Irreducible polynomials over finite fields for coding and cryptography

A thesis presented for the degree of

Doctor rerum naturalium (Dr. rer. nat.)

of the Faculty of Mathematics and the Natural Sciences,

University of Rostock

presented by

Anna-Maurin Graner

Rostock, 2024

https://doi.org/10.18453/rosdok_id00004733

Gutachter:

Prof. Dr. Gohar Kyureghyan, Universität Rostock

Univ.-Doz. Dr. Arne Winterhof, Johann Radon Institute for Computational and Applied Mathematics (RICAM), Linz

Jahr der Einreichung: 2024

Jahr der Verteidigung: 2024

Abstract

Irreducible polynomials over finite fields play a vital role in modern coding and cryptography. In this thesis we determine a closed explicit formula for all generating polynomials of a very popular set of codes - the cyclic and the constacyclic codes over finite fields. This problem is equivalent to the factorization of the polynomials $X^n - 1$ and $X^n - a$ into monic irreducible factors over a finite field for positive integers n and a field element $a \neq 0$. Though partial results existed no closed explicit formula for the factorization of these two polynomials was known. From our formula for the polynomial $X^n - 1$ we derive the explicit factorization of its famous factor Φ_n, the so-called n-th cyclotomic polynomial, over a finite field. Among the first to study the factorization of Φ_n over finite fields was Carl Friedrich Gauss in 1876 and the polynomial has fascinated many mathematicians since then. Additionally, our formula for the polynomial $X^n - a$ can be extended to a closed explicit formula for the factorization of any polynomial of the form $f(X^n)$ where f is a monic irreducible polynomial over a finite field. This formula yields an algorithm for the factorization of $f(X^n)$ which performs much better than the existing factoring algorithms in the computer algebra systems SageMath and Magma for large positive integers n.

For the construction and efficient implementation of finite fields we need irreducible polynomials of a given degree and a low weight. Applications in cryptography call for irreducible polynomials of a high order or other additional properties. In general it is a challenging task to find irreducible polynomials. This task becomes even harder when we are looking for irreducible polynomials with very specific properties. To tackle this problem we present a construction of a large set of irreducible polynomial of the same degree from one given irreducible polynomial. This large number allows the selection of the candidates with the desired properties.

Acknowledgments

Firstly, I thank my advisor Gohar Kyureghyan who introduced me to finite fields, coding and cryptography, which I only fell in love with at second sight. You always let me work freely but whenever I needed it, I found an open ear and a willingness to discuss whatever I needed. Thank you as well for teaching me about the importance of examples and how to present the ideas behind complicated technical proofs.

Secondly, I thank whomever will be reviewer for taking the time to read this thesis.

I thank all of my colleagues at the University of Rostock, Thomas, Solvejg, Bahadir, Sabrina, Max, Susann, Christoph, Heike, Jan-Christoph, Annika, Björn, Jenny, Achill, Lucas, Karin, Andreas, Jessica, Tomass, Peter, Mathias and many more. Thank you for friendship, for lively mathematical and non-mathematical discussions, for your technical, bureaucratic and mental support and for a wonderful time at the Institute of Mathematics. I will miss you all.

I thank Anna Dittus, with whom I could share and discuss the experience of being a woman in academia. Thank you for all the wonderful evenings and for going through the ups and downs with me. I am so happy to have you.

I thank Lukas Kölsch, my PhD buddy, with whom I shared so many of my first moments in academia. Thank you for an amazing time and for being such a good friend.

I thank my colleague and friend Sasha (Alexandr) Polujan, who always believed in me and encouraged me to go on. I am looking forward to having you here in Rostock.

I thank my family, my parents, my siblings, my grandparents and all of my friends who supported me through all the years. Especially my brother Ruben who shares my love for mathematics and who always was eager to hear about my latest research. I enjoy every mathematical discussion with you. Additionally, I want to mention Myriam, who makes me strong and is simply the best. Last but not least I thank Sven for being there for me every single day.

Contents

Notation

In the following table we present the most used notation in this thesis. It is meant as a lookup table for later reference. All parameters are properly defined in the chapters to come.

Notation	Definition
q	a prime power
$\mathbb{F}_q$	a finite field of size q
$\mathrm{char}(\mathbb{F}_q)$	the characteristic of the finite field
$\overline{\mathbb{F}}_q$	the algebraic closure of $\mathbb{F}_q$
$\mathbb{F}_q^*$	$= \mathbb{F}_q \backslash \{0\}$, the multiplicative group of $\mathbb{F}_q$
a	an element of $\mathbb{F}_q$
$\mathrm{ord}(a)$	the multiplicative order of $a \in \mathbb{F}_q^*$
$\mathbb{F}_q(\alpha)$	the smallest finite field $\mathbb{F}$ such that $\mathbb{F}_q \cup \{\alpha\} \subseteq \mathbb{F}$, where $\alpha \in \mathbb{F}_{q^k}$ for a positive integer k
$m_{\alpha,q} = m_\alpha$	the minimal polynomial of α over $\mathbb{F}_q$, where $\alpha \in \mathbb{F}_{q^k}$ for a positive integer k
$\chi_{\alpha,q} = \chi_\alpha$	$= (m_{\alpha,q})^{\frac{k}{d}}$, the characteristic polynomial of $\alpha \in \mathbb{F}_{q^k}$ over $\mathbb{F}_q$, where k and d are positive integers such that $d \mid k$ and α is a proper element of $\mathbb{F}_{q^d}$
$[\mathbb{F}_{q^k} : \mathbb{F}_q]$	$= k$, the degree of the extension $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$, the dimension of the $\mathbb{F}_q$-vector space $\mathbb{F}_{q^k}$
f	a polynomial over $\mathbb{F}_q$, an element of $\mathbb{F}_q[X]$, often monic and irreducible
$\deg(f)$	the degree of a polynomial f, often denoted by k
$\mathrm{ord}(f)$	the order of f the smallest positive integer e such that $f \mid X^e - 1$
e	the order of a polynomial $f \in \mathbb{F}_q[X]$ or the order of $a \in \mathbb{F}_q^*$
$\mathrm{coeffdeg}_q(f)$	$= [\mathbb{F}_q(a_0, \ldots, a_k) : \mathbb{F}_q]$ for a polynomial $f = \sum_{i=0}^{k} a_i X^i$ over $\mathbb{F}_{q^d}$ for a positive integer d
$f^{(j)}$	$= \sum_{i=0}^{k} a_i^{q^j} X^i$ for a polynomial $f = \sum_{i=0}^{k} a_i X^i \in \mathbb{F}_{q^d}[X]$ and positive integers j and d
$\mathrm{spin}_q[f]$	$= \prod_{j=0}^{\mathrm{coeffdeg}_q(f)-1} f^{(j)}$

$\langle f \rangle$	the principal ideal generated by f in $\mathbb{F}_q[X]$
Q	a rational function over $\mathbb{F}_q$, $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ for $g, h \in \mathbb{F}_q[X]$, $h \neq 0$
f^Q	the rational transformation $h^k \cdot f(\frac{g}{h}) \in \mathbb{F}_q[X]$, where $f \in \mathbb{F}_q[X]$ with $\deg(f) = k$ and $Q = \frac{g}{h} \in \mathbb{F}_q(X)$
$\mathbb{N}$	$= \{1, 2, 3, 4, 5, 6, \ldots\}$, the set of the positive integers
n	a positive integer
U_n	the set of all roots of the polynomial $X^n - 1$ for a positive integer n
ζ_n	a primitive n-th root of unity, meaning: an element of U_n with $\mathrm{ord}(\zeta_n) = n$ for a positive integer n such that $\gcd(n, q) = 1$
Φ_n	the cyclotomic polynomial $\Phi_n = \prod_{\substack{0 \leq j \leq n-1 \\ \gcd(j,n)=1}} (X - \zeta_n^j)$ over $\mathbb{F}_q$ for a positive integer n such that $\gcd(n, q) = 1$
p	a prime number, not necessarily the characteristic of $\mathbb{F}_q$!
$\mathrm{rad}(n)$	the product of all distinct prime factors of n
$\nu_p(n)$	the p-adic valuation of a positive integer n
$[m]_n$	the residue class of m in $\mathbb{Z}/n\mathbb{Z}$ for an integer $m \in \mathbb{Z}$ such that $\gcd(m, n) = 1$
$m \bmod n$	the remainder r of the division of n by m, $n = m \cdot z + r$ for $z \in \mathbb{Z}$ and $0 \leq r \leq n - 1$, often we select $r = n$ instead of $r = 0$
$\mathrm{ord}_n(m)$	the multiplicative order of m in $\mathbb{Z}/n\mathbb{Z}$
φ	the Euler totient function
$d^{(t)}$	$= \gcd(n, q^t - 1)$ for positive integers n and t
${d_1}^{(t)}$	$= \gcd(n_1, \frac{q^t - 1}{\mathrm{ord}(\alpha)})$ for positive integers n and t and an element α of $\mathbb{F}_{q^t}$, where $n = n_1 \cdot n_2$ such that $\mathrm{rad}(n_1) \mid \mathrm{ord}(\alpha)$ and $\gcd(n_2, \mathrm{ord}(\alpha)) = 1$
${d_2}^{(t)}$	$= \gcd(n_2, q^t - 1)$ for positive integers n and t and an element α of $\mathbb{F}_{q^t}$, where $n = n_1 \cdot n_2$ such that $\mathrm{rad}(n_1) \mid \mathrm{ord}(\alpha)$ and $\gcd(n_2, \mathrm{ord}(\alpha)) = 1$
$\mathrm{C}_{q,d}(i)$	$= \{i \cdot q^j \pmod d : j \geq 0\}$ the q-cyclotomic coset of i modulo d for a positive integer d
$\mathrm{CR}_q(d)$	a representative system of the q-cyclotomic cosets modulo d

Chapter 1

Introduction

Polynomials are the first abstract mathematical object every school kid gets to know. Even though this might not be visible at first sight, the set of polynomials over a field $\mathbb{F}$,

$$\mathbb{F}[X] = \left\{ a_k X^k + a_{k-1} X^{k-1} + \ldots + a_1 X + a_0 : a_0, \ldots, a_k \in \mathbb{F}, k \in \mathbb{N} \cup \{0\} \right\},$$

hides a beautiful structure. This structure is similar to the ring of integers $\mathbb{Z}$ which we all learned to calculate with in primary school. The polynomials form a commutative ring and also a vector space over $\mathbb{F}$. This allows us to add and multiply polynomials with other polynomials or elements of the field $\mathbb{F}$. In school these operations are used to solve polynomial equations of degree less than or equal to two. We say that a polynomial $g \in \mathbb{F}[X]$ *divides* $f \in \mathbb{F}[X]$, or, equivalently, g is a *factor of* f, if there exists a polynomial $h \in \mathbb{F}[X]$ such that $f = g \cdot h$. In fact, we can even divide f by g with remainder, meaning that there exist polynomials $q, r \in \mathbb{F}[X]$ such that $0 \leq \deg(r) < \deg(g)$ and $f = q \cdot g + r$. In high school, we learn to determine the polynomials q and r with the polynomial long division algorithm, an analogue of the long division algorithm in the ring of integers. In fact, the division with remainder in $\mathbb{F}[X]$ is an analogue of the Euclidean division in the ring of integers since both $\mathbb{F}[X]$ and $\mathbb{Z}$ are so-called *Euclidean domains*.

Many mathematicians will agree that the most fascinating elements in the ring of integers are the prime numbers, the natural numbers $p \geq 2$ whose only factors are 1 and p itself. Even though Euclid's famous and elegant proof (circa 300BC) tells us that there exist infinitely many prime numbers, they are hard to find. In practice, it also is difficult to determine whether a given positive integer is prime or not. The prime numbers have been studied for more than two thousand years and still capture the attention of mathematicians all over the world.

The prime number's equivalent in the Euclidean domain of the polynomials are the monic irreducible polynomials. A polynomial $f = a_k X^k + a_{k-1} X^{k-1} + \ldots + a_1 X + a_0 \in \mathbb{F}[X]$ of degree k is called *monic* if $a_k = 1$. Furthermore, f is *irreducible* if $\deg(f) \geq 1$ and for every $g, h \in \mathbb{F}[X]$ such that $f = g \cdot h$ holds $\deg(g) = 0$ or $\deg(h) = 0$. From the definition follows immediately that every polynomial of degree 1 is irreducible. A polynomial f of degree 2 or 3 is irreducible if and only if there does not exist a *root* α of f in $\mathbb{F}$, meaning an element α satisfying $f(\alpha) = 0$. This is due to the fact that any element $\alpha \in \mathbb{F}$ is a root of f if and only if the polynomial $X - \alpha$ divides f. Things become more complicated for polynomials of a higher degree. Obviously, a polynomial $f \in \mathbb{F}[X]$ of degree ≥ 4 which has a root α in $\mathbb{F}$ is reducible, because then $X - \alpha$ is a factor of f. However, if f does not have a root in $\mathbb{F}$ it is in general challenging to determine whether it is irreducible or not.

For every integer n there exists a product $n = u \cdot p_1^{l_1} \cdots p_m^{l_m}$ which is unique up to reordering, the *prime factorization of* n, where $u \in \{-1, 1\}$ is a unit in $\mathbb{Z}$, $p_1, \ldots, p_m$ are distinct prime

numbers and $l_1, \ldots, l_m \in \mathbb{N}$. Then the product $p_1 \cdot p_2 \cdots p_m$ is called the *radical of* n and denoted by $\mathrm{rad}(n)$. Furthermore, for any $1 \leq i \leq m$, the *p_i-adic valuation of* n, $\nu_{p_i}(n)$, is the exponent l_i. For any other prime p such that $p \nmid n$, we set $\nu_p(n) = 0$.

Like the prime factorization in the ring of integers, every polynomial $f \in \mathbb{F}[X]$ of degree ≥ 1 can be described as a product $a \cdot g_1^{l_1} \cdots g_m^{l_m}$, which is unique up to reordering, of a field element $a \in \mathbb{F}\backslash\{0\}$ and m distinct monic irreducible polynomials $g_1, \ldots, g_m \in \mathbb{F}[X]$, $l_1, \ldots, l_m \in \mathbb{N}$. We call this product the *(unique) factorization of f (into monic irreducible polynomials over $\mathbb{F}$)*. Finding this factorization for a given polynomial is at least as difficult as determining whether the polynomial is irreducible or not, because f is irreducible if and only if the factorization of f equals $a \cdot g_1$ for a monic irreducible polynomial $g_1 \in \mathbb{F}[X]$ and a field element $a \in \mathbb{F} \setminus \{0\}$. Without loss of generality we can assume that f is monic. Indeed, if $a_k \neq 1$, then the polynomial $\tilde{f} = a_k^{-1} \cdot f$ is monic and the factorization of f equals $a_k \cdot g_1^{l_1} \cdots g_m^{l_m}$, where $g_1^{l_1} \cdots g_m^{l_m}$ is the factorization of $\tilde{f}$.

The factorization of a polynomial yields a lot of information about its roots. Let f be a monic polynomial over $\mathbb{F}$ of degree k. Then f has a monic irreducible factor $X - b \in \mathbb{F}[X]$ of degree 1 if and only if $b \in \mathbb{F}$ is a root of f. These factors are called *linear factors*. There exists an extension field $\mathbb{E}$ of $\mathbb{F}$ such that the factorization of f over $\mathbb{E}$ consists only of linear factors. More precisely, there exist $\beta_1, \ldots, \beta_m \in \mathbb{E}$ such that $f = (X-\beta_1)^{l_1} \cdots (X-\beta_m)^{l_m}$, where $l_1, \ldots, l_m \in \mathbb{N}$. Thus, $\beta_1, \ldots, \beta_m$ are the roots of f with multiplicities $l_1, \ldots, l_m$. Moreover, if f has an irreducible factor g of degree greater than 1, then there exists a root α of g in an extension field $\mathbb{E}$ of $\mathbb{F}$ with $\deg(g) = [\mathbb{E} : \mathbb{F}]$. Here $[\mathbb{E} : \mathbb{F}]$ is the dimension of $\mathbb{E}$ seen as a vector space over $\mathbb{F}$. Since α is a root of g and g is a factor of f, α is a root of f.

The irreducibility and the factorization of f also determine the structure of the quotient ring

$$\mathbb{F}[X]/\langle f \rangle = \{r + \langle f \rangle : r \in \mathbb{F}[X]\} = \{[r] : r \in \mathbb{F}[X]\},$$

where $\langle f \rangle = \{f \cdot g : g \in \mathbb{F}[X]\}$ is the principal ideal generated by f. By the Euclidean division in $\mathbb{F}[X]$ the elements of $\mathbb{F}[X]/\langle f \rangle$ are in fact the residue classes $[r] = r + \langle f \rangle$ of the polynomials $r \in \mathbb{F}[X]$ with $0 \leq \deg(r) < \deg(f) = k$.

