-generated ideal $\langle p^k, \varphi(x)^\ell \rangle$ where $\deg_y(E) < k$ and $\ell = \delta \cdot k$. In Section 8.2.1, we will focus on this root-finding job and reduce this to root finding modulo a simpler ideal $\langle p^k, \varphi(z), (x-z)^\ell \rangle$. Then in Section 8.2.2, we further reduce this problem to solving a system of multivariate polynomial equations modulo $\langle p^k, \varphi(z) \rangle$ (namely, over the Galois ring).

8.2.1 Factoring over the Galois Ring

We have $f = \varphi^e + ph$ and prime power p^k. Consider the Galois ring $\mathbb{G} := \mathbb{Z}[z]/\langle p^k, \varphi(z) \rangle$ where $z \in \mathbb{G}$ be a root of the polynomial $\varphi(x)$. Denote the roots of $\varphi(x)$ in $\mathbb{G}$ by z_i with $z_0 := z$ for $i \in \{0, \ldots, b-1\}$ (recall $b = \deg(\varphi)$). Then, we know that $z_i \equiv z^{p^i} \bmod p$ for all $i \in \{0, \ldots, b-1\}$. Let us denote the simpler Galois ring $\mathbb{Z}/\langle p^k \rangle$ by $\mathbb{G}_0$.

By Lemma 2.2.2, f in $\mathbb{G}$ factors as $f = \prod_{i=0}^{b-1} f_i$, where $f_i(x) = (x - z_i)^e + ph_i(x)$ in $\mathbb{G}$. In particular, $f_0 = (x-z)^e + ph_0(x)$. We now use Proposition 2 to prove the following two lemmas, for connecting ramified factors of f in $\mathbb{G}_0[x]$ to ramified factors of f_i's in $\mathbb{G}[x]$.

Notation: We often denote $u(x, z) \in \mathbb{G}[x]$ by $u(z)$ to highlight the relevant parameter 'z'.

Lemma 8.2.1. *If $(\varphi^\delta - py) \mid f(x) \bmod p^k$, for $y \in \mathbb{G}_0[x]$, then for some $u(x, z) \in \mathbb{G}[x]$, $((x - z_i)^\delta - pu(z_i)) \mid f_i(x) \bmod \langle p^k, \varphi(z) \rangle$, for each $i \in \{0, \ldots, b-1\}$.*

Proof. Let $g := \varphi^\delta - py$. Then $g \mid f$ in $\mathbb{G}_0[x]$ and so in $\mathbb{G}[x]$. Now $(x-z)^\delta$ is a factor of $g \bmod p$, as $g \equiv \varphi^\delta \bmod p$, and so there is an $u \in \mathbb{G}[x]$ such that $((x-z)^\delta - pu(z))$ is a factor of g (Hensel Lemma 2.2.2); thus factor of f (since $g \mid f$) in $\mathbb{G}[x]$. Applying Proposition 2, we see that $g_i := ((x - z_i)^\delta - pu(z_i))$ is a factor of f, for each $i \in \{0, \ldots, b-1\}$. Now, g_i divides only $f_i \bmod p$ (by Hensel Lemma 2.2.2); and this finishes the proof. □

Lemma 8.2.2. *If there exists $u \in \mathbb{G}[x]$ s.t. $((x-z)^\delta - pu(z)) \mid f(x) \bmod \langle p^k, \varphi(z) \rangle$ then we can compute a $y \in \mathbb{G}_0[x]$ such that $(\varphi^\delta - py) \mid f(x) \bmod p^k$.*

Proof. Let $g_0 := (x-z)^\delta - pu(z)$, and $g_i := (x-z_i)^\delta - pu(z_i)$, for all $i \in [b-1]$. By applying automorphisms ψ_i (Proposition 2) on g_0, for $i \in [b-1]$, we can easily compute all other g_i's. Also, by applying automorphisms ψ_i, for $i \in [b-1]$, we see that each g_i divides $f(x)$ in $\mathbb{G}[x]$ (since ψ_i keeps $\mathbb{G}_0$ fixed and $f \in \mathbb{G}_0[x]$).

Now define $g(x,z) := \prod_{i=0}^{b-1} g_i$ in $\mathbb{G}[x]$. We see that all g_i's are coprime, since they are coprime over the field $\mathbb{G}/\langle p\rangle$ (i.e, $(x-z_i)^\delta$ is co-prime to $(x-z_j)^\delta$ for $i \neq j$). Hence, $g(x,z) \mid f$ in $\mathbb{G}[x]$.

Applying map ψ_1 on $g(x,z)$ we see that $g(x,z)$ remains unchanged over $\mathbb{G}$; as g_i's permute among each other. But ψ_1 keeps $\mathbb{G}_0$, and only $\mathbb{G}_0$, fixed (Proposition 2); hence $g \in \mathbb{G}_0[x]$ of degree $\delta \cdot b$. So, we can rewrite g as $g =: \varphi^\delta - py$, for a $y \in \mathbb{G}_0[x]$. □

The following extension of Reduction Theorem 7.2.1, from $\mathbb{G}_0$ to the Galois ring $\mathbb{G}$, is evident.

Theorem 8.2.3 (Extended Reduction)**.** *We have* $((x-z_i)^\delta - pu(z_i)) \mid f_i(x) \bmod \langle p^k, \varphi(z)\rangle$ *iff* $E(u) \equiv 0 \bmod \langle p^k, \varphi(z), (x-z_i)^\ell\rangle$*, for all* $i \in \{0,\ldots,b-1\}$*; where* $\ell := \delta \cdot k$ *and* $E(u) := f_i(x)[(x-z_i)^{\delta(k-1)} + (x-z_i)^{\delta(k-2)}(pu) + \ldots + (pu)^{k-1}]$.

Thus we now focus on finding a root of $E(u)$ in the ring $\mathbb{G}[x]/\langle (x-z)^\ell\rangle$.

8.2.2 Reduction to Root Finding in Galois Rings

In this section we show that finding a root of polynomial $E(u)$, in the ring $\mathbb{G}[x]/\langle (x-z)^\ell\rangle$, is equivalent to solving a system of ℓ polynomial equations in ℓ variables of degree same as $\deg_y(E) \leq k-1$ over Galois ring $\mathbb{G}$. We achieve this by simply *eliminating* the variable x.

Theorem 8.2.4 (Reduction to HN)**.** *Given* $E(u) \in (\mathbb{Z}[z,x])[u]$ *and the ring* $\mathbb{G}[x]/\langle (x-z)^\ell\rangle$ *where* $\mathbb{G} = \mathbb{Z}[z]/\langle p^k, \varphi(z)\rangle$ *as before. For new variable tuple* $\mathbf{u} = (u_0,\ldots,u_{\ell-1})$ *define a polynomial* $E_{new}(\mathbf{u}) \in (\mathbb{Z}[z,x])[\mathbf{u}]$ *as* $E_{new}(\mathbf{u}) := E(u_0 + (x-z)u_1 + \cdots + (x-z)^{\ell-1}u_{\ell-1})$.

Let $\mathcal{F}(\mathbf{u}) := \{E_0,\ldots,E_{\ell-1}\}$ *be a system of polynomial equations, where* $E_i(\mathbf{u}) \in (\mathbb{Z}[z])[\mathbf{u}]$ *with* $\deg_z(E_i) < \deg(\varphi(z))$ *and* $\deg_{\mathbf{u}}(E_i) < k$*, such that*

$$E_{new}(\mathbf{u}) \equiv E_0(\mathbf{u}) + E_1(\mathbf{u})\cdot(x-z) + \cdots + E_{\ell-1}(\mathbf{u})\cdot(x-z)^{\ell-1} \bmod \langle p^k, \varphi(z), (x-z)^\ell\rangle.$$

Then for $\mathbf{a} \in \mathbb{G}^\ell$, $E_{new}(\mathbf{a}) \equiv 0 \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$ *iff* $\mathcal{F}(\mathbf{a}) \equiv 0 \bmod \langle p^k, \varphi(z) \rangle$.

Proof. Following the definition of $E_{\mathsf{new}}(\mathbf{u})$, we can rewrite $E_{\mathsf{new}}(\mathbf{u})$, for some polynomials $E_i(\mathbf{u}) \in (\mathbb{Z}[z])[\mathbf{u}]$ as

$$E_{\mathsf{new}}(\mathbf{u}) \;=\; E_0(\mathbf{u}) + E_1(\mathbf{u})(x-z) + \cdots + E_{\ell-1}(\mathbf{u})(x-z)^{\ell-1} .$$

Now, $E_{\mathsf{new}}(\mathbf{a}) \equiv 0 \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$

$\iff$ $E_{\mathsf{new}}(\mathbf{a}) =: t_x(x-z)^\ell$, for some $t_x \in \mathbb{G}[x]$.

$\iff$ $E_0(\mathbf{a}) + \cdots + (x-z)^{\ell-1} E_{\ell-1}(\mathbf{a}) \;=\; t_x(x-z)^\ell$.

Since degree wrt x of LHS is at most $\ell - 1$, so $(x-z)^\ell$ can not divide it over $\mathbb{G}$. So $E_i(\mathbf{a})$ vanishes in $\mathbb{G}$, for each $i \in \{0, \ldots, \ell-1\}$. In other words, $\mathbf{a}$ is $\mathbb{G}$-root of the system $\mathcal{F}(\mathbf{u})$.

Now we prove the other direction. Given that, $E_i(\mathbf{a}) \equiv 0 \bmod \langle p^k, \varphi(z) \rangle$, for each $i \in \{0, \ldots, \ell-1\}$. We easily deduce: $E_{\mathsf{new}}(\mathbf{a}) \equiv 0 \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$.

Moreover, this reduction is efficient when the parameter k is fixed; because $\deg_{\mathbf{u}}(E) < k$ and so E_{new} has at most $\binom{\ell+k}{\ell} \le (\ell+k)^k$ monomials. □

8.2.3 Algorithm and Proofs

Input: Given $f \in \mathbb{Z}[x]$ and a prime-power p^k such that $f \equiv \varphi^e \bmod p$, where $\varphi \in \mathbb{Z}[x]$ is irreducible mod p; and $\deg(f) = b \cdot e$, where $b := \deg(\varphi)$.

Output: A ramification-degree-δ factor $g(x)$ of $f(x) \bmod p^k$.

Algorithm 7 Factoring $f(x) \bmod p^k$

1: **procedure** FACTOR($f(x), p^k$)

2: Let $g = \varphi^\delta - p \cdot y$, where $y = y(x)$ is an unknown such that $g \mid f \bmod p^k$.

3: Consider Galois ring $\mathbb{G} := \mathbb{Z}[z]/\langle p^k, \varphi(z) \rangle$, where $\varphi(x)$ splits completely and z is a $\mathbb{G}$-root of $\varphi(x)$. (Other roots are conjugates of z, by Proposition 2.)

4: Factorize $\varphi(x)$ over $\mathbb{G}/\langle p \rangle$ into b linear (coprime) factors using Theorem 2.2.1 and lift to $\mathbb{G}$ using Hensel's lifting to obtain a coprime factorization $f =: \prod_{i=0}^{b-1} f_i$.

5: Over $\mathbb{G}$, let $g =: \prod_{i=0}^{b-1} g_i$ be a coprime factorizations, such that $g_i \mid f_i$ for all i (Lemma 8.2.1). Fix $j \in \{0, \ldots, b-1\}$ and consider $g_j =: (x-z)^\delta - pu$.

6: Using Theorem 8.2.3 reduce to root-finding question of $E(u) \equiv 0 \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$, where $E(u) := f_j \cdot [(x-z)^{\delta(k-1)} + (x-z)^{\delta(k-2)}(pu) + \cdots + (pu)^{k-1}]$.

7: Substituting $u \to u_0 + (x-z)u_1 + \cdots + (x-z)^{\ell-1}u_{\ell-1}$, compute $E_0(\mathbf{u}), \ldots, E_{\ell-1}(\mathbf{u}) \in \mathbb{G}[\mathbf{u}]$ such that

$E(u) =: E_0 + (x-z)E_1 + \cdots + (x-z)^{\ell-1}E_{\ell-1} \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$.

8: Find a $\mathbb{G}$-root $(a_0, \ldots, a_{\ell-1})$ of the system $\mathcal{F} := \{E_0, \ldots, E_{\ell-1}\}$ using Algorithm 9.

9: **if** no solution exists **then return** $\{\}$, i.e. no such factor g exists.

10: $u := a_0 + (x-z)a_1 + \cdots + (x-z)^{\ell-1}a_{\ell-1}$ is a solution of $E(u) \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$ (from Theorem 8.2.4). This gives us the factor $g_j = (x-z)^\delta - pu$ (Theorem 8.2.3).

