An algorithm to find p-ordering of sets in succinct form [1]

Undergraduate Project (UGP) report submitted to

Indian Institute of Technology Kanpur

Bachelor of Technology

in

Computer Science and Engineering

by

Sayak Chakrabarti

(170648)

Under the supervision of

Prof. Rajat Mittal

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur

Fall Semester, 2020-21

December 12, 2020

[1] This project was jointly done with Aditya Gulati (Roll Number 170046)

DECLARATION

I certify that

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

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

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

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

Date: December 12, 2020	(Sayak Chakrabarti)
Place: Kanpur	(170648)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

KANPUR - 208016, INDIA

CERTIFICATE

This is to certify that the project report entitled "**An algorithm to find p-ordering of sets in succinct form** [2]" submitted by **Sayak Chakrabarti** (Roll No. 170648) to Indian Institute of Technology Kanpur towards no/partial fulfilment of requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering is a record of bona fide work carried out by him under my supervision and guidance during Fall Semester, 2020-21.

Date: December 12, 2020

Place: Kanpur

Prof. Rajat Mittal
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur - 208016, India

[2] This project was jointly done with Aditya Gulati (Roll Number 170046)

Abstract

Name of the student: **Sayak Chakrabarti** Roll No: **170648**

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

Department: **Computer Science and Engineering**

Project title: **An algorithm to find p-ordering of sets in succinct form** [3]

Project supervisor: **Prof. Rajat Mittal**

Month and year of Project submission: **December 12, 2020**

The notion of p-ordering was introduced by Manjul Bhargava in his PhD thesis to extend the definition of factorial functions to arbitrary subsets of Dedekind rings. In this article we will focus only on subsets of integers and give an algortithm to find p-ordering. p-ordering has several nice properties as well, one of them being a way to represent polynomials modulo prime powers, that makes the analysis of root sets easier. We also describe the motivation for analysis of root sets using a new concept called representative roots (Panayi, PhD Thesis, 1995; Dwivedi et. al 2019), and explore their properties.

[3] This project was jointly done with Aditya Gulati (Roll Number 170046)

Acknowledgements

I am grateful to Prof. Rajat Mittal for finding time in this difficult semester and conducting regular discussions remotely. His invaluable guidance has helped me have more clarity of the subject and understand proper methods to tackle a problem. I am also grateful to my friend Aditya Gulati for the helpful discussions and helping with the edge cases of the algorithm. Finally, I would like to thank my parents for providing a proper environment and resources such that my studies do not get hampered.

Contents

Chapter 1

Introduction

The study of polynomials in rings and fields has diverse applications in computer science and mathematics, especially in error correcting codes [8, 14, 20, 21], cryptography [10, 16, 18], computational number theory [1, 2] and computer algebra [15, 22]. The problem of factorization of polynomials and behavious of roots in rings has also been studied extensively. Some examples of factorization of polynomials in fields are [3, 4, 9]. However this becomes increasingly difficult when we move on to rings instead of fields, especially in rings of the form $\mathbb{Z}/p^k\mathbb{Z}$ for a prime p and positive integer k, that are not unique factorization domains.

The concept of *p-ordering* and *p-sequences* was introduced in [6], which is an important tool to analyze properties of root sets and polynomials, that was done in [17]. The initial motivation for introducing p-ordering was to generalize the concept of factorial functions to subsets other than just the set of integers. This concept was applied for arbitrary subsets of Dedekind rings ([6]), but in our problem we work on only subsets of rings of the form $\mathbb{Z}/p^k\mathbb{Z}$. The report [7] deals with p-ordering in subsets of integers.

In order to define p-ordering, we first need to define the valuation function wrt a prime p.

Definition 1.1. Given a non-zero integer a, we define *valuation* wrt p, $v_p(a) = v$ if and only if $p^v | a$ but $p^{v+1} \nmid a$. If $a = 0$, we define $v_p(0) = \infty$.

Definition 1.2 (p-ordering)**.** Given a subset $S \subseteq \mathbb{Z}$, we define *p-ordering* as the sequence (a_n) such that the following holds

- Any element is chosen from S as the first element a_0
- For every i, a_i is chosen from $S \backslash \{a_0, a_1, \ldots a_{i-1}\}$ such that $v_p((a_i - a_0)(a_i - a_1) \ldots (a_i - a_{i-1}))$ is minimum

More properties have been described in Chapter 2. An example of p-ordering from [13] is as follows. Given the set $S = \{1, 3, 4, 6, 9, 10\}$, two valid 3-orderings are $\{4, 6, 1, 9, 3, 10\}$ and $\{3, 10, 6, 4, 9, 1\}$.

An important problem in computer algebra and computational number theory is analysis of *root sets* modulo prime powers.

Definition 1.3. A set $S \subseteq R$ for a ring R is called a *root set* if there is a polynomial $f(x) \in R$ whose roots are exactly the elements of S.