The quotient ring $\mathbb{F}[X]/\langle f \rangle$ is a field if and only if f is irreducible. Since for every $a \in \mathbb{F}$, there exists the residue class $[a]$ in $\mathbb{F}[X]/\langle f \rangle$, the quotient ring $\mathbb{F}[X]/\langle f \rangle$ can be viewed as an extension of the field $\mathbb{F}$. The best known example for this construction are the complex numbers:

$$\mathbb{C} := \mathbb{R}[X]/\langle X^2 + 1 \rangle = \{[a + bX] : a, b \in \mathbb{R}\}.$$

Let $f = X^k + a_{k-1}X^{k-1} + \ldots + a_1 X + a_0$, where $a_0, \ldots, a_{k-1} \in \mathbb{F}$. From the fact that $[0] = [f] = [X^k + a_{k-1}X^{k-1} + \ldots + a_1 X + a_0]$, we obtain that $[X^k] = [-a_{k-1}X^{k-1} - \ldots - a_1 X - a_0]$ and all computations in $\mathbb{F}[X]/\langle f \rangle$ are simplified with this identity. In the complex numbers this is the identity $[X^2] = [X]^2 = [-1]$. The number of operations during the simplification is determined by the number of terms in the polynomial $-a_{k-1}X^{k-1} - \ldots - a_1 X - a_0$, and therefore by the number of terms in f. The number of terms, or more precisely, the number $|\{0 \leq j \leq k \mid a_j \neq 0\}|$ of non-zero coefficients of the polynomial f, is called the *weight of f*. For an efficient computation in the extension field $\mathbb{F}[X]/\langle f \rangle$ it is of great interest to find irreducible polynomials of a low weight. Polynomials of a low weight are also called *sparse polynomials*. They are difficult to find, because the number of sparse polynomials in general is much smaller than that of polynomials with a higher weight.

Note that the element $[X]$ is a root of f in $\mathbb{F}[X]/\langle f \rangle$ because $f([X]) = \sum_{j=0}^{k} a_j [X]^j = [\sum_{j=0}^{k} a_j X^j] = [f] = [0]$. We can write $\alpha := [X]$ and

$$\mathbb{F}[X]/\langle f \rangle = \left\{ b_0 + b_1\alpha + \ldots + b_{k-1}\alpha^{k-1} : b_0, \ldots, b_{k-1} \in \mathbb{F} \right\}, \tag{1.1}$$

where α satisfies $f(\alpha) = 0$. In the complex numbers the root $[X]$ of $X^2 + 1$ is called i, the imaginary unit, which satisfies $i^2 = -1$. With (1.1) we obtain the familiar description of the complex numbers:

$$\mathbb{C} = \{a + bi : a, b \in \mathbb{R}\}.$$

If f is not irreducible, then the factorization $g_1^{l_1} \cdots g_m^{l_m}$ of f into monic irreducible polynomials over $\mathbb{F}$ determines the structure of the quotient ring $\mathbb{F}[X]/\langle f\rangle$. The principal ideal of f satisfies: $\langle f\rangle = \bigcap_{i=1}^{m} \langle g_i^{l_i}\rangle$ and $\langle g_i^{l_i}\rangle + \langle g_j^{l_j}\rangle = \langle 1\rangle$ for all $i \neq j$. Consequently, the Chinese Remainder Theorem implies that:

$$\mathbb{F}[X]/\langle f\rangle = \mathbb{F}[X]/\langle g_1^{l_1}\rangle \ \times \ \ldots \ \times \ \mathbb{F}[X]/\langle g_m^{l_m}\rangle. \tag{1.2}$$

Furthermore, every ideal in $\mathbb{F}[X]/\langle f\rangle$ is defined by a factor of f as follows: $\mathbb{F}[X]/\langle f\rangle$ is a principal ideal domain, because for every nonzero ideal $\mathcal{I} \subseteq \mathbb{F}[X]/\langle f\rangle$ there exists a unique monic polynomial $g \neq 0$ of minimal degree, $\deg(g) < \deg(f)$, such that $[g] \in \mathcal{I}$. Indeed, for every element $[h] \in \mathcal{I}$ there exist polynomials $q, r \in \mathbb{F}[X]$ with $\deg(r) < \deg(g)$ such that $h = q \cdot g + r$. Since $\mathcal{I}$ is an ideal, we obtain that $[r] = [h] - [q] \cdot [g] \in \mathcal{I}$ and from the minimality of g follows that $r = 0$. Thus, $\mathcal{I} = \langle [g]\rangle$. The residue class [1] is an element of $\mathcal{I} = \langle [g]\rangle$ if and only if $\gcd(g, f) = 1$. Consequently, $\mathcal{I}$ is a proper ideal if and only if $\gcd(g, f) = d$ for a polynomial $d \in \mathbb{F}[X]$ of degree $\deg(d) \geq 1$. Then there exist polynomials $q_1, q_2 \in \mathbb{F}[X]$ such that $d = q_1 \cdot g + q_2 \cdot f$. Thus, $[d] \in \langle [g]\rangle$ which implies that $d = g$ and g is a factor of f.

Then the set of ideals in $\mathbb{F}[X]/\langle f\rangle$ is

$$\{\mathcal{I} \subseteq \mathbb{F}[X]/\langle f\rangle : \mathcal{I} \text{ ideal}\} = \{\langle [g_1^{i_1} \cdots g_m^{i_m}]\rangle : i_1 \leq l_1, \ldots, i_m \leq l_m\}, \tag{1.3}$$

where $f = g_1^{l_1} \cdots g_m^{l_m}$ is the factorization of f into monic irreducible factors over $\mathbb{F}$.

In this thesis we concentrate on polynomials over finite fields. From the fact that there are only finitely many elements in a finite field follow many interesting properties which we briefly present here.

1.1 Crash Course: Finite Fields

There exists a finite field $\mathbb{F}_q$ with q elements if and only if $q = p^m$ for a prime p and a positive integer m. If $m = 1$, then $\mathbb{F}_q \cong \mathbb{Z}/p\mathbb{Z}$. Otherwise, there exists an irreducible polynomial $f \in \mathbb{F}_p[X]$ such that $\mathbb{F}_q \cong \mathbb{F}_p[X]/\langle f\rangle$. All finite fields of size q are isomorphic and the characteristic $\mathrm{char}(\mathbb{F}_q)$ of $\mathbb{F}_q$ equals p. The fact that $\mathrm{char}(\mathbb{F}_q) = p$ implies that for any $a, b \in \mathbb{F}_q$ holds $(a+b)^p = a^p + b^p$. Furthermore, the multiplicative group $\mathbb{F}_q^* = \mathbb{F}_q \setminus \{0\}$ is a cyclic group generated by a so-called *primitive element* $b \in \mathbb{F}_q^*$ of order $q - 1$ such that $\langle b\rangle = \mathbb{F}_q^*$. From this fact follows that $a^q = a$ and $\mathrm{ord}(a) \mid q - 1$ for every $a \in \mathbb{F}_q^*$, where $\mathrm{ord}(a)$ denotes the *order of* a in the multiplicative group $\mathbb{F}_q^*$.

We denote by $\varphi : \mathbb{N} \to \mathbb{N}$ Euler's totient function, which is defined as

$$\varphi(n) = |\{1 \leq j \leq n : \gcd(n, j) = 1\}|$$

for any positive integer n. Recall that in every cyclic group there exist exactly $\varphi(d)$ elements of order d for every divisor d of the group's order. Consequently, there exist exactly $\varphi(q - 1)$ primitive elements in $\mathbb{F}_q$. For any positive integer n we further denote by $[m]_n$ the residue class $m + \langle n\rangle$ of $m \in \mathbb{Z}$ in the quotient ring $\mathbb{Z}_n := \mathbb{Z}/n\mathbb{Z} = \mathbb{Z}/\langle n\rangle$. Then $\mathbb{Z}_n^*$, the multiplicative group in $\mathbb{Z}_n$, has order $\varphi(n)$ and we denote by $\mathrm{ord}_n(m)$ the multiplicative order of $[m]_n$ in $\mathbb{Z}_n^*$.

For every positive integer n such that $\gcd(n, q) = 1$, there exists an element ζ_n of order n in the extension field $\mathbb{F}_{q^{\mathrm{ord}_n(q)}}$ of $\mathbb{F}_q$. The element ζ_n is a generator of the multiplicative

group U_n of the roots of the polynomial $X^n - 1$ and we call it a *primitive n-th root of unity*. For any other element $a \in \mathbb{F}_q^* \setminus \{1\}$ there exists a root $b \in \mathbb{F}_q$ of the polynomial $X^n - a$, such that $b^n = a$, if and only if $a^{\frac{q-1}{\gcd(q-1,n)}} = 1$. If $b^n = a$, then the order of a satisfies $\mathrm{ord}(a) = \frac{\mathrm{ord}(b)}{\gcd(\mathrm{ord}(b),n)}$.

Let $f \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k over $\mathbb{F}_q$ and α be a root of f, that is $f(\alpha) = 0$. Then f is called the *minimal polynomial of α over $\mathbb{F}_q$*, which we denote by $m_{\alpha,q}$, and its degree k equals $\mathrm{ord}_{\mathrm{ord}(\alpha)}(q)$. For every other polynomial $g \in \mathbb{F}_q[X]$ with $g(\alpha) = 0$ holds $f = m_{\alpha,q} \mid g$. Furthermore,

$$f = (X - \alpha) \cdot (X - \alpha^q) \cdots (X - \alpha^{q^{k-1}}),$$

and the elements α^{q^j} for $0 \leq j \leq k-1$ are called the $\mathbb{F}_q$*-conjugates* of α. For every $1 \leq j \leq k-1$ the order of α^{q^j} equals the order of α. The order $\mathrm{ord}(g)$ of any polynomial $g \in \mathbb{F}_q[X]$ with $g(0) \neq 0$ is defined as the least positive integer e such that g divides $X^e - 1$. Interestingly, for an irreducible polynomial $f \in \mathbb{F}_q[X]$ of degree k with a root α holds $\mathrm{ord}(f) = \mathrm{ord}(\alpha) = \mathrm{ord}(\alpha^{q^j})$ for all $0 \leq j \leq k-1$. Because of this, we can call the polynomial f a *primitive polynomial* when α is a primitive element or, equivalently, when $\mathrm{ord}(\alpha) = q^k - 1$.

Since α is a root of the monic irreducible polynomial f of degree k, α is an element of the extension field $\mathbb{F}_{q^k}$ and there does not exist a subfield $\mathbb{F}_q \leq \mathbb{F} \leq \mathbb{F}_{q^k}$ such that $\alpha \in \mathbb{F}$. Therefore, we call α a *proper element of* $\mathbb{F}_{q^k}$. By $\mathbb{F}_q(\alpha)$ we denote the smallest field such that $\mathbb{F}_q \cup \{\alpha\} \subseteq \mathbb{F}_q(\alpha)$. Thus, $\mathbb{F}_q(\alpha) = \mathbb{F}_{q^k} \cong \mathbb{F}_q[X]/\langle f \rangle$ and $[\mathbb{F}_q(\alpha) : \mathbb{F}_q] = k$. The algebraic closure of the finite field $\mathbb{F}_q$, the extension field which contains all roots of all polynomials over $\mathbb{F}_q$, is denoted by $\overline{\mathbb{F}}_q$. If $\alpha \in \overline{\mathbb{F}}_q$ is an element of $\mathbb{F}_{q^d}$ for a positive integer d, then the *characteristic polynomial $\chi_{\alpha,q}$ of $\alpha \in \mathbb{F}_{q^d}$ over $\mathbb{F}_q$* is defined as $\prod_{j=0}^{d-1}(X - \alpha^{q^j})$. The characteristic polynomial satisfies $\chi_{\alpha,q} = (m_{\alpha,q})^{d/k}$, where $k = \deg(m_{\alpha,q})$.

Every finite field $\mathbb{F}_{q^m}$ of size q^m forms a vector space of dimension m over $\mathbb{F}_q$. If $(\alpha_1, \ldots, \alpha_m)$ is a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, then every element $a_1 \cdot \alpha_1 + \ldots + a_m \cdot \alpha_m$ in $\mathbb{F}_{q^m}$ can be denoted by its coefficient vector $(a_1, \ldots, a_m) \in \mathbb{F}_q^m$. Many applications call for a simultaneous use of $\mathbb{F}_{q^m}$ as an extension field and as a vector space over $\mathbb{F}_q$. This can easily be realized if the finite field $\mathbb{F}_{q^m}$ is constructed as $\mathbb{F}_q[X]/\langle f \rangle$ with a so-called normal polynomial f. A monic irreducible polynomial of degree m with a root α is called *normal* or *N-polynomial* if its roots $\alpha, \alpha^q \ldots, \alpha^{q^{m-1}}$ are linearly independent over $\mathbb{F}_q$ or, equivalently, if the degree of the greatest common divisor of the polynomials $g_\alpha = \alpha X^{m-1} + \alpha^q X^{m-2} + \ldots + \alpha^{q^{m-2}} X + \alpha^{q^{m-1}}$ and $X^m - 1$ over $\mathbb{F}_{q^m}$ is 0. If f is a normal polynomial, then the elements $[X], [X]^q, \ldots, [X]^{q^{m-1}}$ form a normal basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, because the element $[X]$ is a root of f in $\mathbb{F}_q/\langle f \rangle = \mathbb{F}_{q^m}$. As a generalization of the concept of normality, a polynomial is called *k-normal* if the greatest common divisor of g_α and $X^m - 1$ has degree k or, equivalently, if the dimension of the vector space generated by $\{\alpha, \alpha^q, \ldots, \alpha^{q^{m-1}}\}$ over $\mathbb{F}_q$ has dimension $n - k$.

The Lagrange Interpolation Formula implies that for every map $\phi : \mathbb{F}_q \to \mathbb{F}_q$ there exists a polynomial $f \in \mathbb{F}_q[X]$ such that $f(a) = \phi(a)$ for all $a \in \mathbb{F}_q$. Thus, polynomials over finite fields describe all maps over finite fields. We denote by $\phi_f : \mathbb{F}_q \to \mathbb{F}_q$ the map defined as $\phi_f(a) = f(a)$ for all $a \in \mathbb{F}_q$. From the identity $a^q = a$ for all $a \in \mathbb{F}_q$ follows that $\phi_{X^q} = \phi_X$. Consequently, for every polynomial $f \in \mathbb{F}_q[X]$ there exists a polynomial $g \in \mathbb{F}_q[X]$ of degree strictly smaller than q such that $\phi_f = \phi_g$. Therefore, all maps over a finite field can be described by the polynomials of degrees smaller than q:

$$\{\phi : \mathbb{F}_q \to \mathbb{F}_q\} = \{\phi_f \mid f \in \mathbb{F}_q[X], \deg(f) < q\}.$$

Thus, the polynomials over a finite field play a much more important role than the polynomials over an infinite field.

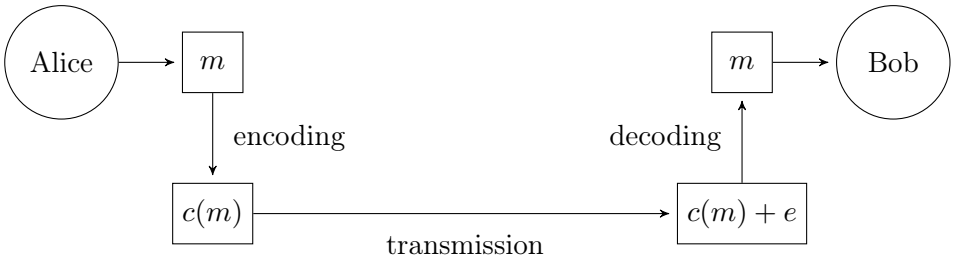
A generalization of the polynomials are the rational functions. Let g and h be polynomials over $\mathbb{F}_q$, where $h \neq 0$, then $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ is called a *rational function over* $\mathbb{F}_q$ and its degree is defined as $\max\{\deg(g), \deg(h)\}$. For any polynomial $f \in \mathbb{F}_q[X]$ of degree k the polynomial $f^Q := h^k \cdot f\left(\frac{g}{h}\right) \in \mathbb{F}_q[X]$ is called the *Q-transform* of f. Rational transformations are often used to construct irreducible polynomials (see Section 1.2.3).

Remark 1.1. In many publications the character p is exclusively used for the characteristic of the finite field $\mathbb{F}_q$ of $q = p^m$ elements, where $m \in \mathbb{N}$. Since we mostly concentrate on positive integers n such that $\gcd(n, q) = 1$, we do not use $\mathrm{char}(\mathbb{F}_q)$ often. Therefore, in this thesis p is simply a prime number. We state explicitly whenever p denotes the characteristic $\mathrm{char}(\mathbb{F}_q)$ of $\mathbb{F}_q$.

1.2 Irreducible polynomials in coding and cryptography

1.2.1 Irreducible polynomials in cyclic and constacyclic codes

Almost all modern error-correcting codes and cryptographic ciphers are defined on bits and bytes, elements of the finite field $\mathbb{F}_2$, the vector space $\mathbb{F}_2^8$ or $\mathbb{F}_2$-vector spaces of a higher dimension. Error-correcting codes are used for the transmission of data through noisy channels. Suppose that a sender (Alice) is sending a message $m \in \mathbb{F}_2^n$ to a receiver (Bob) and some bits of this message are not transmitted correctly. That is, an error $e \in \mathbb{F}_2^n$ occurs so that Bob receives the message $m + e$ instead of m. Probably Bob will not guess correctly what the message m might have been. To avoid this, Marcel J. E. Golay and Richard Hamming started to work on the first error correcting codes at the end of the 1940s [Ham50]. These codes are designed to add redundancy to the message m so that even if some bits of m are changed by the error e, Bob is still able to decode the received message correctly:



A *code* $\mathcal{C}$ is simply a subset of $\mathbb{F}_q^n$. One of the most studied types of codes are the cyclic codes. A code $\mathcal{C}$ is called *cyclic* if it is an $\mathbb{F}_q$-subspace of $\mathbb{F}_q^n$ and for every code word $(c_0, \ldots, c_{n-1}) \in \mathcal{C}$ the shifted word $(c_{n-1}, c_0, \ldots, c_{n-2})$ is an element of $\mathcal{C}$ as well. Any element $(c_0, \ldots, c_{n-1})$ of $\mathbb{F}_q^n$ can be interpreted as the coefficient vector of the element $[c_0 + c_1 X + \ldots + c_{n-1} X^{n-1}]$ in the quotient ring $\mathbb{F}_q[X]/\langle X^n - 1\rangle$. Then multiplication with $[X]$ yields the residue class $[c_{n-1} + c_0 X + \ldots + c_{n-2} X^{n-1}]$ which corresponds to the code word $(c_{n-1}, c_0, \ldots, c_{n-2})$. By abuse of notation, we write $\mathcal{C} \subseteq \mathbb{F}_q[X]/\langle X^n - 1\rangle$ and $\mathcal{C}$ is a cyclic code if and only if $\mathcal{C}$ is an $\mathbb{F}_q$-subspace and $[X] \cdot c \in \mathcal{C}$ for all $c \in \mathcal{C}$. This is equivalent to being an ideal in $\mathbb{F}_q[X]/\langle X^n - 1\rangle$. Suppose that the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$ is given by $g_1^{l_1} \cdots g_m^{l_m}$, where $g_1, \ldots, g_m$ are pairwise distinct monic irreducible polynomials over $\mathbb{F}_q$ and $l_1, \ldots, l_m \in \mathbb{N}$. Then with (1.3) every ideal in $\mathbb{F}_q[X]/\langle X^n - 1\rangle$ is of the form $\langle g_1^{i_1} \cdots g_m^{i_m}\rangle$, where $i_1 \leq l_1, \ldots, i_m \leq l_m$. The polynomial $g_1^{l_1} \cdots g_m^{l_m}$ is called the *generating polynomial* of the cyclic code $\mathcal{C}$. In order to find all cyclic codes in $\mathbb{F}_q^n$ it suffices to factor the polynomial $X^n - 1 \in \mathbb{F}_q[X]$.