11: Using $\mathbb{G}$-automorphisms (Lemma 8.2.2 & Step 4), we can compute $g = \varphi^\delta - py$ from g_j.

12: **return** g

Remark 1. *One can ask for a simpler Nullstellensatz approach: Why do we not reduce root-finding of* $E(u) \bmod \langle p^k, \varphi(z), (x-z)^\ell \rangle$ *to directly solving a system of equations modulo* p*, instead of modulo* p^k*? For e.g., by further substituting* $u_i \to u_{i,0} + pu_{i,1} + \ldots + p^{k-1}u_{i,k-1}$*, for each* $i \in \{0, \ldots, \ell-1\}$*,* $u_{i,j}$*'s in* $\mathbb{F}_p$*?*

The issue is that we need to divide functions of $u_{i,j}$*'s by* p*; and this only makes sense when we think of* $u_{i,j}$*'s as* p*-adic.*

Now we prove Theorem 8.1.1 in a way that already subsumes Corollary 8.1.2.

Proof of Theorem 8.1.1. We have $f(x) = \varphi(x)^e + ph(x)$ and prime-power is p^k. A factor g of $f \bmod p^k$ has the form $g = \varphi^\delta - py$ (ramification-degree δ) where we want to compute $y \in \mathbb{G}_0[x]$ such that $\deg(y) < \delta \deg(\varphi)$; to keep g monic.

Now over $\mathbb{G}$, f and g have coprime factorizations as $f = \prod_{j=0}^{b-1} f_j$ and $\prod_{j=0}^{b-1} g_j$. By Lemma 8.2.1 if $g \mid f \bmod p^k$ then $g_j \mid f_j$ over $\mathbb{G}$, for all j. For a fixed $i \in \{0, \ldots, b-1\}$, let $f_i =: (x-z)^e + ph_i(x,z)$ and $g_i =: (x-z)^\delta - pu(x,z)$ (where u is unknown). Using Lemmas 8.2.1 and 8.2.2, it is sufficient to find unknown g_i. Computing factorizations of f and φ (using Hensel lifting 2.2.2) and getting g from g_i (Lemma 8.2.2) takes time $\mathsf{poly}(\deg(f), k \log p)$.

Using Theorem 8.2.3, finding g_i is reduced to finding a root, of $E(u) := f_i \cdot [(x-z)^{\delta(k-1)} + (x-z)^{\delta(k-2)}(pu) + \cdots + (pu)^{k-1}]$, in $\mathbb{G}[x]/\langle (x-z)^{\ell} \rangle$, where $\ell := \delta k$. Computing $E(u)$ takes time $\mathsf{poly}(\deg(f), \ell, \log p)$.

By Theorem 8.2.4, finding a root of $E(u)$ in $\mathbb{G}[x]/\langle (x-z)^{\ell} \rangle$ is reduced to finding $\mathbb{G}$-root of a system of ℓ-variate ℓ polynomial equations $\mathcal{F} := \{E_0(\mathbf{u}), \ldots, E_{\ell-1}(\mathbf{u})\}$ of degree at most $k-1$. Using Theorem 8.1.3, we get a solution of $\mathcal{F}$ in $\mathbb{G}$. This immediately gives us a root u of $E(u)$ mod $\langle p^k, \varphi(z), (x-z)^{\ell} \rangle$; thus we find the factor $g_i = (x-z)^{\delta} - pu$. The time complexity is dominated by time taken to find a solution of $\mathcal{F}$; which is $\mathsf{poly}(\deg(\mathcal{F})^{(\ell k)^{O((\ell k)^2)}}, \log p, \deg(f))$.

Since $\deg(\mathcal{F}) < k$ and $\ell = \delta k$, so the total time taken is $\mathsf{poly}(k^{(\delta k^2)^{O((\delta k^2)^2)}}, \log p, \deg(f))$. Since $\delta + k$ is constant, the time complexity becomes $\mathsf{poly}(\deg(f), \log p)$. □

8.3 Solving System of Polynomial Equations over Galois Rings

We are given a system of n-variate polynomials $\mathcal{F} := \{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\}$ where $f_i(\mathbf{x}) \in \hat{\mathbb{G}}[\mathbf{x}]$; $\hat{\mathbb{G}} \cong \mathbb{Z}_p[z]/\langle \varphi(z) \rangle$ is an unramified p-adic extension of $\mathbb{Z}_p$ ($\varphi \in \mathbb{Z}_p[z]$ is a monic univariate which is irreducible modulo p). We want to find a common root of $\mathcal{F}$ in $\mathbb{G} := \hat{\mathbb{G}}/\langle p^k \rangle$ for given prime power p^k. Note that $\mathbb{G}$ is a Galois ring and $\hat{\mathbb{G}}$ is its $\mathbb{Z}_p$-lift. Also $\mathbb{G}/\langle p \rangle$ is equivalent to a finite field $\mathbb{F}_q \cong \mathbb{Z}[z]/\langle p, \varphi(z) \rangle$ where $q = p^b$ and $b := \deg(\varphi)$.

This thesis contributes in developing the ideas and algorithms (Algorithm 9 and 8) for root finding of $\mathcal{F}$ mod p^k while giving proofs of only some important lemmas and theorems. Rest of the proofs are only sketched and the ideas behind them explained. Remaining part of the proofs have been developed in Chakrabarti's master's thesis [Cha22]. So we will occasionally refer the reader to [Cha22] and the original paper [CDS22].

The section organisation is as follows: First in Section 8.3.1 we give some notations and preliminary results to be used in later sections. Then in Section 8.3.2 we give the outline and set the agenda for forthcoming sections. Section 8.3.4 gives the main algorithm for the problem while the Section 8.3.3 gives a supporting 'decomposition' algorithm to be used as a subroutine in the main algorithm. Finally in Section 8.3.5 we prove our main Theorem 8.1.3 and give its complexity.

8.3.1 Notations and Preliminaries

For an n-tuple $\mathbf{a} = (a_1, \ldots, a_n)$ and $\mathbf{b} = (b_1, \ldots, b_n)$ in $\mathbb{F}^n$, we have the following notations:

- $c\mathbf{a} + d\mathbf{b} = (ca_1 + db_1, \ldots, ca_n + db_n)$ for scalars $c, d \in \mathbb{F}$,
- $|\mathbf{a}| = \Sigma_i a_i$ and $\mathbf{a}! = a_1! \cdots a_n!$, where $\mathbf{a} \in \mathbb{Z}^n$.

Definition 8.3.1 (Taylor expansion/series)**.** *For a polynomial $f(\mathbf{x})$ of degree d over any field, we can express it as*

$$f(\mathbf{a}+\mathbf{x}) \;=\; \sum_{\ell=0}^{\infty} \left(\sum_{|\mathbf{i}|=\ell} \frac{\partial_{\mathbf{x}^\mathbf{i}} f(\mathbf{a})}{\mathbf{i}!} \cdot \prod_{j=1}^{n} x_j^{i_j} \right), \tag{8.1}$$

where $\partial_{\mathbf{x}^\mathbf{i}} f := \frac{\partial^{|\mathbf{i}|} f}{\partial x_1^{i_1} \ldots \partial x_n^{i_n}}$ is an order-$|\mathbf{i}|$ *partial derivative.*

A root $\mathbf{r}$ of a polynomial $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ is called *singular* or *non-simple* if all the first-order derivatives $f'(\mathbf{x})$ (i.e., $\frac{\partial f}{\partial x_j}$ for all $j \in [n]$) vanish at $\mathbf{r}$, otherwise it is called a *simple* or a *non-singular* root.

A polynomial $h \in \mathbb{F}[x]$ which does not factor over its defining field $\mathbb{F}$ is simply called an *irreducible polynomial*. An irreducible polynomial $h \in \mathbb{F}[\mathbf{x}]$ is called *absolutely* irreducible if it remains irreducible over algebraic closure $\bar{\mathbb{F}}$ otherwise it is called *relatively* irreducible. This notion is particularly distinguished when the polynomial has more than one variable. An irreducible univariate polynomial of degree more than one is always *relatively irreducible* as it must factor over the algebraic closure of the defining field. This notion of irreducibility also extends to polynomial ideals.

Lemma 8.3.2 (Folklore)**.** *If $h \in \mathbb{F}[\mathbf{x}]$ is an irreducible polynomial and one of its zero is non-singular then h must be absolutely irreducible. In other words, all the zeros of a relatively irreducible polynomial are* singular *i.e., if $h(\mathbf{a}) = 0$ then $h'(\mathbf{a}) = 0$ where $h'(\mathbf{x})$ is any non-zero first order derivative.*

The following lemma tells that an absolutely irreducible polynomial has a lot of roots.

Theorem 8.3.3 (Number of roots [Sch74])**.** *An absolutely irreducible polynomial $f(\mathbf{x})$ in n variables of degree d has number of roots in the range, $q^{n-1} \pm ((d-1)(d-2)q^{n-1.5} + 6d^2 q^{n-2})$, over a big enough finite field $\mathbb{F}_q$ (namely, $q > \omega(n^3 d^5)$).*

Following Lemma gives further properties of absolutely irreducible polynomials.

Lemma 8.3.4 (Hensel lifting)**.** *An absolutely irreducible polynomial $f(\mathbf{x})$ in n variables of degree d has at most $O(d^2 \cdot q^{n-2})$ singular roots over a big enough finite field $\mathbb{F}_q$ i.e., almost all the roots are non-singular. Further, each non-singular root can be efficiently lifted to $\mathbb{G}$.*

Proof sketch. We will only sketch the proof (omitting minute details). Following the definition of singular roots we can see that they are exactly the common roots of f and f' (any non-zero first order derivative). The algebraic set $\mathbf{V} := \mathbf{V}_{\mathbb{F}_q}(\langle f, f' \rangle)$ has dimension at most $n-2$ as f is irreducible so it has no common factor with f' ($\deg(f') \leq d-1$). $\mathbf{V}$ may not be irreducible but it can have at most $\deg(f) \cdot \deg(f') < d^2$ absolutely irreducible components of dimension $n-2$ ([HW99, Lemma 2.4]) where each such component have $O(q^{n-2})$ rational points (Theorem 8.3.3).

Further, Taylor expansion of $f(\mathbf{a} + p\mathbf{x})$, for a non-singular root $\mathbf{a}$, contains only linear terms when divided by p and reduced modulo p as at least one first order derivative is non-zero (by the definition of non-singular roots). Linear multivariate polynomials are always absolutely irreducible and so they have a lot of non-singular roots. Pick any such root at random and repeat the process on $g(\mathbf{x}) := f(\mathbf{a} + p\mathbf{x})/p$ to get a lift of $\mathbf{a}$ over $\mathbb{Z}_p$. □

Decomposition Algorithm [HW99]. We will heavily utilise an algorithm (and related theorems) due to [HW99, Section 3] which extracts all the irreducible components of an algebraic set described by a given polynomial system.

The algorithm takes as input a set of multivariate polynomials $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$ each of total degree bounded by d. The algorithm outputs each irreducible component of the algebraic set $\mathbf{V}_{\mathbb{F}}(\langle f_1, \ldots, f_m \rangle)$ in the form of a birationally equivalent hypersurface. It is a classical result in algebraic geometry that an irreducible algebraic set $\mathtt{W}$ of dimension r is birationally equivalent to a hypersurface $\mathtt{H} := \mathbf{V}_{\mathbb{F}}(\langle h \rangle)$ where h is an irreducible polynomial in $r+1$ variables. Precisely, for each $r \leq n$ and for each r-dimensional component $\mathtt{W}$ of $\mathbf{V}_{\mathbb{F}}(\langle f_1, \ldots, f_m \rangle)$ the algorithm outputs a polynomial $h \in \mathbb{F}[z_0, z_1, \ldots, z_r]$ such that $\mathtt{H} := \mathbf{V}_{\mathbb{F}}(\langle h \rangle)$ is birationally isomorphic to $\mathtt{W}$. The algorithm also returns the birational morphism $\psi_2 : \mathtt{W} \to \mathtt{H}$ and its inverse $\psi_1 : \mathtt{H} \to \mathtt{W}$.

Following Theroem 8.3.5 ([HW99, Theorem 2.6]) gives the complexity of the decomposition algorithm. The complexity bounds hold true over any field $\mathbb{F}$ where a randomized polynomial time algorithm exists for polynomial factorization e.g., finite fields and p-adic fields. The theorem also assumes that the size of the field $|\mathbb{F}| \geq d^{cn^2}$ for some constant c. These assumptions are without loss of generality in our case as we work over finite field and p-adic field and smaller field size will make our main theorems work by brute force within the given time complexity.