We deal with the case when the ring R is of the form $\mathbb{Z}/p^k\mathbb{Z}$. Note that this is not a trivial problem. If we consider modulo p^2, every subset is not a root set, for example $\{0, p\}$ is not a root set, but the smallest root set containing this is $\{0, p, 2p, \ldots (p-1)p\}$. This concept of root sets was analyzed in [11] and [17] gave explicit bounds and recursive formulas to calculate the number of root sets modulo p^k. The analysis of root sets also motivates the question of p-ordering, which will be more clearly explained in Chapter 3.

After this we move closer to our main problem of finding a p-ordering. Note that when we are given a subset of $\mathbb{Z}/p^k\mathbb{Z}$ which contains about $\mathcal{O}(p)$ elements, which is exponential in $\log p$! For this we will introduce the concept of minimal notation using representative roots in Chapter 3, and move on to giving an efficient algorithm to find p-ordering on a set given in this form in Chapter 5. This algorithm (Algorithm 1) is a significant improve compared to the naive approach given in [6]. Furthermore, [13] developed an algorithm to find p-ordering on a normal subset of integers as well, not necessarily given in succinct representation. It also led to a method using representative roots to count the total number of root sets modulo small powers of p (given upto modulo p^4 in [13]).

Chapter 2

p-ordering on subsets of integers

The definition of p-ordering is included in 1.2. This leads to many interesting properties which will be described in this chapter.
We have seen from the given example that a subset can have more than one p-ordering. The set S was $\{1, 3, 4, 6, 9, 10\}$ and the two valid 3-orderings were $\{4, 6, 1, 9, 3, 10\}$ and $\{3, 10, 6, 4, 9, 1\}$.
However note that the increase in valuations is the same in both, i.e. for $i = [5]$, we calculate $v_p((a_i - a_0)(a_i - a_1)\ldots(a_i - a_{i-1}))$. Considering the first term of this series as $v_p(a_0)$, we get the series as $\{3^0, 3^0, 3^1, 3^1, 3^2, 3^4\}$. However this sequence being the same is not a coincidence! For this we first define p-sequence.

Definition 2.1 (p-sequence)**.** For a given p-ordering (a_n), the *p-sequence* is defined as the sequence (v_n)

- $v_0 = 1$
- $v_i = v_p((a_i - a_0)(a_i - a_1)\ldots(a_i - a_{i-1}))$

We also denote each the i^{th} of the p-sequence as $v_p(S, i)$, and refer to this as the p-value at i^{th} step.

[6] proved the following theorem for any generalized Dedekind ring. But since in our article we are only concerned with integers, we restrict the ring to $\mathbb{Z}$ or $\mathbb{Z}/p^k\mathbb{Z}$. The analysis over integers has also been illustrated in [7].

Theorem 2.2. *For any two p-orderings on a subset of $\mathbb{Z}$, the associated p-sequences are the same.*

Bhargava also defined the generalized factorial, which in our case of subsets $S \subseteq \mathbb{Z}$ is defined as

$$k!_S = \Pi_{primes\ p} p^{v_p(S,k)} \tag{2.1}$$

Using this definition [7] showed several interesting properties that hold true for normal factorials that we know, hold true for this generalized definition as well. However in our article, since we deal with modulo only a single prime, we will write $k!_S = p^{v_p(k!_S)}$ in Chapter 3.

Lemma 2.3. *For $S \subseteq \mathbb{Z}$ and $k, l \in \mathbb{N}$, $(k+l)!_S$ is always divisible by $k!_S l!_S$.*

We define $d(S, f) = gcd\{f(a)|a \in S\}$. [7] also proved the following lemmas.

Lemma 2.4. *For a primitive polynomial f of degree k, $d(S, k)|k!_S$*

Lemma 2.5. *Let $a_0, a_1, \dots a_n$ be $n+1$ integers. Then the product*

$$\Pi_{i<j}(a_j - a_i) \tag{2.2}$$

is divisible by $1!_S 2!_S \dots n!_S$

Lemma 2.6. *The number of polynomials functions from S to $\mathbb{Z}/n\mathbb{Z}$ is $\Pi_{k=0}^{n-1} \frac{n}{gcd(n,k!_S)}$.*

In the proofs of these lemmas, an important lemma was proved, that will be used in Chapter 3 as well.

Lemma 2.7. *A polynomial f over integers, written in the following form, such that (a_n) is a p-ordering*

$$f(x) = \sum_{i=0}^{k} c_i(x - a_0)(x - a_1)\dots(x - a_{i-1}) \tag{2.3}$$

vanishes on S modulo p^e if and only if every term $c_i(x - a_0)(x - a_1)\dots(x - a_{i-1})$ vanishes $\forall i \in \{0, 1, \dots k\}$

Now we state a few more results related to p-ordering which will help in building the algorithm.

Lemma 2.8. *Let $S \subseteq \mathbb{Z}$ and (a_n) be a p-ordering, then*

1. $(a_0 + x, a_1 + x, a_2 + x, \dots)$ *is a p-ordering on $S + x$ for any $x \in \mathbb{Z}$*
2. $(a_0.x, a_1.x, a_2.x, \dots)$ *is a p-ordering on $x.S$ for any $x \in \mathbb{Z} \backslash \{0\}$*

The next theorem is from [17], another important theorem related to p-ordering, which has been used to find a p-ordering on a subset of $\mathbb{Z}$ (not given in succinct representation) in [13].