Without loss of generality we can restrict ourselves to the positive integers n satisfying $\gcd(n, q) = 1$. Indeed, if $n = \mathrm{char}(\mathbb{F}_q)^l \cdot \tilde{n}$ for positive integers l and $\tilde{n}$ such that $\gcd(\tilde{n}, q) = 1$, then $X^n - 1 = X^{\mathrm{char}(\mathbb{F}_q)^l \cdot \tilde{n}} - 1 = (X^{\tilde{n}} - 1)^{\mathrm{char}(\mathbb{F}_q)^l}$ and we can study the factorization of

the polynomial $X^{\tilde{n}} - 1$ instead. Thus, let n be a positive integer such that $\gcd(n, q) = 1$. Then there exists a primitive n-th root of unity ζ_n in $\overline{\mathbb{F}}_q$ such that $U_n = \langle \zeta_n \rangle$ and $X^n - 1$ decomposes over $\overline{\mathbb{F}}_q$ as

$$X^n - 1 = \prod_{\zeta \in U_n} (X - \zeta) = \prod_{j=0}^{n-1} (X - \zeta_n^j). \tag{1.4}$$

If n divides $q-1$, then $\zeta_n \in \mathbb{F}_q$ and (1.4) is the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$. Otherwise, the factorization of $X^n - 1$ into monic irreducible polynomials over $\mathbb{F}_q$ consists of the minimal polynomials of the elements ζ_n^j, where $0 \leq j \leq n-1$.

The factorization of the polynomial $X^n - 1$ over finite fields has been studied since the 19th century. Carl Friedrich Gauss [Gau76; Gau89] was among the first to study the factors of $X^n - 1$ which are called cyclotomic polynomials over prime fields. For a positive integer n such that $\gcd(q, n) = 1$ the *n-th cyclotomic polynomial* Φ_n is defined as

$$\Phi_n = \prod_{\substack{0 \leq j \leq n-1 \\ \gcd(j,n)=1}} (X - \zeta_n^j).$$

Thus, Φ_n is the polynomial whose roots are the $\varphi(n)$ distinct primitive n-th roots of unity over $\mathbb{F}_q$ and

$$X^n - 1 = \prod_{d|n} \Phi_d.$$

Consequently, if the factorization of Φ_d over $\mathbb{F}_q$ is known for every divisor d of n, these factorizations yield the factorization of $X^n - 1$ over $\mathbb{F}_q$. Vice versa, from a complete factorization of $X^n - 1$ the factorization of all cyclotomic polynomials Φ_d with d dividing n can be derived, if the order of the irreducible factors is known.

The cyclotomic polynomial Φ_n can be described as a rational function over $\mathbb{F}_q$ using the Möbius inversion formula:

$$\Phi_n = \prod_{d|n} (X^d - 1)^{\mu(\frac{n}{d})} \in \mathbb{F}_q(X), \tag{1.5}$$

where μ is the *Möbius function*. The Möbius function is defined for every positive integer n with prime factorization $n = p_1^{l_1} \cdots p_m^{l_m}$ as

$$\mu(n) = \begin{cases} 1 & \text{if } l_1 = \ldots = l_m = 1 \text{ and } m \text{ is even,} \\ -1 & \text{if } l_1 = \ldots = l_m = 1 \text{ and } m \text{ is odd,} \\ 0 & \text{otherwise.} \end{cases}$$

Equation (1.5) is used in computer algebra systems for a fast computation of the factorization of Φ_n over $\mathbb{F}_q$. However, the computation involves polynomial division as (1.5) does not give the factorization of Φ_n over $\mathbb{F}_q$ explicitly.

Though partial results existed, which we present in Section 2.1, the explicit factorization of either $X^n - 1$ or the cyclotomic polynomial Φ_n for arbitrary positive integers n was still unknown.

The concept of cyclic codes has been extended in 1968 by Berlekamp to so-called constacyclic codes [Ber15, p.303]. A *constacyclic code* or an *a-constacyclic code $\mathcal{C}$ of length n* is an ideal in the quotient ring $\mathbb{F}_q[X]/\langle X^n - a \rangle$, where a is an element of $\mathbb{F}_q^*$. This is equivalent to $\mathcal{C}$ being an $\mathbb{F}_q$-subspace of $\mathbb{F}_q^n$ with the property that for every $(c_0, \ldots, c_{n-1}) \in \mathcal{C}$ the element $(a \cdot c_{n-1}, c_0, \ldots, c_{n-2})$ is a code word in $\mathcal{C}$ as well. Again with (1.3) all constacyclic codes are defined by the factorization of the polynomial $X^n - a$ over $\mathbb{F}_q$.

A constacyclic code of length n is called *simple-root* if $\gcd(n, q) = 1$ and *repeated-root* if $\gcd(n, q) > 1$. The code properties of repeated-root and simple-root codes differ, but this distinction is irrelevant for the determination of the generating polynomials. This is due to the fact that the factorization of $X^n - a$ for all positive integers n such that $\gcd(n, q) > 1$ can easily be derived from the factorization of $X^n - a$ for the positive integers satisfying $\gcd(n, q) = 1$, as the following remark shows.

Remark 1.2. Suppose that $n = \tilde{n} \cdot \mathrm{char}(\mathbb{F}_q)^l$, where $\gcd(\tilde{n}, q) = 1$ and $l \in \mathbb{N}$, then there exists an element $b \in \mathbb{F}_q$ such that $a = b^{\mathrm{char}(\mathbb{F}_q)^l}$. If the factorization of $X^{\tilde{n}} - b$ into monic irreducible factors over $\mathbb{F}_q$ is given by $g_1^{l_1} \cdots g_m^{l_m}$, then

$$X^n - a = X^{\tilde{n} \cdot \mathrm{char}(\mathbb{F}_q)^l} - b^{\mathrm{char}(\mathbb{F}_q)^l} = \left(X^{\tilde{n}} - b\right)^{\mathrm{char}(\mathbb{F}_q)^l}$$

factors as $g_1^{l_1 \cdot \mathrm{char}(\mathbb{F}_q)^l} \cdots g_m^{l_m \cdot \mathrm{char}(\mathbb{F}_q)^l}$.

In 1866 Joseph Alfred Serret gave an irreducibility criterion for the polynomial $X^n - a$ over prime fields which can be adapted for arbitrary finite fields $\mathbb{F}_q$ (see Theorem 2.22). The study of the explicit factorization of $X^n - a$ for $a \neq 1$ over finite fields started in the 1990s for $a = -1$ and $n = 2^l$ for a positive integer l [BGM93; Mey96].

In 2012 Chen, Fan, Lin and Liu [Che+12] wrote

> *"It is remarkable that, though all irreducible binomials over $\mathbb{F}_q$ have been explicitly characterized by Serret early in 1866, no effective method was found to characterize the irreducible factors of $X^n - a$ over $\mathbb{F}_q$ so far. It is a challenge to determine explicitly the polynomial generators of all constacyclic codes over finite fields."*

In 2023, though partial results existed, which we present in Section 2.2, the explicit factorization of $X^n - a$ over $\mathbb{F}_q$ for arbitrary elements $a \in \mathbb{F}_q^*$ and arbitrary positive integers n was still unknown.

1.2.2 Overview of Chapter 2

In Chapter 2 we give **one closed explicit formula each** for the factorization of

- $X^n - 1$, (Theorem 2.47)
- Φ_n, (Theorem 2.49)
- $X^n - a$ for arbitrary $a \in \mathbb{F}_q^*$, (Theorem 2.41)
- $f(X^n)$ for arbitrary irreducible polynomials $f \in \mathbb{F}_q[X]$, $f \neq X$, (Theorem 2.61)

over $\mathbb{F}_q$ **for arbitrary positive integers** n such that $\gcd(n, q) = 1$.

Our main theorem, Theorem 2.41, is a closed explicit formula for the factorization of $X^n - a$ into monic irreducible polynomials over $\mathbb{F}_q$ for arbitrary $a \in \mathbb{F}_q^*$ and arbitrary positive integers n satisfying $\gcd(n, q) = 1$. With our main theorem and Remark 1.2 we determine the generating polynomials of all constacyclic codes over finite fields (Corollary 2.45).

As a direct consequence of our main theorem we obtain a closed explicit formula for the factorization of $X^n - 1$ over $\mathbb{F}_q$ for arbitrary positive integers n satisfying $\gcd(n, q) = 1$, Theorem 2.47. Thus, we determine the generating polynomials of all cyclic codes over finite fields (Corollary 2.48).

From Theorem 2.47 we derive Theorem 2.49, a closed explicit formula for the factorization of the n-th cyclotomic polynomial Φ_n for arbitrary positive integers n such that $\gcd(n, q) = 1$.

Since $X^n - a$ is a composition of the form $f(X^n)$ for the monic irreducible polynomial $f = X - a \in \mathbb{F}_q[X]$ one might think that we derive the factorization of $X^n - a$ from

our formula for the factorization of $f(X^n)$. However, instead we use a result by Mullin, Mullen and Yucas, Theorem 2.8, to derive Theorem 2.61, a closed explicit formula for the factorization of the composition $f(X^n)$ for arbitrary monic irreducible polynomials $f \in \mathbb{F}_q[X], f \neq X$, and arbitrary positive integers n satisfying $\gcd(n, q) = 1$, from our main theorem. The connection between the two polynomials $X^n - a$ and $f(X^n)$ is discussed and explained in the beginning of Chapter 2.

The formula given in Theorem 2.61 can be translated into an algorithm, Algorithm 2, for the computation of the factorization of $f(X^n)$ over $\mathbb{F}_q$ for arbitrary positive integers n and arbitrary irreducible polynomials $f \in \mathbb{F}_q[X]$. Our naive implementation of this algorithm in Python is faster and much more stable than the existing factoring functions in the computer algebra systems SageMath and Magma for many positive integers n. In particular, for large positive integers n our algorithm allows us to factor polynomials which cannot be factored with either SageMath or Magma.

Chapter 2 is structured as follows:

Section 2.1 We list all positive integers n for which explicit factorizations of the polynomial $X^n - 1$ and the cyclotomic polynomial Φ_n over $\mathbb{F}_q$ were known before (see Table 2.1 and Table 2.2).

Section 2.2 We list all positive integers n for which explicit factorizations of the polynomial $X^n - a$ for arbitrary $a \in \mathbb{F}_q^*$ were known before (see Table 2.4).

Section 2.3.1 We determine and prove a closed explicit formula for the factorization of $X^n - a$ for arbitrary $a \in \mathbb{F}_q^*$ and all positive integers n such that $\mathrm{rad}(n) \mid q-1$ and $(4 \nmid n$ or $q \equiv 1 \pmod 4)$ (Theorem 2.37).

Section 2.3.2 We derive our main theorem, a closed explicit formula for the factorization of $X^n - a$ for arbitrary $a \in \mathbb{F}_q^*$ and all positive integers n satisfying $\gcd(n, q) = 1$, from the formula given in Section 2.3.1.

Section 2.3.3 As a corollary of our main theorem we obtain a closed formula for the generating polynomials of all constacyclic codes, Corollary 2.45.

Section 2.4 We translate our main theorem into an algorithm for the factorization of $X^n - a$ over $\mathbb{F}_q$, Algorithm 1.

Section 2.5 We derive closed explicit formulas for the factorization of $X^n - 1$ and Φ_n for all positive integers n satisfying $\gcd(n, q) = 1$ from our main theorem (Theorem 2.47 and Theorem 2.49).

Section 2.5.1 As a corollary of Theorem 2.47 we obtain a closed formula for the generating polynomials of all cyclic codes, Corollary 2.48.

Section 2.6 We present and discuss all explicit formulas for the factorization of $f(X^n)$ over $\mathbb{F}_q$ which were known before.

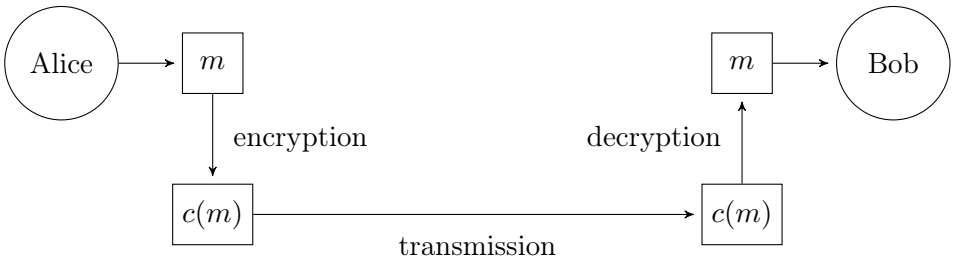
Section 2.7 We derive Theorem 2.61, a closed formula for the factorization of $f(X^n)$ over $\mathbb{F}_q$ for arbitrary monic irreducible polynomials f and all positive integers n satisfying $\gcd(n, q) = 1$, from our main theorem.

Section 2.8 We translate Theorem 2.61 into an algorithm for the factorization of $f(X^n)$ over $\mathbb{F}_q$, Algorithm 2.

Section 2.9 We present an implementation of Algorithm 2 and compare its performance with the existing factoring functions in the computer algebra systems SageMath and Magma.

1.2.3 Irreducible polynomials in cryptography

Another important application of irreducible polynomials over finite fields are cryptographic algorithms, also called *ciphers*. A cipher is needed whenever a sender (Alice) wants to send a message to a receiver (Bob) and fears that the message might be intercepted and read by another party. Then Alice encrypts her message using a cipher and sends the encrypted message to Bob, who can read the original message after decrypting it.



The Advanced Encryption Standard (AES), also known as Rijndael, which is used for the encryption of classified documents by the US government since 2002[1] , is defined on blocks of 128 bits, elements of $\mathbb{F}_2^{128}$, with key lengths of 128, 192 or 256 bits. The specification of the AES can be found in [ST01]. The vector spaces $\mathbb{F}_2^{128}, \mathbb{F}_2^{192}$ and $\mathbb{F}_2^{256}$ over $\mathbb{F}_2$ can be viewed as finite field extensions of $\mathbb{F}_2$ and the security of the cipher also depends on the algebraic properties of the cipher in the finite field. In the AES, the 128bit element of $\mathbb{F}_2^{128}$ is split into 16 bytes, that is, 16 elements of $\mathbb{F}_2^8$ (the so-called *state array*):

$$(\underbrace{b_{127},\ldots,b_{120}}_{\text{byte } s_{0,0}},\ldots,\underbrace{b_7,\ldots,b_0}_{\text{byte } s_{3,3}}) \in \mathbb{F}_2^{128} \to \begin{pmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{pmatrix} \in (\mathbb{F}_2^8)^{4\times 4}.$$

Every byte $(a_7,\ldots,a_0) \in \mathbb{F}_2^8$ of the state array is cast to the finite field $\mathbb{F}_{2^8}$ as $[a_7 \cdot X^7 + \ldots + a_1X + a_0] \in \mathbb{F}_2[X]/\langle f\rangle$, where $f = X^8 + X^4 + X^3 + X + 1 \in \mathbb{F}_2[X]$. The irreducible polynomial f has order 51, is 1-normal and of weight 5 (which is the lowest possible weight for an irreducible polynomial of degree 8 over $\mathbb{F}_2$). The 16 field elements are then substituted using a so-called *S-Box*. An S-Box is simply a permutation of the finite field $\mathbb{F}_{2^8}$. The choice of this S-Box is crucial and a lot of research has been conducted to find S-Boxes with good cryptographic properties (see for example [Can05; Liu+05; CB08; TBD08; KK09; KR08; AHJ10; Cui+11; HJ12]). Conventionally, the AES uses a composition of the inverse function where 0 is mapped to itself with an affine function. In the next step, the rows of the state array are shifted. Then, in the third step of the cipher, the columns of the state array are interpreted as the coefficient vector of a polynomial over $\mathbb{F}_{2^8}$, that is, the column $(s_{0,j}, s_{1,j}, s_{2,j}, s_{3,j})$ defines the polynomial $s_j = s_{3,j}X^3 + s_{2,j}X^2 + s_{1,j}X + s_{0,j} \in \mathbb{F}_{2^8}[X]$. Let $\mathbb{F}_{2^8} = \mathbb{F}_2[X]/\langle f\rangle = \mathbb{F}_2(\alpha)$, then in the third step the column s_j is replaced by the result of the multiplication $s_j \cdot g \mod X^4+1$, where $g = (\alpha+1)\cdot X^3+X^2+X+\alpha \in \mathbb{F}_{2^8}[X]$. In the last step, keys are added bit-wise to the state array. However, to generate the keys (Key Expansion) the polynomial $h = X^3 \in \mathbb{F}_{2^8}[X]$ is used to rotate the entries of an element $(b_0, b_1, b_2, b_3) \in \mathbb{F}_{2^8}$. More precisely, the coefficients of the polynomial $(b_3X^3 + b_2X^2 + bX + b_0)\cdot X^3 \mod X^4+1$ are (b_1, b_2, b_3, b_0). Obviously, the finite field $\mathbb{F}_{2^8}$ plays a central role in this cipher and the complexity of the computations in $\mathbb{F}_{2^8}$ is determined by the polynomial $f = X^8 + X^4 + X^3 + X + 1$.