Theorem 8.3.5 (Simplified [HW99, Theorem 2.6])**.** *Given a system of polynomials $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$ of total degree bounded by d. Then the decomposition algorithm [HW99, Section 3.3] will construct a birational hypersurface $\mathtt{H}$ for each irreducible component $\mathtt{W}$ of $\mathbf{V}_{\mathbb{F}}(\langle f_1, \ldots, f_m \rangle)$ in $m^{O(1)}d^{O(n^2)}$ field operations. Furthermore, the total degree of $\mathtt{H}$ as well as polynomials appearing in rational functions ψ_1 and ψ_2 is upper bounded by $d^{O(n)}$.*

Gröbner Basis. We require some concepts of Gröbner basis in our algorithm to find 'special' lifts to $\widehat{\mathbb{G}}$ (in Lemma 8.3.7). Modulo multivariate polynomial ideals, the remainder on division is not always unique. Thus, we modify the ideal by adding some more generators, depending on a given *ordering* of variables, such that the remainder modulo the ideal is unique.

For a given ideal $\mathtt{I}$, the *S-polynomial* of two polynomials g_1, g_2 in $\mathtt{I}$ is defined as

$$S(g_1, g_2) = \frac{\mathrm{lcm}(\mathrm{LM}(g_1), \mathrm{LM}(g_2))}{\mathrm{LT}(g_1)} \cdot g_1 \; - \; \frac{\mathrm{lcm}(\mathrm{LM}(g_1), \mathrm{LM}(g_2))}{\mathrm{LT}(g_2)} \cdot g_2 \; , \tag{8.2}$$

where LM denotes the leading monomial and LT denotes the leading term.

Buchberger [Buc65] gave the famous algorithm to compute the (reduced) Gröbner basis; by considering every pair of current generators of the ideal and iteratively adding their S-polynomials; until the S-polynomials are zero. More properties of Gröbner basis, and their complexity, can be found in [CLO13].

8.3.2 The Outline

A standard way to find a root modulo p^k is to first find a root in the base field i.e., modulo p and then try lifting it gradually to higher powers of p. As discussed in Chapter 1, even

a single bivariate polynomial can have q many zeros modulo p. So it becomes unclear how to efficiently pick a root modulo p, out of exponentially many (in $\log q$), such that it lifts to modulo p^k as there seems to be no particular structure behind this. For example $(x-y)^2+p$ has no zero modulo p^2 i.e., no zero modulo p lifts to modulo p^2. While in another example $(x-y)^2+px$, the only zero $(0,0)$ lifts to modulo p^2 out of p many zeros modulo p. Thus the idea of picking a random root modulo p and try lifting it will not work.

We use our method of ideals to solve the problem similar to the problem of univariate root counting in Part I. In other words, the method gives a set of special ideals, over $\mathbb{Z}_p$, in nk variables (like split ideals for root count) which collectively contain all the roots of $\mathcal{F}$ mod p^k. We call these ideals 'absolute'-ideals as these are prime ideals and absolutely irreducible when reduced modulo p. The absolute irreducibility of the ideals helps in two ways: (1) Zeroset of an absolutely irreducible ideal has a 'lot' of points and so randomly picking a point modulo p hits to one of its zero and, (2) almost all the points of an absolutely irreducible zeroset modulo p are non-singular points which facilitates Hensel lifting to lift a randomly picked point to $\mathbb{G}$. This gives a randomized method to efficiently pick a zero of $\mathcal{F}$ mod p^k from absolute-ideals.

The algorithm we present has similar outline as for univariate root counting but the implementation details differ quite much and pose a lot of problems to tackle. Our framework previously helped in derandomization for univariate root counting and now it is helping in efficient random sampling.

8.3.3 Decomposition into Ideals with 'Local Properties'

Ideally, we want $\hat{\mathbb{G}}$ ideals $\hat{\mathtt{I}}$ such that its $\mathbb{F}_q$-version $\mathtt{I}$ has only non-singular roots. But this is not always possible even if $\mathtt{I}$ is absolutely irreducible. For example the absolutely irreducible polynomial x^2-y^3 over any $\mathbb{F}_q$ has a singular root $(0,0)$ as both the first order derivatives vanish at $(0,0)$. Algorithm 8 approaches the problem by decomposing the $\mathbb{F}_q$-ideal $\mathtt{I}$ into a set $\mathcal{S}_{abs}$ of absolutely irreducible ideals such that every zero in $\mathbf{V}_{\mathbb{F}_q}(\mathtt{I})$ appears as a non-singular zero of some $\mathtt{C} \in \mathcal{S}_{abs}$ while the zeroset $\mathbf{V}_{\mathbb{F}_q}(\mathtt{I})$ remains unchanged, i.e, $\mathbf{V}_{\mathbb{F}_q}(\mathtt{I}) = \bigcup_{\mathtt{C}\in\mathcal{S}_{abs}} \mathbf{V}_{\mathbb{F}_q}(\mathtt{C})$. There is a blow-up in the number of newly produced ideals $\mathtt{C}$ which is only polynomial when n (and k) is constant (Lemma 8.3.8, Section 8.3.5).

Input: A radical ideal $\mathtt{I} \subseteq \mathbb{F}_q[y_1, \ldots, y_n]$.

Output: A set $\mathcal{S}_{abs}$ of absolutely irreducible ideals $\mathtt{C}$, such that $\mathbf{V}(\mathtt{I}) = \bigcup_{\mathtt{C} \in \mathcal{S}_{abs}} \mathbf{V}(\mathtt{C})$.

Algorithm 8 Decomposing $\mathtt{I}$ into absolutely irreducible components over $\mathbb{F}_q$.

1: **procedure** ABS_DECOMP($\mathtt{I}$)
2: Define $\mathcal{S}_{abs} := \{\}$ and $\mathcal{S}_{irr} := \{\}$.
3: Decompose $\mathtt{I}$ into irreducible components over $\mathbb{F}_q$ using Theorem 8.3.5 ([HW99]), and store them in $\mathcal{S}_{irr}$.
4: **while** $\mathcal{S}_{irr} \neq \emptyset$ **do**
5: $\mathtt{C} \leftarrow \mathtt{pop}(\mathcal{S}_{irr})$. /* Remove $\mathtt{C}$ from $\mathcal{S}_{irr}$*/
6: **if** $\dim(\mathbf{V}(\mathtt{C})) = 0$ **then**
7: Compute $\mathbf{V}(\mathtt{C})$ using [HW99] and for each $\mathbf{a} \in \mathbf{V}(\mathtt{C})$, update $\mathcal{S}_{abs} \leftarrow \mathcal{S}_{abs} \cup \{\langle \mathbf{y} - \mathbf{a} \rangle\}$.
8: **else**
9: **if** $\mathtt{C}$ is absolutely irreducible **then**
10: $\mathcal{S}_{abs} \leftarrow \mathcal{S}_{abs} \cup \{\mathtt{C}\}$
11: Let $\dim(\mathbf{V}(\mathtt{C})) =: r$. Using Theorem 8.3.5 compute a birationally equivalent hypersurface $\mathtt{H} := \mathbf{V}_{\mathbb{F}_q}(h(\ell_1, \ldots, \ell_r, Y))$ and the rational maps $\psi_1 : \mathtt{H} \to \mathbf{V}(\mathtt{C})$ and $\psi_2 : \mathbf{V}(\mathtt{C}) \to \mathtt{H}$. ($\ell_1, \ldots, \ell_r, Y$ are linear forms in $\mathbf{y}$; also see Figure 8.1.)
12: Compute $\mathtt{C}_1 := \mathsf{Rad}(\mathtt{C} + \langle h^* \rangle)$, where h^* is pullback of a first-order partial-derivative $h' \neq 0$.
13: Compute $\mathtt{C}_2 := \mathsf{Rad}(\mathtt{C} + \langle e^* \rangle)$, where e^* is the pullback of e, which is a product of the denominators that appear– in rational functions $\psi_1 =: (\psi_{1,1}, \ldots, \psi_{1,n})$, or in the localization done in Lemma 8.3.7.
14: Decompose the ideals $\mathtt{C}_1, \mathtt{C}_2$ into irreducible components over $\mathbb{F}_q$ using Theorem 8.3.5, and push these components into $\mathcal{S}_{irr}$.
15: **return** $\mathcal{S}_{abs}$

There is a disclaimer though. The decomposition by the Algorithm 8 is not same as the traditional decomposition (as in Theorem 8.3.5). The zero set of one component $\mathtt{C}$ could be contained in the zero set of a different component $\mathtt{C}'$. Conversely a component $\mathtt{C}'$ could be

contained in a component $\mathtt{C}$. For above example $\mathtt{I} = \langle x^2 - y^3 \rangle$ the Algorithm 8 may produce two absolutely irreducible ideals $\mathtt{C} = \langle x, y \rangle$ and $\mathtt{C}' = \langle x^2 - y^3 \rangle$ where the *only* singular root $(0, 0)$ of $\mathtt{I}$ is captured as a non-singular root of $\mathtt{C}$. But $\mathbf{V}_{\mathbb{F}_q}(\mathtt{C}) \subseteq \mathbf{V}_{\mathbb{F}_q}(\mathtt{C}')$ and $\mathtt{C}' \subseteq \mathtt{C}$.

Two varieties are birationally equivalent if and only if their function field are isomorphic. A birational equivalence relation between two varieties is not an isomorphism but informally saying, it connects most of the points of the two sides. It also preserves the important properties such as dimension and absolute irreducibility.

Let us understand steps 12 and 13 of Algorithm 8. The algorithm follows the ideas of [HW99] although in a slightly different way. A singular root of h is contained in its derivative h'. And so the pullback h^* of h' given by our birational map captures the corresponding points of $\mathtt{C}$. Hence, adding $\langle h^* \rangle$ to $\mathtt{C}$ produces an ideal $\mathtt{C}_1$, of lesser dimension, which has only the singular roots of $\mathtt{C}$.

As birational maps do not map all the points (of $\mathbf{V}_{\mathbb{F}_q}(\mathtt{C})$ to $\mathtt{H}$) we also need to extract out these points. Fortunately, these points are not much and are captured as roots of polynomial e (product of the denominator polynomials in the rational function ψ_1). So the variety of pullback polynomial e^* i.e., $\mathbf{V}_{\mathbb{F}_q}(e^*)$ contains such points of $\mathbf{V}_{\mathbb{F}_q}(\mathtt{C})$. Thus $\mathtt{C}_2$ gives that ideal of lesser dimension. Note that at every iteration of the while loop, an ideal is removed from $\mathcal{S}_{irr}$ and a finite number of ideals of lesser dimension are added (decomposed $\mathtt{C}_1, \mathtt{C}_2$). So eventually Algorithm 8 terminates.

Following lemma summarizes and gives properties of the decomposition algorithm. The Lemma 8.3.6 and its proof also serves the purpose of describing the Algorithm 8.

Lemma 8.3.6 (Decomposition Properties)**.** *Let $\mathtt{I}$ be the input and the Set $\mathcal{S}_{abs}$ be the output of Algorithm 8. Then,*

(1) $\mathbf{V}_{\mathbb{F}_q}(\mathtt{I}) = \bigcup_{\mathtt{C} \in \mathcal{S}_{abs}} \mathbf{V}_{\mathbb{F}_q}(\mathtt{C})$. However, $\mathtt{I} \subseteq \bigcap_{\mathtt{C} \in \mathcal{S}_{abs}} \mathtt{C}$.

(2) Each component $\mathtt{C} \in \mathcal{S}_{abs}$ returned by Algorithm 8 is absolutely irreducible.

(3) If $f = 0 \bmod \mathtt{I}$ then $f = 0 \bmod \mathtt{C}$ for all $\mathtt{C} \in \mathcal{S}_{abs}$.

(4) If $\mathbf{a} \in \mathbf{V}_{\mathbb{F}_q}(\mathtt{I})$ then $\exists \mathtt{C} \in \mathcal{S}_{abs}$ such that $\mathbf{a} \in \mathbf{V}_{\mathbb{F}_q}(\mathtt{C})$ and $\mathbf{a}$ is a non-singular root of $\mathtt{C}$.

Proof. Part (2) of the lemma easily follows by looking at Steps 9 and 10. An ideal $\mathtt{C}$ is added to $\mathcal{S}_{abs}$ only at Step-10 after being checked to be absolutely irreducible at Step-9.