Theorem 2.9. *Let $S \subseteq \mathbb{Z}$, and $S_j = \{s \in S | s \equiv j \mod p\}$ for $j = 0, 1, \dots p-1$. Then for any $x \equiv j \mod p$, we have*

$$v_p(\Pi_{a \in S}(x - a)) = v_p(\Pi_{b \in S_j}(x - b)) \tag{2.4}$$

Although this theorem can be easily proved, it is a very important property, based on which the algorithm to find p-ordering on a subset of integers has been developed in [13].

Chapter 3

Root sets modulo prime powers

Root sets were defined in Definition 1.3. An important question in computer science is to calculate the number of root sets modulo a prime power. We showed how every subset $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$ is not a root set. In order to find the total number of root sets modulo a prime power, we make use of some elementary yet important properties which were more elaborately proven in [11].

Theorem 3.1. *If S is a root set modulo a power of prime p, then $\forall j \in [p]$, $S_j = \{a \in S | a \equiv j \mod p\}$ is also a root set modulo that power of p.*

This theorem states that if we have a root set, we can reduce it to an union smaller root sets. Similarly we can take an union of smaller root sets with elements congruent to each other modulo p to find a bigger root set, given by the following theorem.

Theorem 3.2. *If $S_0, S_1, \ldots S_{p-1}$ is a collection of root sets modulo a power of p, such that all elements of S_i are conguent to j modulo p $\forall j \in \{0, 1, \ldots p-1\}$, then $S_1 \cup S_2 \cup \ldots S_{p-1}$ is also a root set modulo the same power of p.*

Based on these theorems, we can reduce any general root set to root sets only having elements divisible by p. This simplifies our analysis of counting root sets, as now we will have to deal only with numbers which are congruent to $0 \mod p$. Now if we have a root set of the form $\{pa_1, pa_2, \ldots pa_n\}$, then we can reduce this to a root set modulo a lower power of prime p. This is the basis of counting the number of root sets done by [17] which we will now explain.

First we define $\mathcal{N}_{p^k}$ to be the number of root sets modulo p^k which contains only elements that are divisible by p. Since we can write a root set S as $(0 + S_0) \cup (1 + S_1) \cup \ldots (p - 1 + S_{p-1})$, where each of S_i's are root sets only with elements divisible by p (Theorem 3.2), we infer that the total number of root sets is $\mathcal{N}_{p^k}^p$. So it is sufficient for us to find the value of $\mathcal{N}_{p^k}$ to find the total number of root sets modulo p^k. [11] gives a table for some small primes and their powers, while [17] gives an explicit recurrence relation with lower and upper bounds. The analysis also extensively uses concepts from p-ordering as described in Chapter 2. We also denote *p-root sets* modulo p^k as root sets in $\mathbb{Z}/p^k\mathbb{Z}$ such that all the elements are divisible by p.

Definition 3.3. For a p-ordering on $S \subseteq \mathbb{Z}$, we define the smallest j such that $p^k | v_p(S, j)$ as $\mu(S, k)$. We also define $\mu(\phi, k) = 0$.

Lemma 3.4. *Let $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$ and (a_n) be a p-ordering on it.Given any polynomial $f(x) \in \mathbb{Z}/p^k\mathbb{Z}$, we can write f in the form*

$$f(x) = \sum_{j=0}^{d} c_j (x - a_0)(x - a_1) \ldots (x - a_j) \tag{3.1}$$

for some $c_j \in \mathbb{Z}/p^k\mathbb{Z}$ and $d \leq \mu(S, k)$

This result is inspired from Lemma 2.7. Note that when we consider S as a p-root set, $\mu(S, t) \leq t$ for every t. This means that over such p-root sets, the representation of any polynomial $f(x)$ will have degree less than or equal to k written in the form of Lemma 3.4. However this is an unexpected result as we can choose a polynomial of any degree, but there is always a representation, which gives the same evaluations, but still has a lower degree. This representation of polynomials over p-root sets has been exploited in the proofs of the theorems and lemmas due to [17] to be stated in the remaining part of this chapter.

Lemma 3.5. *If $T = \{s_0p, s_1p, s_2p, \ldots\} \subset \mathbb{Z}$ be a p-root set modulo p^k, then $S = \{s_0, s_1, s_2, \ldots\}$ is a p-root set modulo $p^{k-\mu(S,k)}$.*

The proof of this follows directly by using the representation of f given by Lemma 3.4 and considering the set of a_i's as a p-ordering over elements of T. Similarly we can show the following lemma as well.