The computational power of our computers grows fast and ciphers that might be secure today will not be secure anymore in the future. The next milestone is going to be the

[1]The AES has been FIPS-approved (Federal Information Processing Standard) and become effective on May 26, 2002.

quantum computer and cryptographers have been working on ciphers resistant against its computational powers for years. The Kyber cipher won the six-year NIST competition for quantum-resistant cryptographic algorithms in 2022[2]. Its specification is given in [Ava+20]. Kyber is based on the quotient ring $R_{3329} := \mathbb{F}_{3329}[X]/\langle X^{256}+1\rangle$. Since the factorization of $X^{256}+1$ into monic irreducible polynomials over $\mathbb{F}_{3329}$ is given by $X^{256}+1 = \prod_{i=0}^{127}(X^2-\zeta_{256}^{2i+1})$, the ring R_{3329} is isomorphic to

$$\mathbb{F}_{3329}[X]/\langle X^2-\zeta_{256}^{0+1}\rangle \times \ldots \times \mathbb{F}_{3329}[X]/\langle X^2-\zeta_{256}^{2\cdot 127+1}\rangle$$

and multiplications in Kyber are computed in this ring. There exist three different versions of Kyber. All of them are defined on 256 bits and the public key consists of $3072 \cdot k$ bits for $k \in \{2,3,4\}$. Without going into any details of the cipher specification, we can see that larger computational power leads to ciphers over much larger finite fields. And that ciphers depend on polynomials over finite fields and their factorization. Thus, irreducible polynomials over finite fields are indispensable in our modern world.

There exist many different approaches to the construction of irreducible polynomials over a finite field. A popular technique is the so-called composition method. With this method one takes an irreducible polynomial f over $\mathbb{F}_q$ of degree k and composes it with a rational function $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ such that the rational transformation $f^Q = h^k f\left(\frac{g}{h}\right)$ is irreducible itself. The new irreducible polynomial f^Q is of degree $k \cdot \deg(Q)$. Usually, the rational function Q is chosen so that the new degree is higher than the initial degree k.

The technique can easily be applied recursively by constructing $f^Q, (f^Q)^Q, ((f^Q)^Q)^Q, \ldots$, and yields a sequence of irreducible polynomials of increasing degree and order. Constructions of this type have been presented in [Wie88; Mey90; Coh92; McN95; Kyu02; Kyu03; Kyu06; AK11; KK11; PRW20].

In [Ugo13; Ugo15; Ugo16] the author considers polynomials such that the rational transformation f^Q is not irreducible but factors into two irreducible polynomials of the same degree. He then selects one of the irreducible factors and applies the Q-transform to this factor. With this procedure he obtains a finite sequence of polynomials of the same degree until the rational transformation is irreducible for one polynomial of the sequence and the classical recursive method applies again.

In [Mey95] the author shows that the polynomials obtained from the construction in [Coh92] are *normal* polynomials. That is, the roots of these polynomials are linearly independent over $\mathbb{F}_q$. [Cha97] gives a simpler proof for the fact that the constructed polynomials in [Coh92] are normal and extends the result to the polynomials constructed in [McN95]. There exist several constructions of normal polynomials [Sem89; VG90; Gao93] - some of which also use the composition method [Kyu04; AM15; SAS22]. Recursive application of the composition method has also be used to construct sequences of k-normal polynomials in [ADM18; KS20].

1.2.4 Overview of Chapter 3

In Chapter 3 we present a recursive construction of a large set of irreducible polynomials (of the same degree) over $\mathbb{F}_q$ which is based on a reversal of the popular composition method for the composition $f(X^n)$. More precisely, from one given monic irreducible polynomial $f = m_{\beta,q} \in \mathbb{F}_q[X]$, we construct $m_{\beta^n,q}$ for every positive integer n such that $\mathrm{rad}(n) \mid (q-1)$, where β is a root of f.

If we are looking for a monic irreducible polynomial of the same degree as f but with better cryptographic properties, chances are high that there are polynomials of low weight (or

[2] https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms

any other desirable property) among them. Moreover, all computations of our construction are carried out in $\mathbb{F}_q$ which is a big advantage since computations in extension fields are costly.

Additionally, the behavior of our construction can be used to gain information on the order of the monic irreducible polynomial f. Recall that the order of f was a divisor of $q^{\deg(f)}-1$. We define Construction 2 which determines a list of candidates for the order of f (much smaller than the set of divisors of $q^{\deg(f)}-1$) with computations carried out only in $\mathbb{F}_q$. For some polynomials f it even yields the exact order of f.

Chapter 3 is structured as follows:

- Section 3.1 We present two known constructions of $m_{\beta^n,q}$ from $m_{\beta,q}$ for positive integers n whose ideas we combine and extend to define our new construction.
- Section 3.2 We present and prove new theoretical results on the characteristic polynomial $\chi_{\beta^n,q}$ and the minimal polynomial $m_{\beta^n,q}$ of β^n over $\mathbb{F}_q$ for positive integers n.
- Section 3.3 We derive the new construction, Construction 1, from our theoretical results in Section 3.2 and discuss how its behavior can be used to gain information on the order of f. These observations result in a second construction, Construction 2.
- Section 3.4 We discuss which polynomials can be constructed with Construction 1 and develop a procedure to compute all of them quite efficiently. Additionally, we present and discuss an implementation of Construction 1, Construction 2 and this procedure.

1.3 How to use the source code in Appendix B

In Appendix B we present the source code of SageMath/Python implementations of many theoretical aspects discussed in Chapters 2 and 3. SageMath is an open-source computer algebra system written in Python. Information on how to install SageMath can be found here:

`https://doc.sagemath.org/html/en/installation/index.html`.

A comprehensive tutorial for the use of SageMath can be found here:

`https://doc.sagemath.org/html/en/tutorial/index.html`.

After having installed SageMath open a terminal and start SageMath with the simple command:

```
sage
```

Inside the `sage:` command line, a program given in a `.sage`-file can be executed with the following command. Here */path/to/sage/files/* should be replaced with the path to the folder, where the `.sage`-file is, and *filename.sage* should be replaced with the name of the `.sage`-file.

```
sage: load("/path/to/sage/files/filename.sage")
```

The first two files presented in Appendix B - B.1 `RichFiniteFieldClass.sage` and B.2 `RichPolynomialClass` - consist of a set of Python classes and functions that are useful for working with the same polynomial over different finite fields and their base or extension fields. Both files need to be preparsed in order to be loaded as a python module in other SageMath programs. To preparse the file `RichFiniteFieldClass.sage` navigate to the folder which contains the file and use the following terminal command (in your computer's terminal, not in the `sage:` command line):

```
sage --preparse RichFiniteFieldClass.sage
```

In the same folder there will appear a file named `RichFiniteFieldClass.sage.py` which can be renamed to `RichFiniteFieldClass.py` and loaded as a Python module by adding the following line to the beginning of your Python or SageMath code:

```
from RichFiniteFieldClass import *
```

The file `RichPolynomialClass.py` can be obtained in the same way. All programs which we present in Appendix B use the classes `RichFiniteFieldClass` and `RichPolynomialClass`. The SageMath script B.3 `RichTutorial.sage` illustrates how to use the two modules.

Remark 1.3. The class `RichExtensionField` from the module `RichFiniteField` is based on the SageMath class `RelativeFiniteFieldExtension`[3] which is marked as experimental. The class throws a warning when SageMath creates an instance of this class. However, at the moment the class simply uses the function `Hom` to obtain a homomorphism between the two finite fields. This function belongs to the standard library of SageMath.

List of Source Code in Appendix B

[3] `https://doc-gitlab.sagemath.org/html/en/reference/coding/sage/coding/relative_finite_field_extension.html` [February 26, 2024]

Chapter 2

Closed formulas for the factorization of $X^n - a$, $X^n - 1$, the n-th cyclotomic polynomial Φ_n and $f(X^n)$

Our main theorem in this chapter, Theorem 2.41, is a closed formula for the factorization of the polynomial $X^n - a$ into monic irreducible polynomials over $\mathbb{F}_q$ for every element $a \in \mathbb{F}_q^*$ and every positive integer n such that $\gcd(n, q) = 1$. From Theorem 2.41 we derive closed explicit formulas for the factorization of $X^n - 1$ (Theorem 2.47), the n-th cyclotomic polynomial Φ_n (Theorem 2.49) and compositions of the form $f(X^n)$ for arbitrary monic irreducible polynomials $f \in \mathbb{F}_q[X], f \neq X$, (Theorem 2.61), for arbitrary positive integers n such that $\gcd(n, q) = 1$. Both Theorem 2.41 and Theorem 2.61 can be translated into an algorithm for the computation of the factorization of $X^n - a$ or $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ (see Section 2.4 and Section 2.8). All results in this chapter (except for the algorithms) have been made available on ArXiV (see [Gra23]).

The polynomial $X^n - a$ for $a \in \mathbb{F}_q^*$ is a composition of the form $f(X^n)$ for an irreducible polynomial itself, since $f = X - a$ is an irreducible polynomial over $\mathbb{F}_q$. It has been known for a very long time under which conditions compositions of this form are irreducible. In 1866 Joseph Alfred Serret gave the following irreducibility criterion, Theorem 2.2, for polynomials of the form $f(X^n)$ for an irreducible polynomial $f \neq X$ over a prime field in [Ser66, Tome 2]. His theorem also holds in finite fields of a prime power size.

Remark 2.1. When considering the composition $f(X^n)$ for an irreducible polynomial f, the polynomial $f = X$ is of no particular interest because $f(X^n) = X^n = (X)^n$ is the factorization over $\mathbb{F}_q$ of its composition with X^n for every positive integer n.

In [Coh69] Stephen D. Cohen attributes Serret's theorem to A. E. Pellet and claims that Pellet stated the result in his note [Pel81] without giving a proof for it. However, in [Pel81] Pellet considers only special choices of n and the general result by Serret was known long before. In fact, in [Pel89] Pellet attributes the general result to Serret. For a clarification we give the exact reference to the original result which we could not find anywhere else.

Theorem 2.2 ([Ser66, Tome 2, 356, Théorème I],[LN94, Theorem 3.35])**.** *Let n be a positive integer and $f \in \mathbb{F}_q[X]$, $f \neq X$, be an irreducible polynomial over $\mathbb{F}_q$ of degree k and order e. Then the polynomial $f(X^n)$ is irreducible if and only if the following three conditions are satisfied:*

(i) $\mathrm{rad}(n) \mid e$,

(ii) $\gcd(n, \frac{q^k-1}{e}) = 1$,

(iii) if $4 \mid n$, *then* $q^k \equiv 1 \pmod 4$.

Almost 100 years later, in 1955, M.C.R. Butler determined the number of irreducible factors of compositions of the form $f(X^n)$, together with their order and degree, for an irreducible polynomial f over $\mathbb{F}_q$:

Theorem 2.3 ([But55])**.** *Let n be a positive integer such that $\gcd(n, q) = 1$ and $f \in \mathbb{F}_q[X]$, $f \neq X$, be an irreducible polynomial over $\mathbb{F}_q$ of degree k and order e. Further, let $n = n_1 \cdot n_2$, where $\mathrm{rad}(n_1) \mid e$ and $\gcd(n_2, e) = 1$. Then*

(i) Each root of $f(X^n)$ has order $d \cdot n_1 \cdot e$ for a divisor d of n_2.

(ii) For every divisor d of n_2, there exist exactly $k \cdot n_1 \cdot \frac{\varphi(d)}{\mathrm{ord}_{d \cdot n_1 \cdot e}(q)}$ irreducible factors of $f(X^n)$ over $\mathbb{F}_q$. Each of these factors has degree $\mathrm{ord}_{d \cdot n_1 \cdot e}(q)$.

Remark 2.4. Many authors include the number of irreducible factors in the statement of their results on the factorization of $X^n - 1$, $X^n - a$ or $f(X^n)$. We omit this information in this thesis and refer to Theorem 2.3 instead.

Remark 2.5. Furthermore, without loss of generality we can assume that f is monic. Indeed, if $f = \sum_{i=0}^{k} a_i X^i$ is not monic, then $a_k \in \mathbb{F}_q^* \setminus \{1\}$ and $f = a_k \cdot g$ for a monic polynomial $g \in \mathbb{F}_q[X]$. Thus, if $\prod_{R \in \mathcal{R}} R$ is the factorization of $g \in \mathbb{F}_q[X]$ over $\mathbb{F}_q$, then the factorization of f over $\mathbb{F}_q$ equals $a_k \cdot \prod_{R \in \mathcal{R}} R$.

The following theorems show how the factorization of $f(X^n)$ for a monic irreducible polynomial $f \in \mathbb{F}_q[X]$ of degree k and a positive integer n can be derived from the factorization of $X^n - \alpha$ over $\mathbb{F}_{q^k}$, where α is a root of f. The existing results on the factorization of $f(X^n)$ in [MRS19] (which we present in Section 2.6) do not use this connection but were proved in a direct way. This is not necessary and actually doubles the work, because the tools in the proof are the same as the tools used in [MGO15] and [WYF18] to determine the factorization of $X^n - 1$.

The composition $f(X^n)$ is a rational transformation of the form $f^Q = h^k f(\frac{g}{h})$ for the rational function $Q = \frac{X^n}{1} \in \mathbb{F}_q(X)$. For any rational function $Q = \frac{g}{h} \in \mathbb{F}_q(X)$, the Q-transform of f is directly connected to the polynomial $g - \alpha h$ over $\mathbb{F}_{q^k}$, as we can see from the following well-known theorem by Cohen.

Theorem 2.6 ([Coh69, Lemma 1])**.** *Let $f \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k and $\alpha \in \mathbb{F}_{q^k}$ be a root of f. Further, let $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ be a rational function over $\mathbb{F}_q$. Then the polynomial f^Q is irreducible over $\mathbb{F}_q$ if and only if the polynomial $g - \alpha h$ is irreducible over $\mathbb{F}_{q^k}$.*

We include a proof of Theorem 2.6 which was given by Kyuregyan and Kyureghyan in [KK11] and shows the close connection between the two polynomials. For this proof we need the following definitions and the next theorem, Theorem 2.7.

For positive integers j and d, and a polynomial $h = \sum_{i=0}^{m} a_i X^i$ over $\mathbb{F}_{q^d}$, we define the polynomial

$$h^{(j)} := \sum_{i=0}^{m} a_i^{q^j} X^i.$$

Note that $h^{(j)} = h$ if j is a multiple of $[\mathbb{F}_q(a_0, \ldots, a_m) : \mathbb{F}_q]$, the extension degree of $\mathbb{F}_q(a_0, \ldots, a_m)$ over $\mathbb{F}_q$. The extension degree $[\mathbb{F}_q(a_0, \ldots, a_m) : \mathbb{F}_q]$ plays a key role in the arguments to come. We denote it by $\mathrm{coeffdeg}_q(h)$ and call it the *degree of the coefficients of h over $\mathbb{F}_q$*.

Theorem 2.7 ([KK11, Lemma 1])**.** *Let $g \in \mathbb{F}_q[X]$ be a monic polynomial of degree dm, where $d, m \in \mathbb{N}$. Then g is irreducible over $\mathbb{F}_q$ if and only if there exists a monic irreducible polynomial h of degree m over $\mathbb{F}_{q^d}$ such that $\mathrm{coeffdeg}_q(h) = d$ and*

$$g = \prod_{j=0}^{d-1} h^{(j)}.$$

From Theorem 2.7 follows directly, that for a monic irreducible polynomial h of degree m over $\mathbb{F}_{q^d}$ such that $\operatorname{coeffdeg}_q(h) = d$ the polynomial $\prod_{j=0}^{d-1} h^{(j)}$ is a monic irreducible polynomial of degree dm over $\mathbb{F}_q$. We call this polynomial the *q-spin of* h and denote it by $\operatorname{spin}_q[h]$. Note that if $\beta \in \mathbb{F}_{q^{dm}}$ is a root of h, then h is the minimal polynomial of β over $\mathbb{F}_{q^d}$ and $\operatorname{spin}_q[h]$ is the minimal polynomial of β over $\mathbb{F}_q$.