Proof of Part (1): The proof goes via maintaining the following invariant at every stage of the while loop (define $\mathcal{S} := \mathcal{S}_{abs} \cup \mathcal{S}_{irr}$): $\mathtt{I} \subseteq \bigcap_{\mathtt{C} \in \mathcal{S}} \mathtt{C}$ and $\mathbf{V}(\mathtt{I}) = \bigcup_{\mathtt{C} \in \mathcal{S}} \mathbf{V}(\mathtt{C})$.

At Step-3, we have $\mathcal{S}_{abs} = \{\}$ and so $\mathcal{S} = \mathcal{S}_{irr}$. $\mathtt{I}$ is decomposed using Theorem 8.3.5 so we have: $\mathtt{I} = \bigcap_{\mathtt{C} \in \mathcal{S}}$ and $\mathbf{V}(\mathtt{I}) = \bigcup_{\mathtt{C} \in \mathcal{S}} \mathbf{V}(\mathtt{C})$. For simplicity, we write $\mathtt{I} = \mathtt{C} \cap \mathtt{D}$ where $\mathtt{C}$ is an irreducible component and $\mathtt{D}$ is intersection of all other irreducible components.

At the first iteration of the while loop: when $\mathtt{C}$ is absolutely irreducible, $\mathtt{C}$ is added to $\mathcal{S}_{abs}$ while $\mathtt{C}_1 = \mathtt{C} + \langle h^* \rangle$ and $\mathtt{C}_2 = \mathtt{C} + \langle e^* \rangle$ are added to $\mathcal{S}_{irr}$. So the intersection $\mathtt{C} \cap \mathtt{D}$ becomes $\mathtt{C} \cap \mathtt{C}_1 \cap \mathtt{C}_2 \cap \mathtt{D}$. Since, $\mathtt{C} \subseteq \mathtt{C}_1$ and $\mathtt{C} \subseteq \mathtt{C}_2$ we have $\mathtt{C} = \mathtt{C} \cap \mathtt{C}_1 \cap \mathtt{C}_2$ and so $\mathtt{C} \cap \mathtt{D} = \mathtt{C} \cap \mathtt{C}_1 \cap \mathtt{C}_2 \cap \mathtt{D}$. Thus $\mathtt{I} = \bigcap_{\mathtt{C} \in \mathcal{S}} \mathtt{C}$ and $\mathbf{V}(\mathtt{I}) = \bigcup_{\mathtt{C} \in \mathcal{S}} \mathbf{V}(\mathtt{C})$.

When $\mathtt{C}$ is relatively irreducible, $\mathtt{C}$ is removed from $\mathcal{S}_{irr}$ while $\mathtt{C}_1 = \mathtt{C} + \langle h^* \rangle$ and $\mathtt{C}_2 = \mathtt{C} + \langle e^* \rangle$ are added to $\mathcal{S}_{irr}$. So the intersection $\mathtt{C} \cap \mathtt{D}$ becomes $\mathtt{C}_1 \cap \mathtt{C}_2 \cap \mathtt{D}$. Since, $\mathtt{C} \subseteq \mathtt{C}_1 \cap \mathtt{C}_2$ we have $\mathtt{C} \cap \mathtt{D} \subseteq \mathtt{C}_1 \cap \mathtt{C}_2 \cap \mathtt{D}$. Thus $\mathtt{I} \subseteq \bigcap_{\mathtt{C} \in \mathcal{S}} \mathtt{C}$ and $\mathbf{V}(\mathtt{I}) \supseteq \bigcup_{\mathtt{C} \in \mathcal{S}} \mathbf{V}(\mathtt{C})$.

Now, $\mathbf{V}(h^*)$ captures almost all the singular points of $\mathbf{V}(\mathtt{C})$ since it is pullback polynomial of h' which is the first order derivative of h where $\mathbf{V}(h) = \mathtt{H}$ is birationally equivalent to $\mathbf{V}(\mathtt{C})$ (recall the definition of singular points). However, $\mathbf{V}(h^*)$ misses some singular points of $\mathbf{V}(\mathtt{C})$ where the map $\psi_1 : \mathtt{H} \to \mathbf{V}(\mathtt{C})$ fails. This is because ψ_1 is a rational function and at some points its denominators vanish. Since polynomial e captures those denominators, the missing points are captures in $\mathbf{V}(e^*)$. Thus $\mathbf{V}(\mathtt{C}_1) \cup \mathbf{V}(\mathtt{C}_2) = \mathbf{V}(\mathtt{C})$ (since $\mathbf{V}(\mathtt{C})$ has only singular points). Thus $\mathbf{V}(\mathtt{I}) = \bigcup_{\mathtt{C} \in \mathcal{S}} \mathbf{V}(\mathtt{C})$. Similarly the process repeats for every iteration of while loop and invariant is maintained. In the end $\mathcal{S}_{irr} = \{\}$ and so $\mathcal{S} = \mathcal{S}_{abs}$; proving Part (1).

Part (3) follows from Part (1) as if $f \in \mathtt{I}$ then $f \in \bigcap_{\mathtt{C} \in \mathcal{S}_{abs}} \mathtt{C}$ therefore $f \in \mathtt{C}$ for all $\mathtt{C} \in \mathcal{S}_{abs}$.

Proof of Part (4): a root $\mathbf{a}$ of $\mathtt{I}$ is either a singular or a non-singular root of some irreducible component $\mathtt{C}$ of $\mathtt{I}$ (at Step-3, Theorem 8.3.5). If $\mathbf{a}$ is a non-singular root of $\mathtt{C}$ then we are done as in this case $\mathtt{C}$ must be absolutely irreducible (birational to an absolutely irreducible polynomial; see Lemma 8.3.2) and added to $\mathcal{S}_{abs}$ at Step-10.

If $\mathbf{a}$ is a singular root of $\mathtt{C}$ then $\mathtt{C}$ can be absolutely irreducible or relatively irreducible. In any case, all the singular points of $\mathtt{C}$ (and hence $\mathbf{a}$) are captured, at Steps 12 and 13, either in $\mathtt{C}_1 = \mathtt{C} + \langle h^* \rangle$ or in $\mathtt{C}_2 = \mathtt{C} + \langle e^* \rangle$ (those missed by h^* as the map $\psi_1 : \mathtt{H} \to \mathbf{V}(\mathtt{C})$ is not defined

on $\mathbf{V}(e)$) as discussed before in the proof of Part (1). The dimensions of $\mathtt{C}_1$ and $\mathtt{C}_2$ are less than the dimension of $\mathtt{C}$ and so $\mathbf{a}$ is now a root of lesser dimension irreducible components (after decomposition of $\mathtt{C}_1$ and $\mathtt{C}_2$ at Step-14). Either $\mathbf{a}$ is now a non-singular root and we are done or the process repeats till the end and $\langle \mathbf{y} - \mathbf{a} \rangle$ is added to $\mathcal{S}_{abs}$ at Step-7 (as the dimension keeps decreasing). □

Lemma 8.3.6 will be helpful in proving and sketching the correctness of Algorithm 9 in next section. For example, Algorithm 9 will perform reduction by suitable lift $\hat{\mathtt{C}}$ of the ideal $\mathtt{C}$ returned by the decomposition Algorithm 8. There Lemma 8.3.6 will be useful in showing that every root of $\mathtt{I}$ lifts through some $\mathtt{C}$ (to $\hat{\mathtt{C}}$) and the shifted polynomial reduced modulo $\hat{\mathtt{C}}$ must be a multiple of p.

8.3.4 Algorithm for getting Roots into 'Absolute' Ideals

We now give our algorithm which takes a system $\mathcal{F}$ of polynomials over Galois ring $\mathbb{G}$ and returns a list of absolute ideals with strong lifting property which contains all the roots of $\mathcal{F}$ in $\mathbb{G}$. After formally presenting the algorithm, we provide the explanation and proof sketch for correctness of the algorithm throughout the section.

Input: The input consists of a system of n-variate polynomials $\{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}) \mid f_j(\mathbf{x}) \in \widehat{\mathbb{G}}[\mathbf{y}_0, \ldots \mathbf{y}_{\ell-1}][\mathbf{x}]\}$ with the required exponent k, and an ideal $\hat{\mathtt{I}} = \hat{\mathtt{I}}_{\ell-1} \subseteq \widehat{\mathbb{G}}[\mathbf{y}_0, \ldots \mathbf{y}_{\ell-1}]$; where p is prime and $\varphi(z)$ is an $\mathbb{F}_p$-irreducible polynomial. We also maintain the ideal tree $\mathcal{T}$ and keep updating it along the algorithm.

Output: The algorithm outputs a list $\mathcal{L}$ of (absolutely irreducible) ideals, collectively containing the lift of the common roots of the system $f_j(\mathbf{x}) \equiv 0 \bmod \langle p^k \rangle + \hat{\mathtt{I}}$, for $j \in [m]$.

Initialization: We initialize the ideal as $\hat{\mathtt{I}} := \langle 0 \rangle$, $\ell := 0$, and the required exponent as k. A system of polynomials $\mathcal{F} := \{F_1(\mathbf{x}), \ldots, F_m(\mathbf{x})\}$ where $F_j(\mathbf{x}) \in \widehat{\mathbb{G}}[\mathbf{x}]$. We pass $\mathcal{F}$ to the algorithm so it starts with $\mathrm{SHN}_{p^k}(F_1, \ldots, F_m, k, \langle 0 \rangle)$.

Algorithm 9 Algorithm to solve a system of polynomial equations over a Galois ring.

1: **procedure** $\mathrm{SHN}_{p^k}(f_1, \ldots, f_m, k, \hat{\mathtt{I}}, \mathcal{T})$

2: **if** Zeroset $\mathbf{V}_{\mathbb{F}_q}(\hat{\mathtt{I}} + \langle p \rangle) = \emptyset$ **then return** $\{\}$.

3: **if** $k \leq 0$ **then return** $\{\hat{\mathtt{I}}\}$.

4: $\mathtt{I} \leftarrow \mathsf{Rad}(\langle f_1(\mathbf{y}_\ell), \ldots, f_m(\mathbf{y}_\ell)\rangle + \hat{\mathtt{I}} + \langle p\rangle)$, for (new) virtual root $\mathbf{y}_\ell := (y_{\ell,1}, \ldots, y_{\ell,n})$.

5: $\mathcal{S} \leftarrow$ ABS_DECOMP($\mathtt{I}$); absolutely irreducible ideals as computed by Algorithm 8.

6: $\mathcal{L} \leftarrow \{\}$

7: **for** each $\mathtt{C} \in \mathcal{S}$ **do**

8: **if** $\mathtt{C} \cap \mathbb{F}_q[\mathbf{y}_0, \ldots, \mathbf{y}_{\ell-1}] = \mathtt{I} \cap \mathbb{F}_q[\mathbf{y}_0, \ldots, \mathbf{y}_{\ell-1}]$ **then**

9: Find the special lift $\hat{\mathtt{C}}$ of $\mathtt{C}$ to $\widehat{\mathbb{G}}$ by computing Gröbner basis and lifting, using Lemma 8.3.7. /*$\hat{\mathtt{C}}$ is *prime*; reduced Gröbner basis w.r.t. $\mathbf{y}_0 < \ldots < \mathbf{y}_{k-1}$.*/

10: Add $\hat{\mathtt{C}}$ as a child of the current node to $\mathcal{T}$ (also add other related information $\mathtt{C}, \mathtt{H}, \hat{\mathtt{H}}$ and respective birational maps from Lemma 8.3.7).

11: For $j \in [m]$, compute $\tilde{f}_j(\mathbf{x}) := p^{-1} f_j(\mathbf{y}_\ell + p\mathbf{x}) \bmod \hat{\mathtt{C}}$, over $\widehat{\mathbb{G}}[\mathbf{y}_0, \ldots \mathbf{y}_\ell][\mathbf{x}]$.

12: $\mathcal{L} \leftarrow \mathcal{L} \cup \mathrm{SHN}_{p^k}(\tilde{f}_1, \ldots, \tilde{f}_m, k-1, \hat{\mathtt{C}}, \mathcal{T})$. /*Maintain the recursion-tree $\mathcal{T}$.*/

13: **else** /*Backtrack & repeat steps*/

14: Find min $s \leq \ell - 1$ s.t. $\mathtt{C} \leftarrow \mathtt{C} \cap \mathbb{F}_q[\mathbf{y}_0, \ldots, \mathbf{y}_s] \supsetneq \mathtt{I} \cap \mathbb{F}_q[\mathbf{y}_0, \ldots, \mathbf{y}_s]$.