Lemma 3.6. *Let $S = \{s_0, s_1, \dots\} \subset \mathbb{Z}$ be a root set modulo p^k, then $T = \{s_0p, s_1p, \dots\}$ is a p-root set modulo $p^{k+\mu(S,k)}$.*

However from Lemmas 3.5 and 3.6, the reader might be tempted to think that for two sets S, T such that $T = p.S$, $\mu(T, k + \mu(S, k)) = \mu(S, k)$ and the converse $\mu(S, k - \mu(T, k)) = \mu(T, k)$ will hold true. However this is not true. We can show that indeed $r!_T = p^r r!_S$, implying $\mu(T, k + \mu(S, k)) = \mu(S, k)$. However the converse is not always true. Since $\mu(T, k) \le k$, we can only infer that $\mu(S, k - \mu(T, k)) \le \mu(T, k)$. However if equality held in this, we would have been able to get a one-one correspondence between T and S in Lemmas 3.5 and 3.6. But since this does not hold true, we will be able to find a lower bound (not a precise recursion) for $F(k, r)$ defined as follows.

Definition 3.7. $F(k, r)$ is defined as the number of p-root sets T modulo p^k such that $\mu(T, k) = r$

Note that $\mathcal{N}_{p^k} = \sum_{r=0}^{k} F(k, r)$. Now we give a lower bound of $F(k, r)$ using the inequalities stated above.

Theorem 3.8. *For a prime $p > k$, consider the function $f(k, r)$ such that $f(k, k) = 1$, $f(k, 0) = 1$, $f(k, r) = 0$ for all $r > k$ and otherwise, we have the recursion*

$$f(k, r) = \sum_{r_0+\dots+r_{p-1}=r} \left(\Pi_{i=0}^{p-1} f(k - r, r_i)\right) \tag{3.2}$$

then $F(k, r)$ is bounded below by $f(k, r)$ $\forall k, r$

The proof follows from recursively breaking down a p-root set to root sets modulo smaller powers of p. A formal proof can be found in [17]. Using a similar technique of this proof leads to another result giving the upper bound of $F(k, r)$ given as

Theorem 3.9. *For a prime $p > k$, consider the function $g(k, r)$ such that $g(k, k) = 1$, $g(k, 0) = 1$, $g(k, r) = 0$ for all $r > k$ and otherwise, we have the recursion*

$$g(k, r) = g(k - r + s, s) + \sum_{r_0+\dots+r_{p-1}=r} \left(\Pi_{i=0}^{p-1} g(k - r, r_i)\right) \tag{3.3}$$

then $F(k,r)$ is bounded above by $g(k,r)$ $\forall k, r$

Theorems 3.8 and 3.9 give lower and upper bounds of $F(k,r)$, which are polynomials. The leading terms of both of $f(k,r)$ and $g(k,r)$ are $\frac{1}{r!}p^{r(k-r)}$. Furthermore [17] showed that if $k - r > 1$ then for both polynomials, we can find the second largest degree term as $-\frac{1}{2(r-2)!}p^{r(k-r)-1}$. It can also be similarly shown that for fixed $k < p$, $\sum_{r=0}^{k} f(k,r)$ and $\sum_{r=0}^{k} g(k,r)$ has leading term $c_k p^{[\frac{k^2}{4}]}$, while the second largest term, for $k > 4$ is $d_k p^{[\frac{k^2}{4}]-1}$, where c_k and d_k are defined as follows:

$$\begin{aligned} c_k &= \left(\frac{k}{2}!\right)^{-1} && \text{if } k \text{ is even} \\ &= \left(\frac{k-1}{2}!\right)^{-1} + \left(\frac{k+1}{2}!\right)^{-1} && \text{otherwise} \end{aligned}$$

and for $k \geq 4$,

$$\begin{aligned} d_k &= \left(\frac{k-2}{2}!\right)^{-1} + \left(\frac{k+2}{2}!\right)^{-1} - \left(2\left(\frac{k}{2}!\right)\right)^{-1} && \text{if } k \text{ is even} \\ &= \frac{-1}{2}\left(\left(\frac{k-3}{2}!\right)^{-1} + \left(\frac{k-5}{2}!\right)^{-1}\right) \end{aligned}$$

Hence the total number of root sets is $\mathcal{N}_{p^k}$, which is approximately $\Theta(e^{\frac{d_k}{c_k}}(c_k p^{[\frac{k^2}{4}]})^p)$ for $k \geq 4$.

[17] also gave a more complicated recursion to explicitly obtain the function that returns the total number of p-root sets modulo p^k using more variables. We refer the reader to Section 4 of [17] for more details regarding this.

Through this chapter we have seen the motivation to use p-ordering in root sets, and methods to obtain $\mu(S,k)$ and other functions. Having an algorithm to do so will help in this and related problems, giving us a clear motivation for Chapter 5. But before moving on to the main algorithm, we also describe representative roots and succinct representation in the next chapter as the tools will be used in the development of the algorithm.

Chapter 4

Representative Roots and Succinct Representation

Representatives were introduced in [19]. We will use the symbol $*$ as a representative, which "represents" an entire ring R. In our article we are mainly concerned with rings of the form $\mathbb{Z}/p^k\mathbb{Z}$ for a prime p and an integer $k \geq 1$.