Proof of Theorem 2.6. Since f is a monic irreducible polynomial of degree k and α is a root of f, the decomposition of f over $\mathbb{F}_{q^k}$ is $f(X) = \prod_{j=0}^{k-1}(X - \alpha^{q^j})$. Thus, the Q-transform of f considered over $\mathbb{F}_{q^k}$ satisfies:

$$f^Q = h^k \cdot f\left(\frac{g}{h}\right) = \prod_{j=0}^{k-1}(g - \alpha^{q^j} \cdot h) = \prod_{j=0}^{k-1}(g - \alpha h)^{(j)}, \tag{2.1}$$

because $g, h \in \mathbb{F}_q[X]$ and therefore $g^{(j)} = g$ and $h^{(j)} = h$ for all $0 \leq j \leq k-1$. Furthermore, α is a proper element of $\mathbb{F}_{q^k}$, which implies that $\operatorname{coeffdeg}_q(g - \alpha h) = k$. Consequently, if $g - \alpha h$ is irreducible over $\mathbb{F}_{q^k}$, then $\prod_{j=0}^{k-1}(g - \alpha h)^{(j)}$ is the q-spin of $g - \alpha h$ over $\mathbb{F}_q$, which is a monic irreducible polynomial over $\mathbb{F}_q$. Otherwise, if $g - \alpha h$ factors as $\prod_{R\in\mathcal{R}} R$ into monic irreducible factors over $\mathbb{F}_{q^k}$, then $(g - \alpha h)^{(j)} = \prod_{R\in\mathcal{R}} R^{(j)}$ and for all $R \in \mathcal{R}$ the polynomial $\prod_{j=0}^{k-1} R^{(j)}$ is a factor of f^Q over $\mathbb{F}_q$ because $\operatorname{coeffdeg}_q(R) \mid k$. □

Equation (2.1) shows the direct connection between f^Q and $g - \alpha h$. With this connection we obtain the factorization of f^Q over $\mathbb{F}_q$ from the factorization of $g - \alpha h$ over $\mathbb{F}_{q^k}$ as the following theorem shows. Since the paper [MYM10] is not publicly available, we include the proof of Theorem 2.8.

Theorem 2.8 ([MYM10, Lemma 13])**.** *Let $f \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k and $\alpha \in \mathbb{F}_{q^k}$ be a root of f. Further, let $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ and $\prod_{R\in\mathcal{R}} R$ be the factorization of $g - \alpha h$ into irreducible factors over $\mathbb{F}_{q^k}$. Then the factorization of f^Q into monic irreducible factors over $\mathbb{F}_q$ is given by*

$$\prod_{R\in\mathcal{R}} \operatorname{spin}_q[R],$$

where $\operatorname{coeffdeg}_q(R) = k$ *for all irreducible factors R of $g - \alpha h$ over $\mathbb{F}_{q^k}$.*

Proof. From (2.1) follows that

$$f^Q = \prod_{j=0}^{k-1}\Big(\prod_{R\in\mathcal{R}} R\Big)^{(j)} = \prod_{R\in\mathcal{R}}\prod_{j=0}^{k-1} R^{(j)}.$$

Since $\prod_{R\in\mathcal{R}} R$ is the factorization of $g - \alpha h$ and $\operatorname{coeffdeg}_q(g - \alpha h) = k$, the degree of the coefficients of R over $\mathbb{F}_q$ divides k for every $R \in \mathcal{R}$. If $\operatorname{coeffdeg}_q(R) = m$ were smaller than k for a factor R, then the polynomial $g - \alpha h$ would have a root $\gamma \in \mathbb{F}_{q^m} < \mathbb{F}_{q^k}$, which satisfies $g(\gamma) - \alpha \cdot h(\gamma) = 0$. However $g(\gamma)$ and $h(\gamma)$ are elements of $\mathbb{F}_{q^m}$, which implies that $\alpha \cdot h(\gamma)$ is a proper element of $\mathbb{F}_{q^k}$ and cannot be equal to $g(\gamma)$, a contradiction. Thus, $\operatorname{coeffdeg}_q(R) = k$ for all factors R of $g - \alpha h$ and $f^Q = \prod_{R\in\mathcal{R}} \operatorname{spin}_q[R]$. □

From Theorem 2.8 follows directly that it suffices to study the factorization of $g - \alpha h$ over $\mathbb{F}_{q^k}$ to obtain the factorization of f^Q for $Q = \frac{g}{h} \in \mathbb{F}_q(X)$. For the composition $f(X^n) = f^Q$ with $Q = \frac{X^n}{1}$ this is the factorization of the polynomial $X^n - \alpha$ over $\mathbb{F}_{q^k}$.

At first sight it seems as if Theorem 2.8 does not yield new factorizations of the form $f(X^n)$ from the known factorizations of the polynomial $X^n - 1$. Since $\alpha = 1$ holds only for the polynomial $f = X - 1$, the composition $f(X^n)$ is again $X^n - 1$. However, the next two results show that the factorization of $X^n - 1$ over $\mathbb{F}_{q^k}$ yields the factorization of $f(X^n)$ for

every monic irreducible polynomial $f \in \mathbb{F}_q[X]$, $f \neq X$, of degree k such that there exists an element β in $\mathbb{F}_{q^k}$ with $\beta^n = \alpha$, where α is a root of f. In this case the factorization of $X^n - \alpha$ can easily be derived from the factorization of $X^n - 1$, as stated in the following proposition. In terms of constacyclic codes, the α-constacyclic codes are called n-equivalent to cyclic codes (see [CDL14]). We reformulate and prove the well-known connection using the concepts introduced above:

Proposition 2.9 ([Hug00, Lemma 3.1])**.** *Let $n \in \mathbb{N}$ and $a \in \mathbb{F}_q^*$ be such that there exists $b \in \mathbb{F}_q^*$ with $b^n = a$. Set $Q = \frac{X}{b} \in \mathbb{F}_q(X)$. If $\prod_{R\in\mathcal{R}} R$ is the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$, then the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is $\prod_{R\in\mathcal{R}} R^Q$.*

Proof. Since $b^n = a$, we can write

$$\begin{aligned} X^n - a = X^n - b^n &= b^n \left(\left(\frac{X}{b} \right)^n - 1 \right) = b^n \prod_{R\in\mathcal{R}} R\left(\frac{X}{b} \right) \\ &= \prod_{R\in\mathcal{R}} b^{\deg(R)} R\left(\frac{X}{b} \right) = \prod_{R\in\mathcal{R}} R^Q. \end{aligned}$$

For every monic irreducible factor $R \in \mathcal{R}$ let γ_R be a root of R. Then γ_R is a proper element of $\mathbb{F}_{q^{\deg(R)}}$ and the polynomial $X - \gamma_R \cdot b$ is irreducible over $\mathbb{F}_{q^{\deg(R)}}$. The element $\gamma_R \cdot b$ is a root of R^Q and with Theorem 2.6 the polynomial R^Q is irreducible over $\mathbb{F}_q$. □

The following new theorem is the combination of Proposition 2.9 and Theorem 2.8. It shows that if there exists an element $\beta \in \mathbb{F}_{q^k}^*$ such that $\beta^n = \alpha$ for a monic irreducible polynomial $f \in \mathbb{F}_q[X]$ of degree k with a root α, the factorization of $f(X^n)$ is given by the q-spin of the factors R^Q, where $Q = \frac{X}{\beta}$ and $X^n - 1 = \prod_{R\in\mathcal{R}} R$ over $\mathbb{F}_{q^k}$.

Theorem 2.10. *Let $n \in \mathbb{N}$ and $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k. Further, let $\alpha \in \mathbb{F}_{q^k}^*$ be a root of f such that there exists $\beta \in \mathbb{F}_{q^k}^*$ with $\beta^n = \alpha$ and let $Q = \frac{X}{\beta} \in \mathbb{F}_{q^k}(X)$. If $\prod_{R\in\mathcal{R}} R$ is the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_{q^k}$, then the factorization of $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{R\in\mathcal{R}} \operatorname{spin}_q \left[R^Q \right],$$

where $\operatorname{coeffdeg}_q \left(R^Q \right) = k$ *for all monic irreducible factors R of $X^n - 1$ over $\mathbb{F}_{q^k}$.*

There exists an element $\beta \in \mathbb{F}_{q^k}^*$ such that $\beta^n = \alpha$ if and only if $\alpha^{\frac{q^k-1}{\gcd(n,q^k-1)}} = 1$. If the order of α is large and n is not coprime with $q^k - 1$, it is likely that such an element β does not exist. Therefore, we need to find the factorization of $X^n - \alpha$ over $\mathbb{F}_{q^k}$ for $\alpha \neq 1$ in general. For this, we study the factorization of $X^n - a \in \mathbb{F}_q[X]$, where $a \in \mathbb{F}_q^*$, in Section 2.3 and present the resulting factorization of $f(X^n)$ in Section 2.7.

2.1 Known results on the factorization of $X^n - 1$ and Φ_n

Recall that for every positive integer n such that $\gcd(n, q) = 1$ there exists a primitive n-th root of unity ζ_n in $\overline{\mathbb{F}}_q$ and $X^n - 1$ decomposes over $\overline{\mathbb{F}}_q$ as

$$X^n - 1 = \prod_{j=0}^{n-1} (X - \zeta_n^j). \tag{2.2}$$

Furthermore, if n divides $q - 1$, then $\zeta_n \in \mathbb{F}_q$ and (2.2) is the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$. If n does not divide $q - 1$, then the factorization of $X^n - 1$ over $\mathbb{F}_q$ can be derived from (2.2) with the following corollary of Theorem 2.7:

Corollary 2.11. *Let g be a polynomial over $\mathbb{F}_q$ and $\prod_{R\in\mathcal{R}} R$ its factorization into monic irreducible factors over $\mathbb{F}_{q^w}$ for $w \geq 1$. We define the equivalence relation $\sim$ on $\mathcal{R}$ as follows: $R \sim \tilde{R}$ if and only if $\tilde{R} = R^{(j)}$ for a positive integer j. Then the factorization of g into monic irreducible polynomials over $\mathbb{F}_q$ is given by:*

$$\prod_{R\in\mathcal{R}/\sim} \operatorname{spin}_q[R].$$

Consequently, with Corollary 2.11 there exists a subset $\mathcal{J}$ of $\{0,\ldots,n-1\}$ such that

$$X^n - 1 = \prod_{j\in\mathcal{J}} m_{\zeta_n^j,q} = \prod_{j\in\mathcal{J}} \operatorname{spin}_q\left[X - \zeta_n^j\right]. \tag{2.3}$$

For any $j \in \mathcal{J}$, the coefficient degree of $X - \zeta_n^j$ over $\mathbb{F}_q$ is

$$\operatorname{coeffdeg}_q\left(X - \zeta_n^j\right) = \operatorname{ord}_{\frac{n}{\gcd(j,n)}}(q),$$

because the order of ζ_n^j is $\frac{n}{\gcd(j,n)}$. As we can see from eq. (2.3) for two integers $0 \leq j, j' \leq n-1$ the elements ζ_n^j and $\zeta_n^{j'}$ are roots of the same monic irreducible polynomial if and only if $j \equiv j' \cdot q^i \pmod{n}$ for an integer i. For this reason we define the *q-cyclotomic coset modulo n containing j* as follows:

$$\mathrm{C}_{q,n}(j) = \{j \cdot q^i \bmod n : i \geq 0\}.$$

The size of $\mathrm{C}_{q,n}(j)$ is $k = \operatorname{ord}_{\frac{n}{\gcd(j,n)}}(q)$ because ζ_n^j is a proper element of $\mathbb{F}_{q^k}$. Let $\mathrm{CR}_q(n)$ denote a complete set of representatives of the q-cyclotomic cosets modulo n. Then we can set $\mathcal{J} = \mathrm{CR}_q(n)$. For $j' \in \mathrm{C}_{n,q}(j)$ holds $\operatorname{ord}(\zeta_n^{j'}) = \operatorname{ord}(\zeta_n^j)$, because the two elements are conjugates over $\mathbb{F}_q$. From this follows directly that $\gcd(j,n) = \gcd(j',n)$. Therefore, we can rewrite eq. (2.3) in the following way:

$$X^n - 1 = \prod_{d|n} \prod_{\substack{j\in\mathrm{CR}_q(n)\\ \gcd(j,n)=\frac{n}{d}}} \left[\prod_{i=0}^{\operatorname{ord}_d(q)-1} (X - \zeta_n^{j\cdot q^i})\right]. \tag{2.4}$$

Recall that the n-th cyclotomic polynomial Φ_n is the polynomial whose roots are the $\varphi(n)$ distinct primitive n-th roots of unity over $\mathbb{F}_q$. Since for every $0 \leq j \leq n-1$ the element ζ_n^j is a primitive n-th root of unity if and only if $\gcd(j,n) = 1$, (2.4) yields the following factorization of Φ_n:

$$\Phi_n = \prod_{\substack{j\in\mathrm{CR}_q(n)\\ \gcd(j,n)=1}} \operatorname{spin}_q\left[X - \zeta_n^j\right]. \tag{2.5}$$

From equation (2.5) follow directly the two well-known facts that Φ_n is a polynomial with coefficients in $\mathbb{F}_q$ and that Φ_n factors into $\frac{\varphi(n)}{\operatorname{ord}_n(q)}$ distinct monic irreducible polynomials over $\mathbb{F}_q$ of degree $\operatorname{ord}_n(q)$ (see [LN94, Theorem 2.47]).

The factorizations in (2.4) and (2.5) rely on the primitive n-th root of unity ζ_n which might lie an extension field of very large degree over $\mathbb{F}_q$. Thus, it is of interest to find a description of the irreducible factors of $X^n - 1$ using only elements of $\mathbb{F}_q$. Or, if this is not possible, elements of extension fields with extension degree as small as possible.

Recall that the factorization of $X^n - 1$ over $\mathbb{F}_q$ is equivalent to the determination of the generating polynomials of all cyclic codes. Many results on the factorization of $X^n - 1$ or Φ_n have been proved multiple times because researchers in the domain of cyclic codes were not always aware of the advances in the domain of discrete mathematics. We only list the references to the first appearance of the results here.

The explicit factorization of the n-th cyclotomic polynomial over the finite field $\mathbb{F}_q$ was known before for the following parameters:

Table 2.1: Known explicit factorizations of Φ_n

n		q	Reference
$n = p$	for a prime $p \neq q$	q prime	[Ste01]
$n = 2^m \cdot p$	for an odd prime p, $p \mid q^2 - 1$, $m \in \mathbb{N}$	q odd	[FY07]
$n = 2^m \cdot 5$	$m \in \mathbb{N}$		[WW12]
$n = 2^m \cdot r$	for $r \geq 3$ odd, $\gcd(q, r) = 1$, $m \in \mathbb{N}$ if Φ_r is factored		[TW13]
$n = p^m \cdot r$	for an odd prime p, $p \mid (q-1)$, $\gcd(r, p) = \gcd(r, q) = 1$, $r, m \in \mathbb{N}$ if Φ_{p^m} and Φ_r are factored		[Wu+17]

The explicit factorization of $X^n - 1$ was known before for the following parameters:

Table 2.2: Known explicit factorizations of $X^n - 1$

n		q	Reference
$n = 2^m$	$m \in \mathbb{N}$	$q \equiv 1 \pmod 4$	[LN94]
$n = 2^m$	$m \in \mathbb{N}$	q prime, $q \equiv 3 \pmod 4$	[BGM93]
$n = 2^m$	$m \in \mathbb{N}$	$q \equiv 3 \pmod 4$	[Mey96]
$n = 2^m \cdot p^l$	for an odd prime p, $p \mid (q-1)$, $m, l \in \mathbb{N}$	q odd	[CLT13]
$n \in \mathbb{N}$	such that $\mathrm{rad}(n) \mid (q-1)$		[MGO15]
$n \in \mathbb{N}$	such that $\mathrm{rad}(n) \nmid (q-1)$, $\mathrm{rad}(n) \mid q^w - 1$, w prime		[WYF18]
$n \in \mathbb{N}$	such that $\mathrm{rad}(n) \nmid (q-1)$, $\mathrm{rad}(n) \mid q^{vw} - 1$, $v \neq w$ primes		[WY21]

We present some results of [MGO15], [WYF18] and [WY21] in detail to show that our Theorem 2.47 simplifies the statement and the computation of the explicit factorization of $X^n - 1$, because it combines and extends all of the existing results in one closed formula for any positive integer n such that $\gcd(n, q) = 1$.

We simplified the notation of Theorems 2.12, 2.13 and 2.17 by introducing the definition $d^{(w)} := \gcd(n, q^w - 1)$ for every positive integer w. Obviously, $d^{(w)}$ is a divisor of $q^w - 1$ and the primitive $d^{(w)}$-th root of unity $\zeta_{d^{(w)}}$ is an element of $\mathbb{F}_{q^w}$. Furthermore, recall that we do not state the number of irreducible factors in the given factorizations because this number was already given in Theorem 2.3.

The following result gives the factorization of $X^n - 1$ over $\mathbb{F}_q$ for every positive integer n such that $\mathrm{rad}(n) \mid (q-1)$ and ($8 \nmid n$ or $q \equiv 1 \pmod 4$).

Theorem 2.12 ([MGO15, Corollary 1]). *Let $n \in \mathbb{N}$ such that $\mathrm{rad}(n) \mid (q-1)$ and ($8 \nmid n$ or $q \equiv 1 \mod 4$). Set $d^{(1)} = \gcd(n, q-1)$. Then the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{t \mid \frac{n}{d^{(1)}}} \prod_{\substack{0 \le u \le d^{(1)}-1 \\ \gcd(u,t)=1}} (X^t - \zeta_{d^{(1)}}^u).$$

The next theorem completes the case $\mathrm{rad}(n) \mid (q-1)$.