15: Find special lift $\hat{\mathtt{C}}$ of $\mathtt{C}$ over $\widehat{\mathbb{G}}$ using Lemma 8.3.7. /* update $\mathcal{T}$ as before*/

16: For all $j \in [m]$, compute $\tilde{f}_j(\mathbf{x}) := p^{-s-1} F_j(\mathbf{y_0} + \cdots + p^s \mathbf{y}_s + p^{s+1}\mathbf{x}) \bmod \hat{\mathtt{C}}$.

17: $\mathcal{L} \leftarrow \mathcal{L} \cup \mathrm{SHN}_{p^k}(\tilde{f}_1, \ldots, \tilde{f}_m, k+\ell-1-s, \hat{\mathtt{C}}, \mathcal{T})$. /*Maintain $\mathcal{T}$ as before.*/

18: **return** $\mathcal{L}$. /*Also, return the recursion-tree $\mathcal{T}$ whose leaves are ideals in $\mathcal{L}$.*/

Description of Algorithm 9:

The outline of the algorithm is very similar to the algorithm for univariate root counting as both these follow the method of ideals. Basically, as was the case with the split ideals, we construct ideals which at any intermediate stage partially represent the roots of the system $\mathcal{F}(\mathbf{x})$ and in the end every ideal $\hat{\mathtt{I}}$ in the returned list $\mathcal{L}$ follows the equation

$$\mathcal{F}(\mathbf{y}_0 + \cdots + p^{k-1}\mathbf{y}_{k-1}) \equiv p^\alpha \mathcal{F}' \bmod \hat{\mathtt{I}}$$

such that $\alpha \geq k$. Thus for any root $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_{k-1})$ of $\hat{\mathtt{I}}$,

$$\mathcal{F}(\hat{\mathbf{a}}_0 + \cdots + p^{k-1}\hat{\mathbf{a}}_{k-1}) \equiv 0 \bmod p^k.$$

We are forming the ideals by adding $f_j(\mathbf{y}_\ell)$ mod p into $\mathtt{I}$ and then 'lifting' the ideal to the p-adic $\hat{\mathtt{I}}$ over $\widehat{\mathbb{G}}$. Our requirement is that all the roots of $\mathtt{I}$ should lift to a root of $\hat{\mathtt{I}}$ which is our next precision absolute ideal. Half the job is done by the decomposition algorithm which breaks $\mathtt{I}$ into many absolutely irreducible ideals $\mathtt{C}$ and making sure that every root of $\mathtt{I}$ must be a 'simple root' (those which lift, Lemma 8.3.6) of some such $\mathtt{C}$. But the task still remains to take a suitable lift $\hat{\mathtt{C}}$ of $\mathtt{C}$ and guarantee the lifting of 'simple roots'. Following lemma achieves this using Gröbner basis and birational equivalence.

Lemma 8.3.7 (Connection of points via hypersurfaces [Cha22])**.** *Given an $\mathbb{F}_q$-irreducible ideal $\mathtt{C}$ (resp. its birational equivalent hypersurface $\mathtt{H}$), we can lift it to a* prime *$\widehat{\mathbb{G}}$-ideal $\hat{\mathtt{C}}$ (resp. its birational equivalent hypersurface $\hat{\mathtt{H}}$), such that their morphism diagram commutes (Figure 8.1).*

In particular, for a non-singular $\mathbb{F}_q$-root of $\mathtt{H}$ (thus a root of $\mathtt{C}$), we can find a $\widehat{\mathbb{G}}$-root of $\hat{\mathtt{H}}$; which gives a root of $\hat{\mathtt{C}}$. This sets up the 'connection' between roots of $\mathtt{C}$ and $\hat{\mathtt{C}}$.

The basic idea in the proof of Lemma 8.3.7 is: Given $\mathbb{F}_q$-irreducible ideal $\mathtt{C}$, compute the reduced Gröbner basis of $\mathtt{C}$, using the block order $\mathbf{y}_0 < \ldots < \mathbf{y}_{k-1}$; and simply see it as a p-adic ideal $\hat{\mathtt{C}}$. This is a $\widehat{\mathbb{G}}$-irreducible ideal, which is the required special lift of $\mathtt{C}$. (Its localized version $B^{-1}\hat{\mathtt{C}}$ has a *triangular* Gröbner basis; where B is a transcendence basis of variables.) The proof closely follows Figure 8.1; and proves its commutativity. We provide the proof of Lemma 8.3.7 below.

Proof. We have a prime ideal $\mathtt{C}$ given by generators in $\mathbb{F}_q[y_1, \ldots, y_N]$. Let $r > 0$ be the dimension of the variety of $\mathtt{C}$. By one of the definitions of dimension, there is a subset $B =: \{\ell_1 < \ldots < \ell_r\}$ of *least* possible variables in $\mathbf{y}$, such that the function field $\mathbb{F}_q(\mathtt{C})$ is a *finite* extension over the transcendental field $\mathbb{F}_q(B)$. So, we consider its defining maximal ideal $B^{-1}\mathtt{C}$; and compute its reduced Gröbner basis (using Buchberger's algorithm [Buc65]); with the graded lexicographical ordering $y_1 < \ldots < y_N$ and variables B *localized*. Let $B' := \mathbf{y} \setminus B =: \{\ell_{r+1} < \ldots < \ell_N\}$ be the remaining variables.

Triangular form. The localization $B^{-1}\mathtt{C}$ is a zero-dimensional prime ideal (= *maximal* ideal). Thus, $B^{-1}\mathtt{C}$ has exactly $N - r$ generators, the i-th one ($r < i \leq N$) corresponding

to a monic *minpoly* (over $\mathbb{F}_q(B)$) for the variable ℓ_i in B' (in particular, having the leading-monomial an ℓ_i-power). Thus, the Gröbner basis $\mathrm{GB}(B^{-1}\mathtt{C})$ is in a special form, that we call the *triangular form* in B' over B.

***p*-adic lift.** Compute the reduced Gröbner basis $\mathrm{GB}(\mathtt{C})$ too, and divide each generator by its leading coefficient (in $\mathbb{F}_q^*$) to make the polynomials *monic*; store them in reduced form where the coefficients are in $\{0, \ldots, p-1\}$. Define the p-adic lift $\hat{\mathtt{C}}$, of $\mathtt{C}$ to $\hat{\mathbb{G}}$, by considering the trivial integral embedding of each generator of $\mathtt{C}$. By Gröbner basis properties and the special generators, this special lift $\hat{\mathtt{C}}$ is a prime $\hat{\mathbb{G}}$-ideal.

Doing the same thing to $\mathrm{GB}(B^{-1}\mathtt{C})$, it is easy to deduce: the $\hat{\mathbb{G}}$-ideal thus obtained, called $B^{-1}\hat{\mathtt{C}}$, is a maximal ideal with a triangular (& reduced) Gröbner basis.

$\mathbb{F}_q$-map. By construction, $\mathbb{F}_q(B)[B']/\mathtt{C}$ is a field, denoted $\mathtt{R}$, of finite degree over $\mathtt{R}_0 := \mathbb{F}_q(B)$. We can compute a hypersurface $\mathtt{H}$ that is *birationally equivalent* to the variety of $\mathtt{C}$ using Theorem 8.3.5 ([HW99]). A standard algebraic way to compute it, is to pick a *random* linear form ℓ_0; assume q to be 'large' enough for random sampling. Let $h(Y)$ be the *minpoly* of the *primitive* element $\ell_0 \in \mathtt{R}$ over the subfield $\mathtt{R}_0$. We can store a representation of h in $\mathbb{F}_q[B][Y]$ such that it gives an $\mathtt{R}_0$-isomorphism ψ_1 between the fields, $\mathtt{R} = \mathtt{R}_0[B']/\mathtt{C} \cong \mathtt{S} := \mathtt{R}_0[Y]/\langle h \rangle$; mapping $\ell_0 \mapsto Y$, and other ℓ_i $(i > r)$ to its implied image.

***p*-adic map.** Take any p-adic lift $\hat{h}$ of h; clearly $\hat{h} \in \widehat{\mathbb{G}}(B)[Y]$. By definition, $\hat{h}(\ell_0) \in \mathtt{C} = \hat{\mathtt{C}} + \langle p \rangle$. Since ℓ_0 is a separable $\mathbb{F}_q$-root of $\hat{h}$, we can Hensel lift it to a $\widehat{\mathbb{G}}$-root $\ell_0' \in \widehat{\mathbb{G}}(B)[B'] =: \mathtt{R}_0'[B']$ such that $\hat{h}(\ell_0') \in \hat{\mathtt{C}}$. So, mapping $Y \mapsto \ell_0'$ gives a $\mathtt{R}_0'$-homomorphism $\hat{\psi}_2 : \mathtt{S}' := \mathtt{R}_0'[Y]/\langle \hat{h} \rangle \longrightarrow \mathtt{R}' = \mathtt{R}_0'[B']/\hat{\mathtt{C}}$; which is a map between integral domains. Moreover, it remains a nontrivial homomorphism if we localize the base ring from $\mathbb{Z}_p$ to $\mathbb{Q}_p$; making it a map between *fields*. Thus, $\hat{\psi}_2$ is an injective $\mathtt{R}_0'$-homomorphism.

Now we know: all the four rings in Figure 8.1 are domains (& two are fields). So, in case $\hat{\psi}_2$ is not an isomorphism, it is injective and non-surjective. Let $v_0 \in \mathtt{R}'$ be an element that is out of the image, but we know that some lift $v_0 + pv_1$ is in the image of $\hat{\psi}_2$ (by traversing the commutative diagram). Similarly, we have that some lift $v_1 + pv_2$, of v_1, is in the image of $\hat{\psi}_2$. Combining these two, we know: $v_0 - p^2 v_2$ is in the image of $\hat{\psi}_2$. Doing this *ad infinitum*, we get v_0 in the image of $\hat{\psi}_2$; contradicting its choice. We conclude: $\hat{\psi}_2$ is an isomorphism,

with the inverse map being (say) $\hat{\psi}_1$.

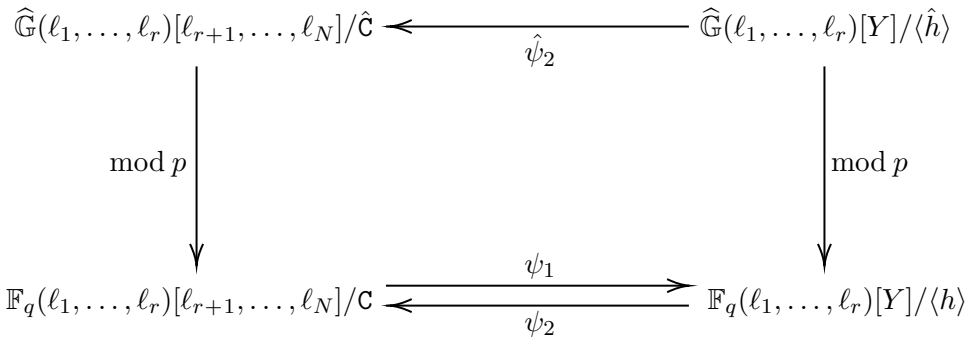


Figure 8.1: Commutative Diagram

In the above diagram let us start with a non-singular $\mathbb{F}_q$-root $\mathbf{a}$ of $\mathtt{H} := \mathbf{V}(h)$. With high probability, it will keep the relevant polynomials in $\ell_1,\ldots,\ell_r$ nonzero mod p; thus it would be consistent with the localization. It has 'pullback' via ψ_1, giving a root of $\mathtt{C}$. By the separability of the $\mathbb{F}_q$-root, $\mathbf{a}$ lifts to a root $\hat{\mathbf{a}}$ of $\hat{\mathtt{H}} := \mathbf{V}(\hat{h})$ (Lemma 8.3.4); from up there it has 'pullback' via $\hat{\psi}_1$, giving a $\widehat{\mathbb{G}}$-root of $\hat{\mathtt{C}}$ too. This connects $\mathbf{V}(\mathtt{C})$ with $\mathbf{V}(\hat{\mathtt{C}})$. □

Lemma 8.3.7 tackles the issue of taking a suitable lift $\hat{\mathtt{C}}$ to guarantee that all the simple roots of $\mathtt{C}$ has a lift in $\hat{\mathtt{C}}$ as well as an efficient way to find a lift via birational maps. But there are more issues to tackle, during the process of decomposition, which arise due to the loss of information when we go modulo p. There is no point in taking suitable lift of the ideal $\mathtt{C}$ to $\hat{\mathtt{C}}$, using Lemma 8.3.7, if $\hat{\mathtt{C}}$ does not have the reminiscences of the previous ideal through which the parent ideal $\hat{\mathtt{I}}$ originated. This is the reason we do the test at Step-8 of $\mathtt{C}$ vs $\mathtt{I}$ till the variable $\mathbf{y}_{\ell-1}$ (one less precision than $\mathtt{C}$ and $\mathtt{I}$). $\mathbf{y}_\ell$ lift does not matter because it will be a new lift so the reduced polynomial (Step-11) will adjust accordingly with the new lift (also think of ideals as triangular with variable ordering $\mathbf{y}_0 < \cdots < \mathbf{y}_{\ell-1}$)

We handle this issue using the idea of *backtracking*. Roughly, when we take suitable lift of $\mathtt{C}$ in the proof of Lemma 8.3.7 we look the Gröbner basis in form of triangular batches– the lowest in variables $\mathbf{y}_0$, the next one in $\mathbf{y}_0, \mathbf{y}_1$ and so on. Also these generators are monic. So we basically check at what level the basis change (corruption) has happened via the comparison $\mathtt{C} \cap \mathbb{F}_q[\mathbf{y}_0,\ldots,\mathbf{y}_{\ell-1}] = \mathtt{I} \cap \mathbb{F}_q[\mathbf{y}_0,\ldots,\mathbf{y}_{\ell-1}]$. Say the corruption has happened at level $\mathbf{y}_s$. If $s = \ell - 1$ then there is no corruption and we move on to take suitable lift of $\mathtt{C}$. Otherwise we

have to discard all the generators involving $\mathbf{y}_{s+1}, \ldots, \mathbf{y}_{\ell-1}$.