In the case of $R = \mathbb{Z}/p^k\mathbb{Z}$, we will use the notation $y = y_0 + y_1 p + \dots y_i p^i + p^{i+1}*$ such that $i+1 < k$ and $y_i \in R/\langle p \rangle$ $\forall i$. This representation of y stands for the entire set $S_y \subseteq R$ given by:

$$S_y = \{y_0 + y_1 p + \dots y_m p^m + z_{m+1} p^{m+1} + \dots z_{k-1} p^{k-1} | z_{m+1}, z_{m+2}, \dots z_{k-1} \in R/\langle p \rangle\} \tag{4.1}$$

Notice that $*$ stands for the entire ring R but in this representation, we already have fixed the first $i+1$-coordinates, and are left with the remaining $k-i-1$ ones. These remaining coordinates are filled by all the elements of R and we get the size of S_y as p^{k-i-1}.

We will also sometimes denote a representative in the above form as $y = \beta + p^{i+1}*$, where $\beta \in \mathbb{R}/\langle p^{i+1} \rangle$. Now, for a polynomial $f(x)$, $y = \beta + p^{i+1}*$ is termed as a representative root if $\forall a \in S_y$, $f(a) \equiv 0 \mod p^k$.

Theorem 4.1 ([5]). *A polynomial $f(x)$ of degree d has at most d-many representative roots modulo p^k.*

Based on this, [12] developed an algorithm to return all possible roots in terms of representative roots of a polynomial $f(x)$ modulo p^k in randomized polynomial time (polynomial in $k, \log p, d$). Note that the total number of roots might be exponential, but the representative roots act as compact data structures that return exponentially many roots in polynomial space.

This property of a succinct data structure gives us the motivation to represent sets of integers of size $O(p^k)$ into more succinct form by considering them as an union of several representative roots. For example the set $(1 + 7^2*) \cup (9 + 7^3*) \subset \mathbb{Z}/7^5\mathbb{Z}$ contains about 392 numbers but we can represent them using only 2 representative roots! A smaller example would be the set $\{1, 3, 6, 8, 11, 13, 16, 18, 21, 23\} \subset \mathbb{Z}/5^2\mathbb{Z}$ can be represented as the union of the representative roots $1 + 5^1*$ and $3 + 5^1*$. In the rest of this chapter we give properties of representative roots and succinct representation of subsets of $\mathbb{Z}/p^k\mathbb{Z}$ based on the analysis provided in [13]. We will start by defining succinct/minimal representation.

Definition 4.2. Let $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$. A set of representative roots $S^{rep} = \{r_1, r_2, \ldots r_l\}$ for $r_i = \beta_i + p^{k_i}*$ is said to be in *minimal representation* if

1. $S = \cup_{i=1}^{l} r_i$
2. $\nexists r_i, r_j \in S^{rep}$ such that $r_i \subseteq r_j$
3. $\forall i,\ \beta_i + p^{k_i - 1}* \notin S$

[13] also proved the following theorem based on a few observations on representative roots.

Theorem 4.3. *Given any set $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$, the minimal representation is unique.*

In order to prove this, we need the following lemma.

Lemma 4.4. *For any two representative roots $r_1 = \beta_1 + p^{k_1}*$ and $r_2 = \beta_2 + p^{k_2}*$, we have the fact that either they are disjoint, or one is contained in another.*

Proof. Suppose that r_1 and r_2 are not disjoint. We prove this in two cases.

First let us assume that $k_1 = k_2 = \ell$ (let). Since $\beta_1, \beta_2 \subseteq \mathbb{Z}/p^{\ell-1}\mathbb{Z}$, if there is an element in their intersection, this implies $\exists b, c$ such that $\beta_1 + p^\ell b = \beta_2 + p^\ell c$. Now

considering this equation modulo p^ℓ we get $\beta_1 = \beta_2$ implying $r_1 = r_2$.
Now, WLOG assume there is an element in their intersection and $k_1 < k_2$. We can find b, c such that $\beta_1 + p^{k_1} b = \beta_2 + p^{k_2} c$. Considering this equation modulo p^{k_1}, we have $\beta_1 = \beta_2 \mod p^{k_1}$. This implies there is some α such that $\beta_2 = \beta_1 + p^{k_1}\alpha$. From this we get $r_1 = (\beta_2 + p^{k_2} *) = \beta_1 + p^{k_1}(\alpha + p^{k_2 - k_1} *) \subset r_1$. □

Based on these results, we consider a set given in succinct representation and give an efficient algorithm to find a p-ordering on this set in the next chapter.