Theorem 2.13 ([MGO15, Corollary 2]). *Let $n \in \mathbb{N}$ such that $\mathrm{rad}(n) \mid (q-1)$, $8 \mid n$ and $q \equiv 3 \mod 4$. Set $d^{(w)} := \gcd(n, q^w - 1)$ for $w \in \{1, 2\}$. Then $d^{(2)} = 2^l \cdot d^{(1)}$, where $l = \min\{\nu_2(\frac{n}{2}), \nu_2(q+1)\}$. Set $\zeta_{d^{(1)}} = \zeta_{d^{(2)}}^{2^l}$. Then the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{\substack{t \mid \frac{n}{d^{(2)}} \\ 2 \nmid t}} \prod_{\substack{0 \le v \le d^{(1)}-1 \\ \gcd(v,t)=1}} (X^t - \zeta_{d^{(1)}}^v) \cdot \prod_{t \mid \frac{n}{d^{(2)}}} \prod_{u \in \mathcal{U}_t} (X^{2t} - (\zeta_{d^{(2)}}^u + \zeta_{d^{(2)}}^{uq})X^t + \zeta_{d^{(1)}}^{u\frac{q+1}{2^l}}),$$

where $\mathcal{U}_t = \{1 \le u \le d^{(2)} : \gcd(u, t) = 1, 2^l \nmid u, u < (uq \bmod d^{(2)})\}$.

With the following lemma we show that the set $\mathcal{U}_t$ in Theorem 2.13 can be chosen more freely and is actually a subset of $\mathrm{CR}_q(d^{(2)})$, the set of the q-cyclotomic representatives modulo $d^{(2)}$:

Lemma 2.14. *Let n and w be positive integers such that $\mathrm{rad}(n) \mid q^w - 1$ and $d^{(w)} = \gcd(n, q^w - 1)$. Furthermore, let $1 \le u \le d^{(w)}$ and $\tilde{u} \in \mathrm{C}_{q,d^{(w)}}(u)$.*

(i) For every positive integer t such that $\mathrm{rad}(t) \mid \mathrm{rad}(n)$ holds $\gcd(u, t) = 1$ if and only if $\gcd(\tilde{u}, t) = 1$.

(ii) For every divisor m of $d^{(w)}$ holds $m \nmid u$ if and only if $m \nmid \tilde{u}$.

Proof. Since $d^{(w)} = \gcd(n, q^w - 1)$ and $\mathrm{rad}(n) \mid q^w - 1$, we have $\mathrm{rad}(d^{(w)}) = \mathrm{rad}(n)$. Let $1 \le u \le d^{(w)}$ and $\tilde{u} \in \mathrm{C}_{q,d^{(w)}}(u)$ or, equivalently, $\tilde{u} = u \cdot q^i + r \cdot d^{(w)}$ for a positive integer i and an integer r.

(i) Suppose that the greatest common divisor of u and t is greater than 1. Then there exists a prime p which divides $\gcd(u, t)$ and since $\mathrm{rad}(t) \mid \mathrm{rad}(n) = \mathrm{rad}(d^{(w)})$, p also divides $d^{(w)}$. From $\tilde{u} = \gcd(u, t) \cdot \frac{u}{\gcd(u,t)} \cdot q^i + r \cdot d^{(w)}$ follows that $p \mid \tilde{u}$. Thus, $\gcd(\tilde{u}, t) > 1$. Since $\tilde{u} \in \mathrm{C}_{q,d^{(w)}}(u)$ if and only if $u \in \mathrm{C}_{q,d^{(w)}}(\tilde{u})$, the condition $\gcd(u, t) = 1$ holds if and only if $\gcd(\tilde{u}, t) = 1$.

(ii) Suppose that $m \mid u$, then m divides $\tilde{u} = m \cdot \frac{u}{m} \cdot q^i + r \cdot d^{(w)}$ as well, because $m \mid d^{(w)}$. From the fact that $\tilde{u} \in \mathrm{C}_{q,d^{(w)}}(u)$ if and only if $u \in \mathrm{C}_{q,d^{(w)}}(\tilde{u})$, follows again that $m \nmid u$ if and only if $m \nmid \tilde{u}$. □

Remark 2.15. In this remark we show that the set $\mathcal{U}_t$ can be chosen more freely and that the monic irreducible factor $X^{2t} - (\zeta_{d^{(2)}}^u + \zeta_{d^{(2)}}^{qu})X^t + \zeta_{d^{(1)}}^{u \cdot \frac{q+1}{2^l}}$ is in fact the q-spin of $X^t - \zeta_{d^{(2)}}^u$ for a positive integer $u \in \mathcal{U}_t$. From $8 \mid n$ and $2 \mid q-1$ follows that $\nu_2\left(\frac{n}{2}\right)$ and $\nu_2(q+1)$ are both greater than or equal to 1. Consequently, $l \ge 1$ and $2 \cdot d^{(1)} \mid d^{(2)}$. Thus, $d^{(2)}$ does not divide $q-1$ and $\zeta_{d^{(2)}}$ is a proper element of $\mathbb{F}_{q^2}$. For every $1 \le u \le d^{(2)}$ the q-cyclotomic coset of u modulo $d^{(2)}$ is either $\mathrm{C}_{q,d^{(2)}}(u) = \{u\}$ or $\mathrm{C}_{q,d^{(2)}}(u) = \{u, (uq \bmod d^{(2)})\}$ depending on whether the order $\frac{d^{(2)}}{\gcd(u,d^{(2)})}$ of $\zeta_{d^{(2)}}^u$ divides $q-1$ or not. This is true if and only if $\frac{d^{(2)}}{d^{(1)}} = 2^l$ divides u. Thus, in $\mathcal{U}_t$ we collect only elements u of $\{1, \ldots, d^{(2)}\}$ such that $\zeta_{d^{(2)}}^u$ is a proper element of $\mathbb{F}_{q^2}$. Furthermore, for every such u we select exactly one representative

of the q-cyclotomic coset of u modulo $d^{(2)}$, namely the smaller positive integer in $\mathrm{C}_{q,d^{(w)}}(u)$. However, the monic irreducible factor $(X^{2t} - (\zeta_{d^{(2)}}^{u} + \zeta_{d^{(2)}}^{uq})X^t + \zeta_{d^{(1)}}^{u\frac{q+1}{2^l}})$ is independent of the choice of the representative of $\mathrm{C}_{q,d^{(w)}}(u)$, because $\zeta_{d^{(1)}}^{\frac{q+1}{2^l}\cdot u} = \zeta_{d^{(2)}}^{(q+1)u} = \zeta_{d^{(2)}}^{u} \cdot \zeta_{d^{(2)}}^{qu}$. From Lemma 2.14 (i) and (ii) follows that the conditions $\gcd(u,t) = 1$ and $2^l \nmid u$ are satisfied for any other representative of $\mathrm{C}_{q,d^{(w)}}(u)$. Thus, if $\mathrm{CR}_q(d^{(2)})$ is a set of representatives of the q-cyclotomic cosets modulo $d^{(2)}$, then we can select u from the set

$$\{u \in \mathrm{CR}_q(d^{(2)}) : \gcd(u,t) = 1, 2^l \nmid u\}$$

instead of $\mathcal{U}_t$. Furthermore, for every $u \in \mathcal{U}_t$, the polynomial $X^{2t} - (\zeta_{d^{(2)}}^{u} + \zeta_{d^{(2)}}^{qu})X^t - \zeta_{d^{(1)}}^{u\frac{q+1}{2^l}}$ equals $(X^t - \zeta_{d^{(2)}}^{u})(X^t - \zeta_{d^{(2)}}^{qu}) = \mathrm{spin}_q\left[X^t - \zeta_{d^{(2)}}^{u}\right]$.

The authors of [WYF18; WY21] use the following result to derive the factorization of $X^n - 1$, for positive integers n such that $\mathrm{ord}_{\mathrm{rad}(n)}(q)$ is prime or the product of two primes, from Theorem 2.12 and Theorem 2.13:

Proposition 2.16 ([Wu+17, Lemma 14])**.** *Let n be a positive integer such that $\gcd(n,q) = 1$ and let $m = \mathrm{ord}_{\mathrm{rad}(n)}(q)$. Then if $g \in \mathbb{F}_{q^m}[X]$ is an irreducible factor of Φ_n over $\mathbb{F}_{q^m}$, then $\prod_{j=0}^{m-1} g^{(j)}$ is an irreducible factor of Φ_n over $\mathbb{F}_q$.*

Since $X^n - 1 = \prod_{d|n} \Phi_d$, the same statement holds for the irreducible factors of $X^n - 1$. We present a sketch of the proofs in [WYF18] and [WY21]:

Proof. Let $\mathrm{ord}_{\mathrm{rad}(n)}(q) = w$ be either prime or the product of two primes.

(1) Since $\mathrm{rad}(n) \mid (q^w - 1)$, Theorem 2.12 or Theorem 2.13 yield the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_{q^w}$: $X^n - 1 = \prod_{R \in \mathcal{R}} R$.

(2) For every $R \in \mathcal{R}$ determine $\mathrm{coeffdeg}_q(R)$ (which is either 1, w itself or one prime factor of w).

(3) Define an equivalence relation $\sim$ on $\mathcal{R}$ as follows: $R \sim R'$ if and only if $R' = R^{(j)}$ for an integer $0 \le j \le \mathrm{coeffdeg}_q(R) - 1$.

(4) Determine $\mathcal{R}/\sim$.

(5) With Proposition 2.16 the factorization of $X^n - 1$ over $\mathbb{F}_q$ is given by

$$X^n - 1 = \prod_{R \in \mathcal{R}/\sim} \left[\prod_{j=0}^{\mathrm{coeffdeg}_q(R)-1} R^{(j)} \right].$$

□

Note that in step (2) the exact order of the coefficients of R is not determined. Furthermore, the product $\prod_{j=0}^{\mathrm{coeffdeg}_q(R)-1} R^{(j)}$ is in fact the q-spin of R. Indeed, Proposition 2.16 follows directly from Corollary 2.11.

The proof of our main theorem in Section 2.3.2 has a similar structure as the proofs in [WYF18] and [WY21]. Instead of Theorem 2.12 and Theorem 2.13 we use Theorem 2.37, our new closed formula for the factorization of $X^n - a$ for arbitrary $a \in \mathbb{F}_q^*$ and every positive integer n such that $\mathrm{rad}(n) \mid q-1$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$) which also specifies the exact order of all the coefficients of the irreducible factors. With the exact order of the coefficients we are able to generalize step (2). Additionally, we generalize step (5) with the use of Corollary 2.11. In fact we do not even need to complete the case $\mathrm{rad}(n) \mid q-1$ but can include the factorization of $X^n - a$ for all positive integers n such that $\mathrm{rad}(n) \mid q-1$ and $4 \mid n$ and $q \equiv 3 \pmod 4$ in our main result, Theorem 2.41. Step (4) is handiwork and the details in our proof differ from the details hidden in step (4) here.

The following nine theorems are given in [WYF18] and [WY21] to describe the factorization for every positive integer n such that $\mathrm{ord}_{\mathrm{rad}(n)}(q)$ is either prime or the product of two primes:

Table 2.3: Overview of the results in [WYF18] and [WY21]

Theorem	for positive integers n such that
[WYF18, Theorem 3.2]	$\mathrm{ord}_{\mathrm{rad}(n)}(q)$ is an odd prime and $(8 \nmid n$ or $q \equiv 1 \pmod 4)$
[WYF18, Theorem 3.4]	$\mathrm{ord}_{\mathrm{rad}(n)}(q)$ is an odd prime and $8 \mid n$ and $q \equiv 3 \pmod 4$
[WYF18, Theorem 3.6]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = 2$
[WY21, Theorem 3.3]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = w^2$ for an odd prime w, $(8 \nmid n$ or $q \equiv 1 \pmod 4)$
[WY21, Theorem 3.5]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = w^2$ for an odd prime w, $8 \mid n$ and $q \equiv 3 \pmod 4$
[WY21, Theorem 3.8]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = 4$
[WY21, Theorem 4.2]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = wv$ for two distinct odd primes w and v, $(8 \nmid n$ or $q \equiv 1 \pmod 4)$
[WY21, Theorem 4.4]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = wv$ for two distinct odd primes w and v, $8 \mid n$ and $q \equiv 3 \pmod 4)$
[WY21, Theorem 4.7]	$\mathrm{ord}_{\mathrm{rad}(n)}(q) = 2w$ for an odd prime w

Theorem 2.12, Theorem 2.13 and all of the theorems listed in Table 2.3 are covered by our closed new formula for the factorization of $X^n - 1$ for every positive integer n such that $\gcd(n, q) = 1$, Theorem 2.41.

We only present the first result from [WYF18], the factorization of $X^n - 1$ for all positive integers n such that $w = \mathrm{ord}_{\mathrm{rad}(n)}(q)$ is prime and $(8 \nmid n$ or $q \equiv 1 \pmod 4)$. The other results become more and more technical and do not give further insight into the structure of the factorization.

Theorem 2.17 ([WYF18, Theorem 3.2])**.** *Let n be a positive integer such that $\mathrm{ord}_{\mathrm{rad}(n)}(q) = w$ for an odd prime w and $(8 \nmid n$ or $q \equiv 1 \pmod 4)$. We write $n = w^{\nu_w(n)} \cdot n_1 \cdot n_2$ with $\mathrm{rad}(n_1) \mid q-1$ and $\mathrm{rad}(n_2) \mid \frac{q^w-1}{q-1}$. Furthermore, we set $d^{(s)} := \gcd(n, q^s - 1)$ for $s \in \{1, w\}$ and $\zeta_{d^{(1)}} = \zeta_{d^{(w)}}^{d^{(w)}/d^{(1)}}$. Then the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{t \mid \frac{n_1}{\gcd(n_1, q-1)}} \prod_{\substack{1 \le v \le d^{(1)} \\ \gcd(v,t)=1}} (X^t - \zeta_{d^{(1)}}^v) \cdot \prod_{t \mid \frac{n}{d^{(w)}}} \prod_{u \in \mathcal{U}_t} \left[\prod_{j=0}^{w-1} (X^t - \zeta_{d^{(w)}}^{uq^j}) \right],$$

where for all $t \mid \frac{n}{d^{(w)}}$ the set $\mathcal{U}_t$ is defined as:

$$\mathcal{U}_t = \{1 \le u \le d^{(w)} : \gcd(u, t) = 1,\ \frac{q^w - 1}{q - 1} \nmid u \cdot \frac{q^w - 1}{d^{(w)}},$$
$$u = \min\{(u \bmod d^{(w)}), (uq \bmod d^{(w)}), \ldots, (uq^{w-1} \bmod d^{(w)})\}\}.$$

Remark 2.18. In this remark we show that instead of $\mathcal{U}_t$ we can select the following subset of the q-cyclotomic representatives modulo $d^{(w)}$:

$$\left\{ u \in \mathrm{CR}_q(d^{(w)}) : \gcd(u, t) = 1,\ \frac{d^{(w)}}{d^{(1)}} \nmid u \right\}$$

and that the monic irreducible factor $\prod_{j=0}^{w-1}(X^t - \zeta_{d^{(w)}}^{uq^j})$ is in fact the q-spin of $(X^t - \zeta_{d^{(w)}}^u)$. Note that for all other results in [WYF18] and [WY21] hold similar statements.

Since $w = \mathrm{ord}_{\mathrm{rad}(n)}(q)$, the $d^{(w)}$-th primitive root of unity $\zeta_{d^{(w)}}$ is a proper element of $\mathbb{F}_{q^w}$. For every $1 \leq u \leq d^{(w)}$, the element $\zeta_{d^{(w)}}^u$ is either an element of $\mathbb{F}_q$ or a proper element of $\mathbb{F}_{q^w}$ because w is prime. Furthermore, $\frac{q^w-1}{q-1} \mid u \cdot \frac{q^w-1}{d^{(w)}}$ holds if and only if the element $\zeta_{q^w-1}^{u \cdot \frac{q^w-1}{d^{(w)}}} = \zeta_{d^{(w)}}^u$ is an element of $\mathbb{F}_q$. This is true if and only if $\frac{d^{(w)}}{d^{(1)}} \mid u$ because $\gcd(d^{(w)}, q-1) = \gcd(\gcd(n, q^w-1), q-1) = \gcd(n, q-1) = d^{(1)}$. Thus, for all $1 \leq u \leq d^{(w)}$ such that $\frac{q^w-1}{q-1} \nmid u \cdot \frac{q^w-1}{d^{(w)}}$, the q-cyclotomic coset of u modulo $d^{(w)}$ is

$$\mathrm{C}_{q,d^{(w)}}(u) = \{(u \bmod d^{(w)}), (uq \bmod d^{(w)}), \ldots, (uq^{w-1} \bmod d^{(w)})\}$$

and [WYF18] select the smallest positive integer in $\mathrm{C}_{q,d^{(w)}}(u)$ as a representative. However, the product $\prod_{j=0}^{w-1}(X^t - \zeta_{d^{(w)}}^{uq^j})$ is independent of the choice of the representative of $\mathrm{C}_{q,d^{(w)}}(u)$. With Lemma 2.14 (i) and (ii) follows that the conditions $\gcd(u,t) = 1$ and $\frac{d^{(w)}}{d^{(1)}} \nmid u$ also hold for every other element of $\mathrm{C}_{q,d^{(w)}}(u)$. Furthermore, since $\zeta_{d^{(w)}}$ is a proper element of $\mathbb{F}_{q^w}$ and $\mathrm{coeffdeg}_q\left(X^t - \zeta_{d^{(w)}}^u\right) = [\mathbb{F}_q(\zeta_{d^{(w)}}^u) : \mathbb{F}_q] = w$, the product $\prod_{j=0}^{w-1}(X^t - \zeta_{d^{(w)}}^{uq^j})$ is in fact the q-spin of $X^t - \zeta_{d^{(w)}}^u$.