Invariant. In the recursion tree $\mathcal{T}$ either a node is created when moving from $\hat{\mathtt{I}}_{\ell-1}$ to $\hat{\mathtt{I}}_\ell$ at Step-12 and k decreases or backtracking happens at Step-17 and $\hat{\mathtt{I}}_s$ is redefined for $s < \ell$ but the $\dim(\mathbf{V}(\hat{\mathtt{I}}_s))$ reduces. Thus redefinition of $\hat{\mathtt{I}}_s$ happens at most n times in a path. This bounds the maximum path length $\leq k + kn$ in the tree.

Now we will see that the set of ideals returned by the algorithm contains all and only the roots of the given system of polynomials in form of their *non-singular* roots.

Following Proposition 4 is the easy direction which shows that for any root $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_{k-1})$ of any $\hat{\mathtt{I}} \in \mathcal{L}$, $\mathcal{F}(\hat{\mathbf{a}}_0 + \cdots + p^{k-1}\hat{\mathbf{a}}_{k-1}) \equiv 0 \bmod p^k$. Basically, as said before the objective of the Algorithm 9 is to produce ideals $\hat{\mathtt{I}}$ such that $\mathcal{F}(\mathbf{y}_0 + \cdots + p^{k-1}\mathbf{y}_{k-1}) \equiv p^\alpha \mathcal{F}' \bmod \hat{\mathtt{I}}$ for $\alpha \geq k$. Thus modulo p^k, any zero of $\hat{\mathtt{I}}$ gives a zero of $\mathcal{F}$. This is the idea; the proof details are given in [Cha22].

Proposition 4 (Root in $\mathcal{L} \longrightarrow$ Root of $\mathcal{F}$ [Cha22])**.** *Given a root of a leaf in $\mathcal{L}$ (using $\mathcal{T}$ and Lemma 8.3.9), we can find a common $\mathbb{G}$-root of the system $\mathcal{F}$ of polynomials f_j, for $j \in [m]$.*

The converse Proposition 5 is little bit involved and we prove it here. We show that every $\mathbb{G}$-root of the system $\mathcal{F}$ has its p-adic lift present in some ideal of $\mathcal{L}$.

Proposition 5 (Root of $\mathcal{F} \longrightarrow$ Root in $\mathcal{L}$)**.** *If the system of polynomials $\mathcal{F}$ has a root $(\mathbf{a}_0 + p\mathbf{a}_1 + \cdots + p^{k-1}\mathbf{a}_{k-1})$ in $\mathbb{G}$, then there is a leaf ideal $\hat{\mathtt{I}}_{k-1} \in \mathcal{L}$ which has a root $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_{k-1})$ such that $\mathbf{a}_0 + p\mathbf{a}_1 + \cdots + p^{k-1}\mathbf{a}_{k-1} \equiv \hat{\mathbf{a}}_0 + p\hat{\mathbf{a}}_1 + \cdots + p^{k-1}\hat{\mathbf{a}}_{k-1} \bmod p^k$.*

Proof. Let us assume that the system of polynomials has a $\mathbb{G}$-root, given by $\mathbf{a} := (\mathbf{a}_0 + p\mathbf{a}_1 + \cdots + p^{k-1}\mathbf{a}_{k-1})$, with $\mathbf{a}_i$'s effectively 'in' $\mathbb{F}_q$. We use a technique, similar to that in the proof of Proposition 4, to inductively show that a root up to precision ℓ digits gives a p-adic root of the ideal grown for ℓ-steps (possibly with backtrackings). We use the same notation as of Proposition 4 for $f_j^{(\ell)}, \hat{\mathtt{I}}_\ell$.

For the *base case* of induction, let us consider the root $\mathbf{a}_0$ of $f_1(\mathbf{y}_0), \ldots, f_m(\mathbf{y}_0)$ over $\mathbb{F}_q$. Now, each of these equations were added to the ideal $\hat{\mathtt{I}}_0$, on which we performed the decomposition algorithm to find components $\hat{\mathtt{C}}$'s. Since, $\mathbf{a}_0$ is a root of $\hat{\mathtt{I}}_0 + \langle p \rangle$, it must also be a root of some $\hat{\mathtt{C}} + \langle p \rangle$; let us fix this ideal $\hat{\mathtt{C}}$. Now, by definition (Algorithm 8 & Lemma

8.3.7), $\hat{\mathtt{C}}$ is prime; and absolutely irreducible mod$p$. If $\mathbf{a}_0$ is a non-singular $\mathbb{F}_q$-root of $\hat{\mathtt{H}}$ (= hypersurface birationally equivalent to $\hat{\mathtt{C}}$), then it has a lift, say $\hat{\mathbf{a}}_0$, using Hensel's lifting (Lemma 8.3.4). Thus, we get a corresponding $\widehat{\mathbb{G}}$-root of $\hat{\mathtt{C}} \in \mathcal{T}$. On the other hand, if $\mathbf{a}_0$ is a singular root, or a root whose preimage does not exist in the hypersurface, then it will be present in some ideal of *lesser* dimension (eg. in another branch of recursion-tree $\mathcal{T}$). So, Algorithm 8 will locate $\mathbf{a}_0$ as a non-singular root of some other absolutely irreducible ideal of dimension ≥ 0. Thus, we always get a corresponding $\widehat{\mathbb{G}}$-root of some ideal, say $\hat{\mathtt{D}}$, in $\mathcal{T}$.

Now, for our *induction hypothesis*, assume that the root of the system modulo p^ℓ, $(\mathbf{a}_0 + \cdots + p^{\ell-1}\mathbf{a}_{\ell-1})$, gives a p-adic root, $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_{\ell-1})$, of some $\hat{\mathtt{C}}$ which is an absolutely irreducible component of $\hat{\mathtt{I}}_{\ell-1}$ such that

$$\sum_{i=0}^{t-1} p^i \mathbf{a}_i \;\equiv\; \sum_{i=0}^{t-1} p^i \hat{\mathbf{a}}_i \bmod p^t, \text{ for } t \leq \ell\,. \tag{8.3}$$

Let us consider the *induction step*. Consider $f_j^{(\ell)}(\mathbf{y}_\ell) = f_j(\mathbf{y}_0 + \ldots + p^{\ell-1}\mathbf{y}_{\ell-1} + p^\ell \mathbf{y}_\ell) \bmod \hat{\mathtt{C}}$, where $\hat{\mathtt{C}}$ is the component where the p-adic root $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_{\ell-1})$ can be found. After substituting the first ℓ variables by $(\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_{\ell-1})$, $f_j^{(\ell)}(\mathbf{y}_\ell)$ has a root, say $\hat{\mathbf{a}}_\ell$, modulo $p^{\ell+1}$; simply because— $f_j(\mathbf{a}_0 + \ldots + p^{\ell-1}\mathbf{a}_{\ell-1} + p^\ell \mathbf{x})$ has the root $\mathbf{a}_\ell$ modulo $p^{\ell+1}$, and by the induction hypothesis (esp. Equation 8.3). Like we did in the base case, we can consider two broad cases: $\hat{\mathbf{a}}_\ell$ is a non-singular $\mathbb{F}_q$-root, or it is a singular root (or a root whose preimage does not exist in the birationally equivalent hypersurface of Lemma 8.3.7). In the first case, we find a suitably lifted root in $\hat{\mathtt{C}} \in \mathcal{T}$ itself. While in the second case, we find a suitably lifted root in some lower-dimensional $\hat{\mathtt{D}} \in \mathcal{T}$ (though with the same $\{\hat{\mathtt{I}}_0, \ldots, \hat{\mathtt{I}}_{\ell-1}\}$). Thus, in all cases we ensure that

$$\hat{\mathbf{a}}_0 + \ldots + p^{\ell-1}\hat{\mathbf{a}}_{\ell-1} + p^\ell \hat{\mathbf{a}}_\ell \;\equiv\; (\mathbf{a}_0 + \ldots + p^\ell \mathbf{a}_\ell) \bmod p^{\ell+1},$$

for the lifted $\hat{\mathbb{G}}$-root of some ideal $\hat{\mathtt{I}}_\ell$ (which gets defined in Step 9 of Algorithm 9, possibly after many backtrackings); finishing the induction step.

Thus, with $\ell = k-1$, we deduce: $\mathbf{a}$ is represented as a $\widehat{\mathbb{G}}$-root of some ideal $\hat{\mathtt{I}}_{k-1}$ in $\mathcal{L}$. $\square$

8.3.5 Correctness and Root Finding

A major component of the time complexity of the algorithm is the size of the virtual tree created which is doubly exponential in nk as shown in the following lemma. Thus it subsumes all other times taken at other steps of the algorithm as well as time taken to fetch a root from the list of ideals returned by the algorithm. Also it is clear then that if $n+k$ is constant then the algorithm is efficient.

Lemma 8.3.8 (Size of tree)**.** *The total number of leaves $\mathcal{L}$ of the recursion-tree $\mathcal{T}$ is at most $d^{(nk)^{O((nk)^2)}}$. Indeed, the size of $\mathcal{T}$ and the degree of any ideal in $\mathcal{T}$ are bounded by $d^{(nk)^{O((nk)^2)}}$.*

Proof. We build tree $\mathcal{T}$ by first passing an ideal $\mathtt{I}_0$, in n variables, in Algorithm 8 with generators of degree at most $d_0 := d$. The set of absolutely irreducible ideals returned by Algorithm 8 forms the branches in the first level of $\mathcal{T}$. Each of these ideals (branches) at level-1, say $\mathtt{I}_1$, of degree d_1 (now in $2n$ variables) recurse in Algorithm 8, and produce more branches (ideals) at level-2. This process continues till $(k-1)$-th level.

The analysis of producing branches from an ideal at level $(\ell-1)$ to level ℓ is the same as that of [HW99]. This will allow us to use their estimates for number of branches and degree of new generators produced [HW99, Lem.2.7].

Similar to [HW99], we first decompose ideal $\mathtt{I}_{\ell-1}$ in $n\ell$ variables at Step 3 (Algorithm 8). However, we add h^* and e^* in the ideal at Steps 12-13 and then iterate. The idea of Step 12 is same as in [HW99] to capture the *singular* points of $\mathbf{V}(\mathtt{I})$ in separate absolutely irreducible ideals by adding h^* to the ideal. In [HW99], it was shown that the dimension of variety reduces when we add h^*.

When we add e^*, we make the 'free' variables $\ell_1, \ldots, \ell_r$ in the hypersurface (in Lemma 8.3.7) to satisfy an equation $e(\ell_1, \ldots, \ell_r) = 0$. Therefore, the transcendence degree reduces by 1, and it can reduce at most dimension-many times. So, complexity wise Step 13 is subsumed in Step 12 as degree of h^* and e^* have similar bound [HW99, Lem.2.7, Thm.2.6].