Chapter 5

Main Result: An algorithm to find p-ordering

In Chapter 3 we saw the importance of p-ordering in analysis of root sets modulo prime powers. Since the recurrence given by [17] uses p-ordering to find the total number of root sets, a natural question arises to find an efficient algorithm to return a p-ordering on a subset of integers.
A naive approach was given in [6]. Given $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$, we can find a p-ordering by checking using brute force which element gives the minimum valuation wrt p for the expression given in Definition 1.2. When we have already chosen a p-ordering of length l given by $\{a_0, a_1, \dots a_{l-1}\}$, the next element x is checked using brute force on elements of $S \backslash \{a_0, a_1, \dots a_{l-1}\}$ to minimize $v_p((x-a_0)(x-a_1)\dots(x-a_{l-1}))$. This takes time $\mathcal{O}(n^3 k \log p)$. Moreover, [13] gave a more efficient algorithm to find p-ordering in $\mathcal{O}(n^2 k \log p)$ steps. However note that if the given subset is exponential (of the order of a polynomial in p) and we want to find p-ordering upto only a few number of terms (since we usually need to find the value $\mu(S,k)$ which is usually much less than the given susbet), this algorithm will not be quite efficient. This gives us the motivation to find an algorithm for finding p-ordering on a subset of integers given in succinct representation (Definition 4.2).

In Chapter 5 we saw how representative roots can be used to succinctly represent a set $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$ containing exponentially many values in much smaller space. An example (from [13]) on a p-ordering on this is as follows.

Given a set $S = \{1, 2, 4, 7, 10, 11, 13, 16, 19, 20, 22, 25\}$ and a prime $p = 3$. Considering the ring $\mathbb{Z}/3^3\mathbb{Z}$, it can be written succinctly as $\{1+3*, 2+3^2*\}$. A 3-ordering on this set is given by $\{1, \mathbf{2}, 4, 7, \mathbf{11}, 10, \mathbf{20}, 13, 16, 19, 22, 25\}$. In this chapter, we give an algorithm to find p-ordering like this with the set given in input as an union of representative roots. The main theorem is as follows.

Theorem 5.1. *Given a set $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$ written succinctly in terms of d representative roots, we can efficiently find a p-ordering on this set of length n in $\mathcal{O}(d^2 k \log p + nk \log p + np)$ time.*

We prove this theorem by giving an algorithm to compute p-ordering.

5.1 The Algorithm

In order to construct an algorithm, we need to consider the change of p-values corresponding to the addition of each element from some representative root. We consider d representative roots given to us for the minimal representation which are of the form $\beta_i + p^{k_i}*$ for $i \in [d]$.

Lemma 5.2. *If $r_1 = \beta + p^{k_1}*$, $r_2 = \beta + p^{k_2}*$ be two representative roots, then for any $a \in r_1$ and $b \in r_2$,*

$$v_p(a - b) = v_p(\beta_1 - \beta_2)$$

Proof. If $a = \beta_1 + p^{k_1}y_1$ and $b = \beta_2 + p^{k_2}y_2$ for some integers y_1, y_2, we have $a - b = (\beta_1 - \beta_2) + (p^{k_1}y_1 - p^{k_2}y_2)$. WLOG assume that $k_1 \leq k_2$.
Now if $v_p(\beta_1 - \beta_2) \geq k_1$, then the first k_1 coordinates in the p-adic decomposition β_2 is the same as β_1, implying that $r_2 \subseteq r_1$, which is a contradiction, as the representative roots are disjoint as given in the succinct representation. □

Lemma 5.3. *If $S = \{a_0, a_1, \dots\}$ is a p-ordering, and $S_j = \{a_{i_1}, a_{i_2}, \dots\} \subseteq S$, consisting of elements from S which correspond to the j^{th} representative root taken in the same order as in S, form a p-ordering*

Proof. First note from Lemma 2.8 that $\beta_j + p^{k_j}\{0, 1, 2, \dots\}$ is a p-ordering. So it is sufficient to show that all the elements corresponding to r_j in S are present in this form in a valid p-ordering. We prove this using induction on the length of p-ordering. For one element definitely we can choose any element from r_j, and for the sake of our algorithm we choose $\beta_j + p^{k_j}.0 = \beta_j$. This is a valid choice and hence base case of induction is correct.
Suppose that we have added $\beta_j + p^{k_j}\{0, 1, 2, \dots t-1\}$ to S in this order. Now, we add another element, say x, from r_j to the already existing p-ordering. Note that this element will contribute only $v_p(\beta_i - \beta_j)$ to the p-value for any element from representative root r_i for $i \neq j$ (Lemma 5.2) irrespective of the choice of x from r_j. So, we need to minimize the valuation added only due to the already existing elements from r_j in S. Now since $\beta_j + p^{k_j}\{0, 1, 2, \dots t-1\}$ are the elements present in S from r_j until now, adding the element $\beta_j + p^{k_j}t$ gives us a valid p-ordering as well. This concludes the proof. □

Hence, from Lemma 5.3, we know that $\beta_j + p^{k_j}\{0, 1, 2, \dots\}$ are sub-sequences of the required p-ordering. Our main task is to merge them properly such that valuation wrt p is minimized at every step of addition of a new element.

Now, in order to consider the p-value, when we consider a new element x to be added to the p-ordering, we need to take summation over $v_p(x - a_i)$ for all a_i's already added to the p-ordering. When we consider this summation, there can be 2 cases, x can belong to the same representative root of a_i, or a different one.