2.2 Known results on the factorization of $X^n - a$

The explicit factorization of $X^n - a$ over a finite field $\mathbb{F}_q$ for $a \in \mathbb{F}_q^* \backslash \{1\}$ and a positive integer n has mainly been studied for the construction of constacyclic codes. Recall that an a-constacyclic code of length n is an ideal in $\mathbb{F}_q[X]/\langle X^n - a\rangle$ which is generated by a factor of $X^n - a$. For $a = -1$ an a-constacyclic code is called *negacyclic*. Negacyclic and cyclic codes are closely connected. For even q these two sets of codes are actually the same. For odd q and positive integers n such that $2 \mid n$ the factorization of $X^n - 1$ yields the factorization of $X^{\frac{n}{2}} + 1$, because

$$X^n - 1 = X^{2 \cdot \frac{n}{2}} - 1 = (X^{\frac{n}{2}} - 1)(X^{\frac{n}{2}} + 1).$$

Thus, a specific factorization of $X^n - (-1) = X^n + 1$ over $\mathbb{F}_q$ is not necessary if the factorization of $X^{2n} - 1$ over $\mathbb{F}_q$ is known.

Recall that a (consta-)cyclic code is either called *simple-root* if $\gcd(n, q) = 1$ or *repeated-root* if $\gcd(n, q) > 1$. As we discussed in Remark 1.2, this distinction is irrelevant for the factorization of $X^n - a$. However, the code properties of repeated-root and simple-root codes differ. A code $\mathcal{C} \subseteq \mathbb{F}_q^n$ is considered to be good if

- it contains many code words, that is, if $|\mathcal{C}|$ is large,
- its minimum distance $d(\mathcal{C})$ is large, where

$$d(\mathcal{C}) = \min\{|\{0 \leq i \leq n-1 : c_i \neq c_i'\}| : c, c' \in \mathcal{C}\},$$

- its length n is small,
- the encoding and decoding algorithm are of a low complexity.

[Cas+91] and [Lin91] showed that some repeated-root cyclic codes are *optimal*, meaning that they contain the largest possible number of codewords for their given length and minimum distance. Additionally, their decoding algorithm is of low complexity. However, in general the relative minimum distance $\frac{d(C)}{n}$ tends to zero as $\gcd(n, q)$ grows, making repeated-root constacyclic codes asymptotically bad.

Because of the differences in the code quality many authors determine the factorization of $X^n - a$ over $\mathbb{F}_q$ for positive integers $n = \mathrm{char}(\mathbb{F}_q)^l \cdot \tilde{n}$, where $l \geq 1$ and $\gcd(\tilde{n}, q) = 1$. Since with Remark 1.2 this factorization is equivalent to the factorization of $X^{\tilde{n}} - a$ over $\mathbb{F}_q$ we list the positive integers $\tilde{n}$ instead of n. We further omit all results which do not hold for arbitrary $a \in \mathbb{F}_q^*$. The explicit factorization of $X^n - a$ over $\mathbb{F}_q$ for arbitrary $a \in \mathbb{F}_q^*$ was known before for the following positive integers n:

Table 2.4: Known explicit factorizations of $X^n - a$

n		q	Reference
$n = 2$			[Din12]
$n = p^m$	for a prime $p \neq \mathrm{char}(\mathbb{F}_q)$, $m \in \mathbb{N}$		[Che+12]
$n = 3$			[Din13b]
$n = 4$		q odd	[Din13a]
$n = p^m$	for an odd prime $p \neq \mathrm{char}(\mathbb{F}_q)$, $m \in \mathbb{N}$		[Sha15]
$n = 2 \cdot p^m$	for an odd prime $p \neq \mathrm{char}(\mathbb{F}_q)$, $m \in \mathbb{N}$	q odd	[CDL15]
$n = 3 \cdot p$	for an odd prime $p \neq \mathrm{char}(\mathbb{F}_q)$		[Liu+16]
$n = 4 \cdot p^m$	for an odd prime $p \neq \mathrm{char}(\mathbb{F}_q)$, $m \in \mathbb{N}$	q odd	[SR16]
$n = p_1 \cdot p_2$	for two distinct odd primes p_1, p_2 $p_1, p_2 \neq \mathrm{char}(\mathbb{F}_q)$ such that $p_1 = 2p_3 + 1$ for p_3 prime or $p_3 = 1$		[Liu+17]
$n = p_1 \cdot p_2^m$	for two distinct odd primes $p_1, p_2 \neq \mathrm{char}(\mathbb{F}_q)$ such that $p_2 \nmid \mathrm{ord}_{p_1}(q)$	q odd	[Ton16]
$n = p^m$	for a prime p such that $\mathrm{ord}_p(q)$ is prime and $\gcd(p, q(q-1)) = 1$		[LY18]
$n = 8 \cdot p^m$	for an odd prime $p \neq \mathrm{char}(\mathbb{F}_q)$, $m \in \mathbb{N}$	q odd	[Ran19]
$n \in \mathbb{N}$	such that $\mathrm{rad}(n) \mid (q-1)$		[WY18]
$n = 2^{m_1} \cdot p^{m_2}$	for an odd prime $p \neq \mathrm{char}(\mathbb{F}_q)$, $m_1, m_2 \in \mathbb{N}$, such that $2^{m_1} \mid (q^2 - 1)$		[DR19]
$n = p_1^{m_1} p_2^{m_2}$	for two distinct odd primes p_1, p_2 such that $\mathrm{ord}_{p_1 p_2}(q)$ prime and $\gcd(p_1 p_2, q(q-1)) = 1$		[SF20]
$n = p_1^{m_1} p_2^{m_2} p_3^{m_3}$	for three distinct odd primes p_1, p_2, p_3 such that $\mathrm{ord}_{p_1 p_2 p_3}(q)$ is prime and $\gcd(p_1 p_2 p_3, q(q-1)) = 1$		[Rak+22]

In [WY18] Wu and Yue specify the factorization of $X^n - a$ for all positive integers n such that $\mathrm{rad}(n) \mid (q-1)$ in the following three separate theorems, Theorem 2.19, Theorem 2.20 and Theorem 2.21. The results are derived from the factorizations of $X^n - 1$ for all positive integers n such that $\mathrm{rad}(n) \mid (q-1)$ by [MGO15] and they use similar arguments in their proofs. As in Section 2.1 we simplified the notation of their results by introducing the definition of $d^{(w)} = \gcd(n, q^w - 1)$ for positive integers w.

The following theorem, Theorem 2.19 gives the factorization for all elements $a \in \mathbb{F}_q^*$ such that there exists an element b in $\mathbb{F}_q$ satisfying $b^n = a$. With the use of Proposition 2.9 the theorem is a direct consequence of Theorem 2.12 and Theorem 2.13.

Theorem 2.19 ([WY18, Theorem 6]). *Let n be a positive integer satisfying $\mathrm{rad}(n) \mid (q-1)$ and let a be an element of $\mathbb{F}_q^*$ such that there exists $b \in \mathbb{F}_q^*$ with $a = b^n$. We set $d^{(w)} := \gcd(n, q^w - 1)$ for $w \in \{1, 2\}$.*

(1) If $8 \nmid n$ or $q \equiv 1 \pmod 4$, then the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{t \mid \frac{n}{d^{(1)}}} \prod_{\substack{1 \le u \le d^{(1)} \\ \gcd(u,t)=1}} (X^t - b^t \cdot \zeta_{d^{(1)}}^u).$$

(2) If $8 \mid n$ and $q \equiv 3 \pmod 4$, then let $l = \min\{\nu_2(\frac{n}{2}), \nu_2(q+1)\}$ and the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{\substack{t \mid \frac{n}{d^{(2)}} \\ 2 \nmid t}} \prod_{\substack{1 \le v \le d^{(1)} \\ \gcd(v,t)=1}} (X^t - b^t \cdot \zeta_{d^{(1)}}^v) \cdot \prod_{t \mid \frac{n}{d^{(2)}}} \prod_{u \in \mathcal{U}_t} (X^{2t} - b^t \cdot (\zeta_{d^{(2)}}^u + \zeta_{d^{(2)}}^{uq}) X^t + b^{2t} \cdot \zeta_{d^{(1)}}^{u\frac{q+1}{2^l}}),$$

where $\mathcal{U}_t = \{1 \le u \le d^{(2)} : \gcd(u, t) = 1, 2^l \nmid u, u < (qu \bmod d^{(2)})\}$.

The following theorem specifies the factorization of $X^n - a$ for all positive integers n such that $\mathrm{rad}(n) \mid q-1$ and all elements a of $\mathbb{F}_q^*$ such that $a = b^n \cdot \zeta_{q-1}^j$ for an element $b \in \mathbb{F}_q^*$ and an integer $1 \le j \le d^{(1)} - 1$ satisfying $\nu_p(j) < \nu_p(d^{(1)})$ for all prime factors p of $d^{(1)}$ with $\nu_p(n) > \nu_p(q-1)$.

Theorem 2.20 ([WY18, Theorem 8]). *Let n be a positive integer satisfying $\mathrm{rad}(n) \mid q-1$ and let $d = \gcd(n, q-1) = d_1 \cdot d_2$ with $\nu_p(n) \le \nu_p(q-1)$ for all $p \mid d_1$ and $\nu_p(n) > \nu_p(q-1)$ for all $p \mid d_2$. Further, let $a = b^n \cdot \zeta_{q-1}^j \in \mathbb{F}_q^*$, where $b \in \mathbb{F}_q$ and $1 \le j \le d-1$ such that $\nu_p(j) < \nu_p(d_2)$ for all primes $p \mid d_2$. Set $d' := \gcd(j, d)$ and $l := \{\nu_2(n), \nu_2(q+1)\}$.*

(1) If $4 \nmid n$ or $q \equiv 1 \pmod 4$, then the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{i=0}^{d'-1} (X^{\frac{n}{d'}} - b^{\frac{n}{d'}} \zeta_{q-1}^{\frac{j+i\cdot(q-1)}{d'}}).$$

(2) If $4 \mid n$ and $q \equiv 3 \pmod 4$, then the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{i=0}^{d'-1} \prod_{\substack{u=1 \\ 2 \nmid u}}^{2^l-1} (X^{\frac{2n}{2^l \cdot d'}} - h_i \cdot (bx)^{\frac{n}{2^l \cdot d'}} (\zeta_{2^{l+1}}^u + \zeta_{2^{l+1}}^{uq}) + (h_i \cdot b^{\frac{n}{2^l \cdot d'}})^2 \zeta_{2^{l+1}}^{u(q+1)}),$$

where for $0 \le i \le d'-1$ the parameter h_i is defined as $h_i = \zeta_{q-1}^{\frac{v_i \cdot (q+1)}{2^{l+1}}}$ and $v_i = \frac{q-1}{2} + \frac{j}{d'} + i\frac{q-1}{d'}$ is even.

The next theorem completes the case $\mathrm{rad}(n) \mid (q-1)$ and gives the factorization of $X^n - a$ over $\mathbb{F}_q$ for all elements $a \in \mathbb{F}_q^*$ such that $a = b^n \cdot \zeta_{q-1}^j$ for an element $b \in \mathbb{F}_q^*$ and an integer $1 \le j \le d^{(1)} - 1$ satisfying $\nu_p(j) \ge \nu_p(d^{(1)})$ for a prime factor p of $d^{(1)}$ with $\nu_p(n) > \nu_p(q-1)$.

Theorem 2.21 ([WY18, Theorem 9]). *Let n be a positive integer satisfying $\mathrm{rad}(n) \mid q-1$ and let $d = \gcd(n, q-1) = d_1 \cdot d_2$ with $\nu_p(n) \le \nu_p(q-1)$ for all $p \mid d_1$ and $\nu_p(n) > \nu_p(q-1)$ for all $p \mid d_2$. Further, let $a = b^n \cdot \zeta_{q-1}^j \in \mathbb{F}_q^*$, where $b \in \mathbb{F}_q$ and $1 \le j \le d-1$ such that $\nu_p(j) \ge \nu_p(d_2)$ for a prime $p \mid d_2$. Set $n' := p_1^{\nu_{p_1}(n)} \cdots p_t^{\nu_{p_t}(n)}$, where $p_1 \cdots p_t = \mathrm{rad}(d_2)$ and $d' := \gcd(n', q-1)$ and $d'^{(2)} := \gcd(n', q^2 - 1)$. Let $l = \min\{\nu_2(n), \nu_2(q+1)\}$ and $w = \gcd(\frac{n}{n'}, j)$. There exists $\beta \in \mathbb{N}$ such that $(\zeta_{q-1}^{j\beta})^{n'} = \zeta_{q-1}^j$.*

(1) If either (j is odd and ($4 \nmid n$ or $q \equiv 1 \pmod 4$)) or (j is even and ($8 \nmid n$ or $q \equiv 1 \pmod 4$)), then the factorization of $X^n - a$ over $\mathbb{F}_q$ into monic irreducible factors is

$$\prod_{i=0}^{w-1} \prod_{t \mid \frac{n'}{d'}} \prod_{\substack{1 \le u \le d' \\ \gcd(u,t)=1}} (X^{\frac{nt}{wn'}} - b^{\frac{nt}{wn'}} \cdot \zeta_{q-1}^{\frac{(tj\beta + u\frac{q-1}{d'}) + i(q-1)}{w}}).$$

(2) If j is odd, $q \equiv 3 \pmod 4$ and $4 \mid n$, then the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{i=0}^{w-1} \prod_{t \mid \frac{n'}{d'}} \prod_{\substack{1 \le u \le d' \\ \gcd(u,t)=1}} \prod_{\substack{1 \le k \le 2^l - 1 \\ 2 \nmid k}} ((X^{\frac{nt}{2^l wn'}})^2 - h_i \cdot (bX)^{\frac{nt}{2^l wn'}} (\zeta_{2^{l+1}}^k + \zeta_{2^{l+1}}^{kq}) + (h_i \cdot b^{\frac{nt}{2^l wn'}})^2 \cdot \zeta_{2^{l+1}}^{k(q+1)}),$$

where for all $0 \le i \le w-1$ the parameter h_i is defined as $h_i = \zeta_{q-1}^{\frac{v_i \cdot (q+1)}{2^{l+1}}}$ and $v_i = \frac{q-1}{2} + \frac{tj\beta}{w} + i\frac{q-1}{w} + \frac{u\frac{q-1}{d'}}{w}$.

(3) If j is even, $q \equiv 3 \pmod 4$ and $8 \mid n$, then the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{i=0}^{w-1} \prod_{\substack{t \mid n' d'^{(2)} \\ t \text{ odd}}} \prod_{\substack{1 \le v \le d' \\ \gcd(v,t)=1}} (X^{\frac{nt}{wn'}} - b^{\frac{nt}{wn'}} \cdot \zeta_{q-1}^{\frac{tj\beta + v \cdot \frac{q-1}{d'} + i(q-1)}{w}})$$

$$\cdot \prod_{i=0}^{w-1} \prod_{t \mid \frac{n'}{d'^{(2)}}} \prod_{u \in \mathcal{U}_t} \left[(X^{\frac{nt}{wn'}} - b^{\frac{nt}{wn'}} \zeta_{q^2-1}^{\frac{tj\beta(q+1) + u\frac{q^2-1}{d'^{(2)}}}{w}} \zeta_{q-1}^{i\frac{q-1}{w}}) \right.$$

$$\left. \cdot (X^{\frac{nt}{wn'}} - b^{\frac{nt}{wn'}} \zeta_{q^2-1}^{\frac{tj\beta(q+1) + qu\frac{q^2-1}{d'^{(2)}}}{w}} \zeta_{q-1}^{i\frac{q-1}{w}}) \right],$$

where

$$\mathcal{U}_t = \{1 \le u \le d'^{(2)} : \gcd(u,t) =, 2^{\min\{\nu_2(\frac{n}{2}), \nu_2(q+1)\}} \nmid u, u < (qu \bmod d'^{(2)})\}.$$

For all positive integers n such that $\mathrm{ord}_{\mathrm{rad}(n)}(q)$ is prime and such that the number of distinct prime factors of n is less than or equal to three [LY18], [SF20] and [Rak+22] determine the explicit factorization of $X^n - a$. No complete factorization for the positive integers n such that $\mathrm{ord}_{\mathrm{rad}(n)}(q)$ is prime had been given so far.

2.3 New results on the factorization of $X^n - a$

In this section we prove Theorem 2.41, the closed formula for the explicit factorization of $X^n - a$ over $\mathbb{F}_q$ for every $a \in \mathbb{F}_q^*$ and every positive integer n such that $\gcd(n, q) = 1$. The element $a = 0$ does not need to be considered since $X^n - 0$ factors as $X^n = (X)^n$ over $\mathbb{F}_q$ for every positive integer n. Thus, let a be an element of $\mathbb{F}_q^*$ throughout this section.

Our proof can be briefly summarized as follows:

(0) We give a new proof for a new closed formula for the factorization of $X^n - a$ for all positive integers n such that $\mathrm{rad}(n) \mid (q-1)$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$), Theorem 2.37.

Additionally, we determine the exact order of the coefficients of the irreducible factors R, which allows us to determine $\mathrm{coeffdeg}_q(R)$ in Step (2).

(1) Let $w = \mathrm{ord}_{\mathrm{rad}(n)}(q)$. If $4 \nmid n$ or $q^w \equiv 1 \pmod 4$, then set $s := w$. Otherwise, set $s := 2w$.

Then Theorem 2.37 yields the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_{q^s}$: $X^n - a = \prod_{R \in \mathcal{R}} R$.

(2) All $R \in \mathcal{R}$ are monic binomials. Thus, $R = X^t - \beta$, for a positive integer t and an element β of $\mathbb{F}_{q^s}$, and $\mathrm{coeffdeg}_q(R)$ equals $\mathrm{ord}_{\mathrm{ord}(\beta)}(q)$. We determine $\mathrm{ord}_{\mathrm{ord}(\beta)}(q)$ explicitly.