Applying analysis of [HW99] on ideal $\mathtt{I}_0$ at level-0, the number of branches (ideals) produced are $d_0^{n^{O(n)}}$ and the degree of generators at most $d_0^{n^{O(n)}} =: d_1$ at level-1. Each such branch (ideal) further produces (at level-2) $d_1^{(2n)^{O(2n)}}$ new branches with degree at most $d_1^{(2n)^{O(2n)}}$. By

induction, the generator-set size, and degree, at level-nk (i.e. the leaves) is $\leq d^{(nk)^{O((nk)^2)}}$. □

Extracting a Root: Algorithm 9 returns the set of ideals $\mathcal{L}$ (the leaves of tree $\mathcal{T}$). If $\mathcal{L}$ is not empty then the ideals are either points (only one root) or other absolutely irreducible ideals modulo p. We are required to extract roots from these ideals to get a solution of $\mathcal{F}$ modulo p^k. The task is trivial for point ideals. Otherwise we use the extra information contained in $\mathcal{T}$ i.e, for an ideal $\hat{\mathtt{I}}$ and corresponding $\mathtt{I}$ we utilize the corresponding hypersurfaces $\hat{\mathtt{H}}$ and $\mathtt{H}$ and the respective birational maps. Pick a random point of $\mathtt{H}$ which should be a simple point with high probability ($\mathtt{H}$ is absolutely irreducible) and lift it to a point of $\hat{\mathtt{H}}$. Then, using the inverse birational map we get the corresponding point of $\hat{\mathtt{I}}$ which gives a zero of $\mathcal{F}$ mod p^k This is the rough idea, the ideas with more details are summarized in the following lemma from [Cha22].

Lemma 8.3.9 (Extracting roots [Cha22])**.** *Given $\widehat{\mathbb{G}}$-ideal $\hat{\mathcal{I}}_{k-1}$ in a leaf of the tree $\mathcal{T}$, we can find a generic common $\widehat{\mathbb{G}}$-root (if one exists) of the preceding ideals $\{\hat{\mathcal{I}}_\ell \,|\, \ell\}$.*

Now we have all the ingredients to prove our main theorem. Recall, the theorem statement implies that when $n+k$ is constant, our method to solve the system of polynomial equations is efficient.

Proof of Theorem 8.1.3. As proved in Propositions 5 & 4, Algorithm 9 (using Algorithms 8 and Lemma 8.3.9) correctly returns a root (via an absolutely irreducible ideal), if and only if one exists.

Tree $\mathcal{T}$ built by Algorithm 9 has size $D := d^{(nk)^{O((nk)^2)}}$ and each of the ideal in $\mathcal{T}$ has at most nk variables with degree at most D (Lemma 8.3.8). At each step, we perform arithmetic with the reduced Gröbner basis of the ideal, which has polynomials of degree $\leq D$ and $\leq nk$ variables, and requires $\mathsf{poly}(D)$-time $\widehat{\mathbb{G}}$ arithmetic [Dub90]. After these arithmetic operations are performed, we check for an $\mathbb{F}_q$-root of the ideals using [HW99], which takes randomized $\mathsf{poly}(m, D^{(nk)^{O(nk)}}, \log q)$ time. Thus, the net time complexity is randomized $\mathsf{poly}(m, d^{c_{nk}}, \log p^b)$, where $c_{nk} \leq (nk)^{O((nk)^2)}$ and $q = p^b$ with $b = \deg(\varphi)$.

However, the algorithm uses [HW99] as a blackbox, which requires the additional condition that $q = p^b > d^{c_{nk}}$. If this condition is not satisfied, i.e. q is *small*, then we can determin-

istically find a root using exhaustive search of $q^{nk} \leq d^{c_{nk} \cdot nk}$ many iterations. This case has the time complexity as deterministic $\mathsf{poly}(m, d^{c'_{nk}}, \log p^b)$, where $c'_{nk} \leq c_{nk} \cdot nk \leq (nk)^{O((nk)^2)}$. This proves Theorem 8.1.3 in all cases. □

8.4 Summary

In this chapter, we dealt with the problem of finding a common root of a system of polynomial equations over Galois rings. We extend the results of [HW99] to find roots of a system of equations from Galois fields to Galois rings. This is achieved using our framework– the method of ideals. Essentially, we gave efficient construction of a set of 'absolute' ideals encoding all the roots of given system and utilize the "algebraic niceness" of these ideals to randomly retrieve a root of the system. The outline is similar to as was with 'split' ideals for univariate root counting. A major difference with the split ideals is that absolute ideals, although collectively contain all the zeros of the system, they do not give exact count on the number of zeros of the system but only provide an upper bound.

We also make progress towards finding factors of univariate polynomials in prime-power rings. The problem has been of interest since the time of Hensel [Hen18] who gave a method to lift (coprime) factors to modulo any prime-powers. It is easier to factorize in fields, as seen before, but factorization modulo small prime powers has been elusive to computer scientists; owing to the fact that these rings are not integral domains and there can be exponentially many factors. This difficulty has been explained in [CL01, Sir17, vzGH96, vzGH98]. Giving first general progress towards *small* k, we generalize the algorithm in Chapter 7 ([DMS21]) to find factors of *small* ramification-degree modulo p^k.

Chapter 9

Conclusion and Open Problems

In this thesis, we studied some fundamental computational problems related to finding or counting roots and factors of polynomials over integers modulo prime powers or more generally Galois rings. These problems are:

- Efficiently factoring a univariate polynomial,
- Efficiently counting roots of a univariate polynomial and computing p-adic Igusa's local zeta function which encodes the root-count and,
- Efficiently solving a system of 'constant'-variate polynomial system.

All these problems are solved, completely or partially, by reducing them to root finding/counting of a different set of polynomials. A common hurdle in all these problems is that at intermediate stages the access to roots are prohibited. This is either due to the requirement of deterministic algorithm or the roots are too many to be handled efficiently. Moreover, known tools like Hensel lifting help only in special cases. For this purpose, we develop a common framework of polynomial ideals where ideals 'compactly' represent all the roots, in an efficient way, without giving individual access to them. Later, using special algebraic-geometric properties of these ideals we retrieve the required information about the roots.

Polynomial Factoring

We look for an efficient 'randomized' algorithm to factor a univariate integral polynomial $f(x)$ modulo a prime power p^k. The problem was solved for 'large' k via extended Hensel lemma but was open for 'constant' k, in particular for $k = 3$. We gave a unified method to factor f in random polynomial-time for $k \leq 4$. The problem is still open for $k \geq 5$. However, if degree of f is assumed to be constant then we achieve the original objective of giving a random polynomial-time algorithm for constant k. In fact, our result assumes the degree of the desired factor to be constant while f can have any degree. We leave it open to factor (and to test irreducibility of) $f \bmod p^k$ for $k = 5$, and beyond, in randomized polynomial-time.

Root Counting and Igusa Zeta Function

We presented the first efficient deterministic algorithm to count the number of basic-irreducible factors modulo a prime-power. Restricting it to degree-one irreducibles, we get a deterministic polynomial-time algorithm to count the roots. This is achieved by storing and refining the precision of the roots virtually using split ideals.

Many interesting questions still remain to be tackled. For p-adic fields, there is only a randomized method to count the number of irreducible factors. Analogously, the question of counting irreducible factors modulo a prime-power also remains open; no efficient method is known even in the randomized setting. The ramified roots seem to elude practical methods. On the other hand, the problem of actually finding an irreducible factor (respectively a root) deterministically, seems much harder; it subsumes the analogous classic problem in prime characteristic.

We presented the first complete solution to the problem of computing Igusa's local zeta function for any given integral univariate polynomial and a prime p. We also found an explicit closed-form expression for $N_k(f)$ (root count of $f \bmod p^k$) and efficiently computed the explicit parameters involved therein. As a corollary, we get a deterministic algorithm to count roots (with multiplicity) in p-adic rings (respectively formal power-series ring). The following important open questions need to be addressed:

(1) A natural question to study is whether we could generalize our method to compute

Igusa's local zeta function for n-variate integral polynomials (say, $n = 2$?). Note that for growing n this problem is at least #P-hard.

(2) A related problem is of counting roots of n-variate polynomials modulo prime power p^k. We know an efficient quantum algorithm modulo p for $n = 2$ due to Kedlaya. Kedlaya further asks, if we can reduce the problem of counting points mod p^k to counting points mod p for fixed k and $n = 2$.

Solving a System of Multivariate Polynomial Equations

Finding a common zero of a system of multivariate polynomials over a Galois field is a fundamental problem and has efficient randomized algorithm known, due to Huang and Wong, when number of variables n is constant. For unbounded number of variables the decision version of problem is NP-hard. We studied the more general problem of finding a common solution over a Galois ring (of characteristic p^k) and gave the first randomized polynomial time algorithm for $n + k$ constant. This leaves the following question open.

Solve a system of n-variate integral polynomials modulo p^k, for fixed n but arbitrary k (respectively over the p-adic integers $\mathbb{Z}_p$) in randomized polynomial time.

Bibliography

[AB01] Noga Alon and Richard Beigel. Lower bounds for approximations by low degree polynomials over $\mathbb{Z}_m$. In *Proceedings 16th Annual IEEE Conference on Computational Complexity*, pages 184–187. IEEE, 2001. 2

[AB03] Manindra Agrawal and Somenath Biswas. Primality and identity testing via chinese remaindering. *Journal of the ACM (JACM)*, 50(4):429–443, 2003. 2

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of mathematics*, pages 781–793, 2004. 2

[BBR94] David A Barrington, Richard Beigel, and Steven Rudich. Representing boolean functions as polynomials modulo composite numbers. *Computational Complexity*, 4(4):367–382, 1994. 2

[BDT21] Charles Bouillaguet, Claire Delaplace, and Monika Trimoska. A simple deterministic algorithm for systems of quadratic polynomials over $\mathbb{F}_2$. *Cryptology ePrint Archive*, 2021. 6

[Ber67] Elwyn R Berlekamp. Factoring polynomials over finite fields. *Bell System Technical Journal*, 46(8):1853–1859, 1967. 3, 19

[BFSS13] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Pierre-Jean Spaenlehauer. On the complexity of solving quadratic boolean systems. *Journal of Complexity*, 29(1):53–75, 2013. 6

[BGL06] Nayantara Bhatnagar, Parikshit Gopalan, and Richard J Lipton. Symmetric polynomials over $\mathbb{Z}_m$ and simultaneous communication protocols. *Journal of Computer and System Sciences*, 72(2):252–285, 2006. 2

[Bha97] Manjul Bhargava. P-orderings and polynomial functions on arbitrary subsets of dedekind rings. *Journal fur die Reine und Angewandte Mathematik*, 490:101–128, 1997. 5

[BKW19] Andreas Björklund, Petteri Kaski, and Ryan Williams. Solving systems of polynomial equations over GF(2) by a parity-counting self-reduction. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019. 6

[BLQ13] Jérémy Berthomieu, Grégoire Lecerf, and Guillaume Quintin. Polynomial root finding over local rings and application to error correcting codes. *Applicable*

Algebra in Engineering, Communication and Computing, 24(6):413–443, 2013. https://link.springer.com/article/10.1007/s00200-013-0200-5. 5, 6, 11, 14, 15, 22, 24, 25, 29, 63, 97, 102

[Bro87] W Dale Brownawell. Bounds for the degrees in the Nullstellensatz. *Annals of Mathematics*, 126(3):577–591, 1987. 6

[BS86] Zenon Ivanovich Borevich and Igor Rostislavovich Shafarevich. *Number theory*, volume 20 of *Pure and Applied Mathematics*. Academic Press, New York NY, 1986. 4

[Buc65] Bruno Buchberger. Ein algorithmus zum auffinden der basiselemente des restklassenringes nach einem nulldimensionalen polynomideal. *PhD thesis, Universität Innsbruck*, 1965. 118, 125

[CDS22] Sayak Chakrabarti, Ashish Dwivedi, and Nitin Saxena. Solving polynomial systems over non-fields and applications to modular polynomial factoring. *Submitted*, 2022. xv, 115

[CG00] David G Cantor and Daniel M Gordon. Factoring polynomials over p-adic fields. In *International Algorithmic Number Theory Symposium*, pages 185–208. Springer, 2000. 4, 5, 108

[CGRW18] Qi Cheng, Shuhong Gao, J Maurice Rojas, and Daqing Wan. Counting roots of polynomials over prime power rings. In *Thirteenth Algorithmic Number Theory Symposium, ANTS-XIII*. Mathematical Sciences Publishers, 2018. arXiv:1711.01355. 5, 12

[Cha22] Sayak Chakrabarti. Multivariate polynomials modulo prime powers: their roots, zeta-function and applications. Master's thesis, Indian Institute of Technology Kanpur, 2022. https://www.cse.iitk.ac.in/users/nitin/theses/chakrabarti-2022.pdf. xv, 115, 125, 128, 131