If x and a_i are from i^{th} and j^{th} representative roots respectively, $v_p(\beta_i - \beta_j)$ is added to the p-value. However if they are from the same representative root, say the i^{th} representative root, we do that following. From Lemma 2.8, we get that $\beta_i + p^{k_1}.\{0, 1, 2, 3, \dots\}$ is a p-ordering. If there are already t elements in the p-ordering from this i^{th} representative root, when we add the $(t+1)^{th}$ element next, the p-value contributed due to this i^{th} representative root will be $tk_i + v_p(t!)$. Using this idea, we maintain the *valuations* array. The i^{th} entry of this array basically stores the p-value that we will encounter if we add an element to the already existing p-ordering from the i^{th} representative root. Also, due to Lemma 2.8, we know that $\beta_j + p^{k_j}\{0, 1, 2, \dots\}$ is a p-ordering on elements of r_j. Hence, we store pointers i_j for every representative root, which represents the number of terms already added to the p-ordering from the j^{th} representative root. The next term to be added to the p-ordering from r_j will thus be $\beta_j + p^{k_j} i_j$, and we will increment i_j by one.

Based on these, we minimize the p-value increase at every step and give the following algorithm, Algorithm 1.
The inputs are given as n, the required length of the p-ordering and S, a set given in minimal representation consisting of d representative roots $r_1, r_2, \dots r_d$ where each of $r_i = \beta_i + p^{k_i}*$.

Algorithm 1 Find p-ordering from minimal notation

```
procedure CORRELATE($S$)
    $Corr \leftarrow [0]_{d \times d}$
    $Corr \leftarrow [0]_{d \times d}$
    for $j \in [1, ..., d]$ do
        for $k \in [1, ..., d]$ do
            $Corr[j][k] \leftarrow v_p(\beta_j - \beta_k)$
    return $Corr$
procedure $p$-EXPONENT_INCREASE($n$)
    $v_p(1) \leftarrow 1$
    for $j \in [1, ..., n]$ do
        $v_p((j+1)!) \leftarrow v_p(j+1) * v_p(j!)$
        $p\_exponent[j] \leftarrow v_p((j+1)!) - v_p(j!)$
    return $p\_exponent$
procedure FIND_$p$-ORDERING($S, n$)
    $corr \leftarrow$ CORRELATE($S$)
    $increase \leftarrow p$-EXPONENT_INCREASE($n$)
    $valuations \leftarrow [0]_d$
    $p\_ordering \leftarrow \{\}$
    $i_1, i_2 \dots i_{|S|} \leftarrow 0$
    for $i \in \{1, 2, \dots n\}$ do
        $min \leftarrow \min\{valuations\}$
        $index \leftarrow \text{argmin}\{valuations\}$
        $p\_ordering.append(\beta_{index} + p^{k_{index}} * i_{index})$
        for $j \in [1, ..., d]$ do
            if $j = index$ then
                $valuations[j] \leftarrow valuations[j] + k_{index} + increase[i_j]$
            else
                $valuations[j] \leftarrow valuations[j] + corr(index, j)$
        $i_{index} \leftarrow i_{index} + 1$
    return $p\_ordering$
```

5.2 Proof of Correctness and Time Complexity

Theorem 5.4. *Algorithm 1 correctly returns a valid p-ordering on S.*

Proof. According to the definition of *valuations* array, we first show that it correctly stores the p-value which will occur if we add the next element from the i^{th} representative root in its i^{th} entry. When we add a new element from the r_j to the p-ordering, we update $valuations[t]$ in Step 33 for every $i \neq j$ according to Lemma 5.2.

Now, when we update $valuations[j]$, we have seen how the valuations corresponding to the same representative root will be $i_j k_j + v_p((i_j - 1)!)$, for a pointer i_j storing the number of elements from r_j in the p-ordering until now. However after adding this element to the p-ordering, the p-value due to the same representative root would be $(i_j)k_j + v_p((i_j)!)$, and hence the increase is $k_j + v_p((i_j)!) - v_p((i_j - 1)!)$. This is precisely what is being done in Step 31.

Next, we add an element from r_j such that $valuations[j]$ has minimum valuation wrt p, and hence pertains to the definition of p-ordering (Definition 1.2) □

Theorem 5.5. *Algorithm 1 takes $\tilde{\mathcal{O}}(d^2 k \log p + nk \log p + np)$ time to return a p-ordering of length n, where the set $S \subseteq \mathbb{Z}/p^k\mathbb{Z}$ is given as a union of d many representative roots.*

Proof. The procedure Correlate(S) runs a double for loop, calculating valuations every time taking $\mathcal{O}(k \log p)$ time, and hence takes $\tilde{\mathcal{O}}(d^2 k \log p)$ in total. The procedure p-Exponent_Increase(n) runs a single for loop n times with each iteration taking $\tilde{\mathcal{O}}(k \log p)$-time, and hence $\tilde{\mathcal{O}}(nk \log p)$ due to this. Finally in Find_p-Ordering(S), the main for loop runs n-times and in each iteration we perform operations taking $\tilde{O}(d)$ time, and hence total time due to Find_p-Ordering(S) is $\tilde{\mathcal{O}}(nk \log p)$. Adding these, the total time taken by Algorithm 1 is $\tilde{\mathcal{O}}(d^2 k \log p + nk \log p + np)$. □

Proof of Theorem 5.1 follows from the proofs of Theorems 5.4 and 5.5.