[Chi87] AL Chistov. Efficient factorization of polynomials over local fields. *Dokl. Akad. Nauk SSSR*, 293(5):1073–1077, 1987. 4, 5

[Chi94] AL Chistov. Algorithm of polynomial complexity for factoring polynomials over local fields. *Journal of mathematical sciences*, 70(4):1912–1933, 1994. 4, 5

[CL01] Howard Cheng and George Labahn. Computing All Factorizations in $\mathbb{Z}_N[x]$. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC'01*, pages 64–71, 2001. 4, 14, 132

[CLO13] David Cox, John Little, and Donal OShea. *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer Science & Business Media, 2013. 6, 30, 118

[Con03] J Brian Conrey. The riemann hypothesis. *Notices of the AMS*, 50(3):341–353, 2003. 8

[CP56] M Chojnacka-Pniewska. Sur les congruences aux racines données. In *Annales Polonici Mathematici*, volume 3, pages 9–12. Instytut Matematyczny Polskiej Akademii Nauk, 1956. 5

[CS23] Sayak Chakrabarti and Nitin Saxena. An effective description of the roots of multivariates mod p^k and the related Igusa's local zeta function. *Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC*, 2023. https://www.cse.iitk.ac.in/users/nitin/papers/IZF-issac23.pdf. 5, 9

[CZ81] David G Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, pages 587–592, 1981. 3, 19

[Del74] Pierre Deligne. La conjecture de Weil. I. *Publications Mathématiques de l'Institut des Hautes Études Scientifiques*, 43(1):273–307, 1974. 8

[DH01] Jan Denef and Kathleen Hoornaert. Newton polyhedra and Igusa's local zeta function. *Journal of number Theory*, 89(1):31–64, 2001. 5

[Din21a] Itai Dinur. Cryptanalytic applications of the polynomial method for solving multivariate equation systems over GF(2). In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I*, pages 374–403, 2021. 6

[Din21b] Itai Dinur. Improved algorithms for solving polynomial systems over GF(2) by multiple parity-counting. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2550–2564. SIAM, 2021. 6

[DM97] Bruce Dearden and Jerry Metzger. Roots of polynomials modulo prime powers. *European Journal of Combinatorics*, 18(6):601–606, 1997. 5

[DMS19] Ashish Dwivedi, Rajat Mittal, and Nitin Saxena. Counting Basic-Irreducible Factors Mod p^k in Deterministic Poly-Time and p-Adic Applications. In Amir Shpilka, editor, *34th Computational Complexity Conference (CCC 2019)*, volume 137 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:29, 2019. xv, 85

[DMS21] Ashish Dwivedi, Rajat Mittal, and Nitin Saxena. Efficiently factoring polynomials modulo p^4. *Journal of Symbolic Computation*, 104:805 – 823, 2021. Preliminary version appeared in The 44th ACM International Symposium on Symbolic and Algebraic Computation (ISSAC) 2019. xv, 79, 132

[DS20] Ashish Dwivedi and Nitin Saxena. Computing Igusa's local zeta function of univariates in deterministic polynomial-time. 14*th Algorithmic Number Theory Symposium (ANTS XIV), Open Book Series*, 4(1):197–214, 2020. xv, 5, 73

[Dub90] Thomas W Dubé. The structure of polynomial ideals and gröbner bases. *SIAM Journal on Computing*, 19(4):750–773, 1990. 131

[Dwi17] Ashish Dwivedi. On the complexity of Hilbert's Nullstellensatz over positive characteristic. Master's thesis, Indian Institute of Technology Kanpur, 2017. 6

[EK90] Andrzej Ehrenfeucht and Marek Karpinski. *The computational complexity of (xor, and)-counting problems*. International Computer Science Inst., 1990. 6, 9

[FS15] Michael A Forbes and Amir Shpilka. Complexity theory column 88: Challenges in polynomial factorization. *ACM SIGACT News*, 46(4):32–49, 2015. 3

[GGL08] Parikshit Gopalan, Venkatesan Guruswami, and Richard J Lipton. Algorithms for modular counting of roots of multivariate polynomials. *Algorithmica*, 50(4):479–496, 2008. 6

[GNP12] Jordi Guàrdia, Enric Nart, and Sebastian Pauli. Single-factor lifting and factorization of polynomials over local fields. *J. Symb. Comput.*, 47(11):1318–1346, November 2012. 4

[Gop06] Parikshit Gopalan. *Computing with polynomials over composites*. PhD thesis, 2006. 2

[Gop14] Parikshit Gopalan. Constructing Ramsey graphs from boolean function representations. *Combinatorica*, 34(2):173–206, 2014. 2

[Gre66] Marvin J Greenberg. Rational points in henselian discrete valuation rings. *Publications Mathématiques de l'IHÉS*, 31:59–64, 1966. 85

[Gro64] Alexander Grothendieck. Formule de Lefschetz et rationalité des fonctions L. *Séminaire Bourbaki*, 9:41–55, 1964. 8

[Gro00] Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit ramsey graphs. *Combinatorica*, 20(1):71–86, 2000. 2

[GS20] Abhibhav Garg and Nitin Saxena. Special-case algorithms for blackbox radical membership, Nullstellensatz and transcendence degree. In *Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation*, pages 186–193, 2020. 6

[Hen18] Kurt Hensel. Eine neue theorie der algebraischen zahlen. *Mathematische Zeitschrift*, 2(3):433–452, Sep 1918. 3, 19, 132

[HKC+94] A Roger Hammons, P Vijay Kumar, A Robert Calderbank, Neil JA Sloane, and Patrick Solé. The $\mathbb{Z}_4$-linearity of kerdock, preparata, goethals, and related codes. *IEEE Transactions on Information Theory*, 40(2):301–319, 1994. 6

[HW99] M-D Huang and Y-C Wong. Solvability of systems of polynomial congruences modulo a large prime. *computational complexity*, 8(3):227–257, 1999. Preliminary version appeared in The IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS) 1996. 6, 13, 117, 118, 120, 121, 126, 130, 131, 132

[Igu74] Jun-Ichi Igusa. Complex powers and asymptotic expansions. I. Functions of certain types. *Journal für die reine und angewandte Mathematik*, 268:110–130, 1974. 8

[Igu75] Jun-Ichi Igusa. Complex powers and asymptotic expansions. II. *Journal für die reine und angewandte Mathematik*, 278:307–321, 1975. 8

[Igu00] Jun-Ichi Igusa. *An introduction to the theory of local zeta functions*, volume 14 of *AMS/IP Studies in Advanced Mathematics*. American Mathematical Society, Providence, RI; International Press, Cambridge, MA, 2000. 9

[IR78] Jun-Ichi Igusa and S Raghavan. *Lectures on forms of higher degree*, volume 59. Springer Berlin-Heidelberg-New York, 1978. 8

[Kal92] Erich Kaltofen. Polynomial factorization 1987–1991. In *Latin American Symposium on Theoretical Informatics*, pages 294–313. Springer, 1992. 3

[Kay05] Neeraj Kayal. Solvability of a system of bivariate polynomial equations over a finite field. In *International Colloquium on Automata, Languages, and Programming*, pages 551–562. Springer, 2005. 6

[Kli97] Adam Klivans. Factoring polynomials modulo composites. Technical report, Carnegie-Mellon Univ, Pittsburgh PA, Dept of CS, 1997. 14

[Kob77] Neal Koblitz. P-adic numbers. In *p-adic Numbers, p-adic Analysis, and Zeta-Functions*, pages 1–20. Springer, 1977. 18

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar Signature Schemes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 206–222, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. 6

[KRRZ20] Leann Kopp, Natalie Randall, J Maurice Rojas, and Yuyu Zhu. Randomized polynomial-time root counting in prime power rings. *Mathematics of Computation*, 89(321):373–385, 2020. 14

[KU11] Kiran S Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM Journal on Computing*, 40(6):1767–1802, 2011. 3, 19

[Lau04] Alan GB Lauder. Counting solutions to equations in many variables over finite fields. *Foundations of Computational Mathematics*, 4(3):221–267, 2004. 5

[LN94] Rudolf Lidl and Harald Niederreiter. *Introduction to finite fields and their applications*. Cambridge university press, 1994. 3, 21, 76

[LPT+17] Daniel Lokshtanov, Ramamohan Paturi, Suguru Tamaki, Ryan Williams, and Huacheng Yu. Beating brute force for systems of polynomial equations over finite fields. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2190–2202. SIAM, 2017. 6

[Mau01] Davesh Maulik. Root sets of polynomials modulo prime powers. *Journal of Combinatorial Theory, Series A*, 93(1):125–140, 2001. 5

[McD74] Bernard R McDonald. Finite Rings with Identity. *Monographs and Textbooks in Pure and Applied Mathematics. Marcel Dekker, Inc*, 28:409–424, 1974. 21

[NZM13] Ivan Niven, Herbert S Zuckerman, and Hugh L Montgomery. *An Introduction to The Theory of Numbers*. John Wiley & Sons INC., U.K., 2013. 3

[Pan95] Peter N Panayi. *Computation of Leopoldt's P-adic regulator*. PhD thesis, University of East Anglia, 1995. 22, 24, 25, 97, 102

[Pat96] Jacques Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, pages 33–48, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg. 6

[Rie59] Bernhard Riemann. Über die Anzahl der Primzahlen unter einer gegebenen Grosse. *Ges. Math. Werke und Wissenschaftlicher Nachlaß*, 2:145–155, 1859. 8

[RRZ21] Caleb Robelle, J Maurice Rojas, and Yuyu Zhu. Sub-linear point counting for variable separated curves over prime power rings. *arXiv preprint arXiv:2102.01626*, 2021. 13

[Săl05] Ana Sălăgean. Factoring polynomials over $\mathbb{Z}_4$ and over certain galois rings. *Finite fields and their applications*, 11(1):56–70, 2005. 5, 14, 95

[Sch74] Wolfgang M Schmidt. A lower bound for the number of solutions of equations over finite fields. *Journal of Number Theory*, 6(6):448–480, 1974. 116

[Sha93] Adi Shamir. On the generation of multivariate polynomials which are hard to factor. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 796–804. ACM, 1993. 2

[Sho09] Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge university press, 2009. 37

[Sie55] Wacław Sierpiński. Remarques sur les racines d'une congruence. *Annales Polonici Mathematici*, 1(1):89–90, 1955. 5

[Sir17] Carlo Sircana. Factorization of polynomials over $\mathbb{Z}/(p^n)$. In *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*, pages 405–412. ACM, 2017. 5, 14, 95, 132

[Sud97] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of complexity*, 13(1):180–193, 1997. 3

[Tit86] Edward Charles Titchmarsh. The Theory of the Riemann Zeta Function. *Oxford Science Publishers*, 1986. 8

[TMB98] Gabor Tardos and David A Mix Barrington. A lower bound on the mod 6 degree of the OR function. *Computational Complexity*, 7(2):99–108, 1998. 2

[vzGG13] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3rd edition, 2013. 76, 77

[vzGH96] Joachim von zur Gathen and Silke Hartlieb. Factorization of polynomials modulo small prime powers. Technical report, Paderborn Univ, 1996. 4, 14, 108, 132

[vzGH98] Joachim von zur Gathen and Silke Hartlieb. Factoring modular polynomials. *Journal of Symbolic Computation*, 26(5):583–606, 1998. (Conference version in ISSAC'96). 2, 4, 14, 108, 132

[vzGP01] Joachim von zur Gathen and Daniel Panario. Factoring polynomials over finite fields: A survey. *Journal of Symbolic Computation*, 31(1-2):3–17, 2001. 3, 65, 66

[Wei48] André Weil. *Variétés abéliennes et courbes algébriques*. Paris: Hermann, 1948. 8

[Wei49] André Weil. Numbers of solutions of equations in finite fields. *Bull. Amer. Math. Soc*, 55(5):497–508, 1949. 8

[Wei64] André Weil. Sur certains groupes d'opérateurs unitaires. *Acta mathematica*, 111(143-211):14, 1964. 8

[Wei65] André Weil. Sur la formule de Siegel dans la théorie des groupes classiques. *Acta mathematica*, 113:1–87, 1965. 8

[Zas69] Hans Zassenhaus. On hensel factorization, I. *Journal of Number Theory*, 1(3):291–311, 1969. 19

[ZG03] WA Zuniga-Galindo. Computing Igusa's local zeta functions of univariate polynomials, and linear feedback shift registers. *Journal of Integer Sequences*, 6(2):3, 2003. 5, 9, 12

[Zhu20] Yuyu Zhu. *Trees, point counting beyond fields, and root separation*. PhD thesis, Texas A&M University, 2020. 5, 13