Bibliography

[1] Leonard Adleman and Hendrik Lenstra. "Finding Irreducible Polynomials over Finite Fields". In: *Proc. 18th Annual ACM Symp. on Theory of Computing (STOC), 350 - 355 (1986).* Nov. 1986, pp. 350–355. DOI: 10.1145/12130.12166.

[2] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. "PRIMES is in P". In: *Annals of mathematics* (2004), pp. 781–793.

[3] E. R. Berlekamp. "Factoring polynomials over finite fields". In: *Bell System Technical Journal* 46(8) (1967), pp. 1853–1859.

[4] E.R. Berlekamp. "Factoring polynomials over large finite fields". In: *Mathematics of Computation* 24 (July 1970), pp. 713–735. DOI: 10.1090/S0025-5718-1970-0276200-X.

[5] Jérémy Berthomieu, Grégoire Lecerf, and Guillaume Quintin. "Polynomial root finding over local rings and application to error correcting codes". In: *Applicable Algebra in Engineering, Communication and Computing* 24.6 (2013), pp. 413–443.

[6] Manjul Bhargava. "P-orderings and polynomial functions on arbitrary subsets of Dedekind rings". In: *Journal Fur Die Reine Und Angewandte Mathematik - J REINE ANGEW MATH* 1997 (Jan. 1997), pp. 101–128. DOI: 10.1515/crll.1997.490.101.

[7] Manjul Bhargava. "The Factorial Function and Generalizations". In: *American Mathematical Monthly* 107 (Nov. 2000). DOI: 10.2307/2695734.

[8] R.C. Bose and D.K. Ray-Chaudhuri. "On a class of error correcting binary group codes *". In: *Information and Control* 3 (Mar. 1960), pp. 68–79. DOI: 10.1016/S0019-9958(60)90287-4.

[9] David Cantor and Hans Zassenhaus. "A New Algorithm for Factoring Polynomials Over Finite Fields". In: *Mathematics of Computation* 36 (Apr. 1981). DOI: 10.2307/2007663.

[10] Benny Chor and Ronald Rivest. "A Knapsack Type Public Key Cryptosystem Based On Arithmetic in Finite Fields". In: *IEEE Transactions on Information Theory* 34 (Sept. 2001). DOI: 10.1109/18.21214.

[11] Bruce Dearden and Jerry Metzger. "Roots of Polynomials Modulo Prime Powers". In: *Eur. J. Comb.* 18 (Aug. 1997), pp. 601–606. DOI: 10.1006/eujc.1996.0124.

[12] Ashish Dwivedi, Rajat Mittal, and Nitin Saxena. "Efficiently Factoring Polynomials Modulo p^4". In: *International Symposium on Symbolic and Algebraic Computation (ISSAC)* (July 2019), pp. 139–146. DOI: 10.1145/3326229.3326233.

[13] Aditya Gulati, Sayak Chakrabarti, and Rajat Mittal. "On algorithms to find p-ordering". In: *International Conference on Algorithms and Discrete Applied Mathematics (CALDAM)* (Feb. 2021).

[14] A. Hocquenghem. "Codes Correcteurs D'Erreurs". In: *Chiffres, Revue de l'Association Française de Calcul* 2 (Jan. 1959).

[15] Arjen Lenstra, H. Lenstra, and László Lovász. "Factoring Polynomials with Rational Coefficients". In: *Mathematische Annalen* 261 (Dec. 1982). DOI: 10.1007/BF01457454.

[16] H. Lenstra. "On the Chor—Rivest knapsack cryptosystem". In: *Journal of Cryptology* 3 (Jan. 1991), pp. 149–155. DOI: 10.1007/BF00196908.

[17] Davesh Maulik. "Root Sets of Polynomials Modulo Prime Powers". In: *J. Comb. Theory, Ser. A* 93 (Jan. 2001), pp. 125–140. DOI: 10.1006/jcta.2000.3069.

[18] A. Odlyzko. "Discrete logarithms and their cryptographic significance". In: *Advances in Cryptography, EUROCRYPT '84, Proceedings, Lecture Notes in Computer Science* 209 (1985), pp. 224–314.

[19] Peter N Panayi. "Computation of Leopoldt's P-adic regulator." PhD thesis. University of East Anglia, 1995.

[20] I. Reed and G. Solomon. “Polynomial Codes Over Certain Finite Fields”. In: *Journal of the Society for Industrial and Applied Mathematics* 8 (June 1960), pp. 300–304. DOI: 10.2307/2098968.

[21] M Sudan. “Decoding Reed Solomon Codes beyond the Error-Correction Bound”. In: *Journal of Complexity* 13 (Mar. 1997), pp. 180–193. DOI: 10.1006/jcom.1997.0439.

[22] Hans Zassenhaus. “On Hensel factorization II”. In: *Journal of Number Theory* 1 (July 1969), pp. 291–311. DOI: 10.1016/0022-314X(69)90047-X.