(3) We define an equivalence relation $\sim$ on $\mathcal{R}$ as follows: $R \sim R'$ if and only if $R' = R^{(j)}$ for an integer $0 \leq j \leq \mathrm{coeffdeg}_q(R) - 1$.

(4) We determine $\mathcal{R}/\sim$.

(5) With Corollary 2.11 the factorization of $X^n - a$ over $\mathbb{F}_q$ is given by

$$X^n - a = \prod_{R \in \mathcal{R}/\sim} \mathrm{spin}_q[R].$$

A big part of this section is devoted to step (0) (see Section 2.3.1). Recall that the factorization of $X^n - a$ for all positive integers n such that $\mathrm{rad}(n) \mid q - 1$ was given by [WY18] in Theorem 2.19, Theorem 2.20 and Theorem 2.21. However, we use new ideas and a new approach to this problem which allows us to give one closed formula for all positive integers n satisfying $\mathrm{rad}(n) \mid q-1$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$), Theorem 2.37. Only with this new closed formula we obtain the general closed explicit formula for the factorization of $X^n - a$ for every positive integer n such that $\gcd(n, q) = 1$. Theorem 2.37 covers and extends Theorem 2.19 (1), Theorem 2.20 (1) and Theorem 2.21 (1). As can be seen from a direct comparison of Theorem 2.37 and Theorem 2.19 (1), Theorem 2.20 (1) and Theorem 2.21 (1), our new closed formula simplifies the statement and the computation of the factorization of $X^n - a$ for the specified integers n.

Furthermore, determining the order of the coefficients of the irreducible factors is a non-trivial task because the coefficients are products of elements with a non-coprime order. For any $b, c \in \mathbb{F}_q$ such that $\gcd(\mathrm{ord}(b), \mathrm{ord}(c)) = 1$ holds $\mathrm{ord}(bc) = \mathrm{ord}(b) \cdot \mathrm{ord}(c)$. However, in general, if $\gcd(\mathrm{ord}(b), \mathrm{ord}(c)) > 1$, the order of bc can be anything from 1 to $q - 1$.

Note that [MGO15] and [WY18] prove the case $\mathrm{rad}(n) \mid (q-1)$, $4 \mid n$ and $q \equiv 3 \pmod 4$ separately. However, it is not necessary to prove this case separately. We include the positive integers n satisfying $\mathrm{rad}(n) \mid (q-1)$, $4 \mid n$ and $q \equiv 3 \pmod 4$ in our general theorem for all positive integers n such that $\gcd(n, q) = 1$, Theorem 2.41. Thus, Theorem 2.19 (2), Theorem 2.20 (2) and Theorem 2.21 (2) are covered by Theorem 2.41.

The results on the factorization of $X^n - 1$ and $X^n - a$ given in [MGO15] and [WY18] are proved with a top-down approach. More precisely, first, the authors name the irreducible factors of $X^n - a$ over $\mathbb{F}_q$. Then they show that the named polynomials are irreducible over $\mathbb{F}_q$ and factors of $X^n - a$. In the last step, the argument that the sum of the degrees of the named polynomials equals n concludes their proof.

We use a bottom-up approach and prove our results in Section 2.3.1 via induction. More precisely, if $n = p_1 \cdots p_m$ is the factorization of n into (not necessarily distinct) prime factors, we factor

$$X^{p_1} - a,\ (X^{p_2})^{p_1} - a,\ ((X^{p_3})^{p_2})^{p_1} - a, \ldots, (\ldots (X^{p_m})^{p_{m-1}} \ldots)^{p_1} - a$$

inductively. The advantage of this method is that the proof itself shows why the factors are given as they are and it allows us to give one closed formula for the factorization of $X^n - a$.

2.3.1 A new closed formula for the factorization of $X^n - a$ for every positive integer n such that $\mathrm{rad}(n) \mid (q-1)$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$)

In this section let n always be a positive integer such that $\mathrm{rad}(n) \mid q-1$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$). We use an induction on the number of prime factors of the positive integer n to obtain one closed formula for the factorization of $X^n - a$ and the exact order of the irreducible factors' coefficients. Our proof can be summarized as follows:

(1) We determine the order of the roots of the polynomial $X^p - a$ for a prime p such that $p \mid (q-1)$ (Proposition 2.23). There arise the two cases $p \nmid \mathrm{ord}(a)$ and $p \mid \mathrm{ord}(a)$.

(2) We establish a connection between the order of the roots of $X^p - a$ and the factorization of the polynomial $X^{tp} - a$ for an irreducible binomial $X^t - a$ and a prime p such that $p \mid (q-1)$ (see Proposition 2.27 and Theorem 2.32).

(3) By induction, we obtain the factorization of the polynomial $X^{tn} - a$ and the order of the appearing coefficients, for an irreducible binomial $X^t - a$ and a positive integer n such that

- $\mathrm{rad}(n) \mid \mathrm{ord}(a)$ (Theorem 2.33)
- $\gcd(n, \mathrm{ord}(a) \cdot t) = 1$ (Theorem 2.35).

(4) We combine Theorem 2.33 and Theorem 2.35 to a closed formula for the factorization of $X^n - a$ for every positive integer n satisfying $\mathrm{rad}(n) \mid q-1$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$), Theorem 2.37. In this result we also specify the exact order of all appearing coefficients.

The condition $\mathrm{rad}(n) \mid q-1$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$) on the positive integer n comes from the following theorem, Theorem 2.22, by Serret. It characterizes all irreducible binomials $X^t - a$ over $\mathbb{F}_q$, where t is a positive integer and $a \in \mathbb{F}_q^*$. We use it in every induction step to decide on the irreducibility of the appearing factors. The result is a specification of Theorem 2.2 and we give an exact reference to the original result.

Theorem 2.22 ([Ser66, Tome 2, 358, Théorème III], [LN94, Theorem 3.75])**.** *Let $t \geq 2$. Then the binomial $X^t - a$ is irreducible in $\mathbb{F}_q[X]$ if and only if the following three conditions are satisfied:*

(i) $\mathrm{rad}(t) \mid \mathrm{ord}(a)$,

(ii) $\gcd(t, \frac{q-1}{\mathrm{ord}(a)}) = 1$,

(iii) $4 \nmid t$ *or* $q \equiv 1 \pmod 4$.

If $X^t - a$ is irreducible, then its order equals $t \cdot \mathrm{ord}(a)$.

In this section we simply assume that condition (iii) always holds so that we can ignore it for now.

Let p be a prime integer such that $\gcd(p, q) = 1$ and let $a \in \mathbb{F}_q^*$. Then there exist a primitive p-th root of unity ζ_p and an element b such that $b^p = a$ in the algebraic closure $\overline{\mathbb{F}}_q$ of $\mathbb{F}_q$. The polynomial $X^p - a$ decomposes as $\prod_{j=0}^{p-1}(X - \zeta_p^j b)$ in $\overline{\mathbb{F}}_q[X]$. The following proposition gives the order of the roots $\zeta_p^j b$ for the two cases $p \nmid \mathrm{ord}(a)$ and $p \mid \mathrm{ord}(a)$.

Proposition 2.23. *Let $a \in \mathbb{F}_q^*$ and p prime such that $\gcd(p, q) = 1$. Then the following statements hold:*

(i) If $p \nmid \mathrm{ord}(a)$, then there exists a positive integer r such that $(a^r)^p = a$ and $\mathrm{ord}(a^r) = \mathrm{ord}(a)$. If $a = 1$, then $r = 1$, else r satisfies $rp \equiv 1 \pmod{\mathrm{ord}(a)}$. Furthermore, $\mathrm{ord}(\zeta_p^j a^r) = p \cdot \mathrm{ord}(a)$ for all $1 \leq j \leq p-1$.

(ii) If $p \mid \mathrm{ord}(a)$, then every root $b \in \overline{\mathbb{F}}_q$ of the polynomial $X^p - a$ has order $\mathrm{ord}(b) = p \cdot \mathrm{ord}(a)$.

Proof. Note that for any element $b \in \overline{\mathbb{F}}_q$ such that $b^p = a$, the order of b satisfies

$$\mathrm{ord}(b) = \mathrm{ord}(a) \cdot \gcd(\mathrm{ord}(b), p) \in \{\mathrm{ord}(a), \mathrm{ord}(a) \cdot p\}. \tag{2.6}$$

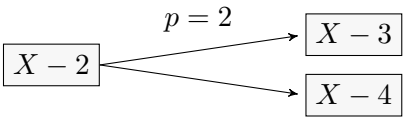
(i) If $p \nmid \mathrm{ord}(a)$ and $\mathrm{ord}(a) > 1$, then there exists $r \in \mathbb{N}$ such that $rp \equiv 1 \pmod{\mathrm{ord}(a)}$ and the element $a^r \in \mathbb{F}_q$ satisfies $(a^r)^p = a$. If $a = 1$, then obviously $(a^1)^p = a$ and we set $r = 1$. Since a^r is an element of the multiplicative group $\langle a \rangle \leq \mathbb{F}_q^*$, its order divides $\mathrm{ord}(a)$. This fact combined with eq. (2.6) yields that $\mathrm{ord}(a^r) = \mathrm{ord}(a)$. For every $1 \leq j \leq p-1$, the element ζ_p^j has order p, which is coprime with $\mathrm{ord}(a)$. Therefore, we can conclude that $\mathrm{ord}(\zeta_p^j a^r) = p \cdot \mathrm{ord}(a)$ for $1 \leq j \leq p-1$.

(ii) If $p \mid \mathrm{ord}(a)$, then with eq. (2.6) also $p \mid \mathrm{ord}(b)$. Suppose that $\mathrm{ord}(b) = \mathrm{ord}(a)$. With the fact that $a = b^p$, we know that $\mathrm{ord}(a) = \frac{\mathrm{ord}(b)}{\gcd(\mathrm{ord}(b),p)} = \frac{\mathrm{ord}(b)}{p} = \frac{\mathrm{ord}(a)}{p}$, a contradiction. Consequently, our assumption must have been false and we have $\mathrm{ord}(b) = \mathrm{ord}(a) \cdot p$. □

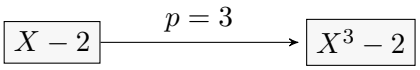
The following example illustrates the implications of Proposition 2.23 for the element $a = 2$ in $\mathbb{F}_7$. All examples in this chapter (2.24, 2.29,2.31, 2.34, 2.36, 2.42, 2.46, 2.64) have been computed with the SageMath program B.7 `Ch2-Example.sage`. Detailed information on how to use the program can be found in Appendix B.

Example 2.24. Let $q = 7$ and $a = 2$. Then $\mathrm{ord}(a) = 3$ and in this example we discuss which information about the polynomial $X^p - 2 \in \mathbb{F}_7[X]$, its roots and their order can be obtained from Proposition 2.23 for the primes $2, 3, 5$ and 19.

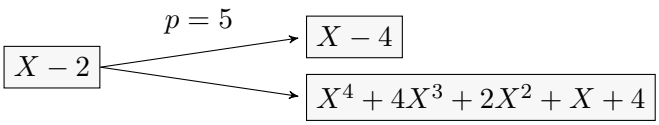
Let $p = 2$, then $p \nmid \mathrm{ord}(2)$ and with Proposition 2.23 (i) there exists a positive integer r such that $r \cdot 2 \equiv 1 \pmod 3$ and $b = 2^r$ is an element of $\mathbb{F}_q$ such that $b^2 = 2$. Since $2 \cdot 2 \equiv 1 \pmod 3$, this element b is $2^2 = 4 \in \mathbb{F}_7$, which satisfies $\mathrm{ord}(4) = \mathrm{ord}(2) = 3$. Since $\zeta_2 = -1 \in \mathbb{F}_7$, the other root of $X^2 - 2$ is $\zeta_2 b = -4 = 3 \in \mathbb{F}_7$ with $\mathrm{ord}(3) = 2 \cdot \mathrm{ord}(2) = 6$. The polynomial $X^2 - 2$ factors as $(X - 4)(X - 3)$ in $\mathbb{F}_7[X]$, because 4 and 3 are both elements of $\mathbb{F}_7$.



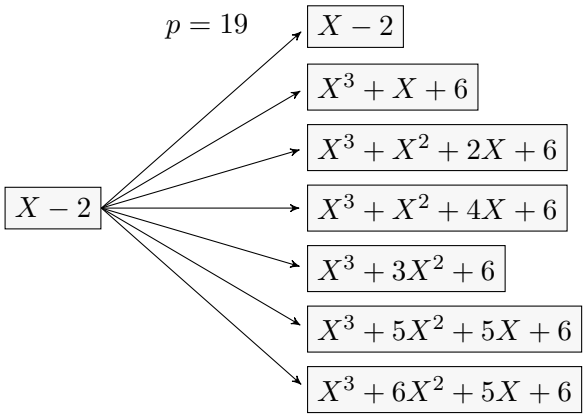
Let $p = 3$, then $p \mid \mathrm{ord}(2)$ and with Proposition 2.23 (ii) all roots of the polynomial have order $3 \cdot \mathrm{ord}(2) = 9$. Since 9 does not divide $7 - 1 = 6$, but divides $7^3 - 1 = 342 = 2 \cdot 3^2 \cdot 19$, all three roots $\zeta_3^j b$ for $0 \leq j \leq 2$ are proper elements of $\mathbb{F}_{7^3}$. Proposition 2.23 (ii) does not specify the element b or the other roots of $X^3 - 2 \in \mathbb{F}_7$. However, from our observations follows that the polynomial $X^3 - 2$ is irreducible over $\mathbb{F}_7$.



Let $p = 5$, then $p \nmid \mathrm{ord}(2)$ and the positive integer r such that $r \cdot 5 \equiv 1 \pmod 3$ is $r = 2$. Thus, $b = 2^2 = 4$ satisfies $4^5 = 2$ and $\mathrm{ord}(4) = \mathrm{ord}(2) = 3$. Since $\mathrm{ord}_5(7) = 4$, all primitive 5-th roots of unity are proper elements of the extension field $\mathbb{F}_{7^4}$. The roots $\zeta_5^j b$ for $1 \leq b \leq 4$ all have order $5 \cdot \mathrm{ord}(2) = 15$. Since $15 \mid 7^4 - 1 = 2^5 \cdot 3 \cdot 5^2$, they are proper elements of $\mathbb{F}_{7^4}$ as well. By polynomial division we obtain that the polynomial $X^5 - 2$ factors as $(X - 4) \cdot (X^4 + 4X^3 + 2X^2 + X + 4)$ over $\mathbb{F}_7$.



Let $p = 19$, then $p \nmid \operatorname{ord}(2)$ and also $p \nmid q - 1$ so that all primitive 19-th roots of unity lie in a true extension field of $\mathbb{F}_7$. With Proposition 2.23 (i) the element $b = 2$ satisfies $b^{19} = 2$, because $19 \equiv 1 \pmod 3$. Obviously, $\operatorname{ord}(b) = \operatorname{ord}(2) = 3$. The other 18 roots $\zeta_{19}^j b$ for $1 \leq j \leq 18$ all have order $19 \cdot 3 = 57$. They are proper elements of $\mathbb{F}_{7^3}$, because $7^3 - 1$ factors as $2 \cdot 3^2 \cdot 19$ and 19 does not divide $7^2 - 1 = 48$. Thus, the polynomial $X^{19} - 2$ factors in $\mathbb{F}_7[X]$ as a product of the linear factor $X - 2 \in \mathbb{F}_7[X]$ and 6 distinct irreducible polynomials of degree 3. In fact, the factorization is $(X-2)\cdot(X^3+X+6)\cdot(X^3+X^2+2X+6)\cdot(X^3+X^2+4X+6)\cdot(X^3+3X^2+6)\cdot(X^3+5X^2+5X+6)\cdot(X^3+6X^2+5X+6)$.



Example 2.24 shows that even though the statement of Proposition 2.23 is quite simple, many different factorizations of $X^p - a$ can result from the two cases $p \nmid \operatorname{ord}(a)$ and $p \mid \operatorname{ord}(a)$.

If $p \mid (q-1)$, then ζ_p is an element of $\mathbb{F}_q$ and for every $0 \leq j \leq p-1$, the root $\zeta_p^j b$ is an element of $\mathbb{F}_q$ if and only if b is an element of $\mathbb{F}_q$. Thus, if one root b of $X^p - a$ is an element of $\mathbb{F}_q$, then all roots $\zeta_p^j b$ for $0 \leq j \leq p-1$ are elements of $\mathbb{F}_q$. Then the polynomial $X^p - a$ decomposes as $\prod_{j=0}^{p-1}(X - \zeta_p^j b)$ over $\mathbb{F}_q$. Furthermore, with Proposition 2.23 there exists at least one root of $X^p - a$ of order $p \cdot \operatorname{ord}(a)$ and the following corollary of Proposition 2.23 holds:

Corollary 2.25. *Let p be a prime satisfying $p \mid (q-1)$. Then there exists $b \in \mathbb{F}_q$ such that $b^p = a$ if and only if $p \cdot \operatorname{ord}(a) \mid (q-1)$.*

Example 2.26 (Example 2.24 continued). For $p = 2$ and $p = 3$ holds $p \mid (q-1) = 7 - 1$ in Example 2.24. If $p = 2$, then $p \cdot \operatorname{ord}(a) = p \cdot \operatorname{ord}(2) = 2 \cdot 3$ divides $7 - 1$ and we see that both elements $b = 4$ and $\zeta_2 b = 3$ are elements of $\mathbb{F}_7$. If $p = 3$, then $p \cdot \operatorname{ord}(a) = 3 \cdot 3$ does not divide $7 - 1$ and all three roots of the polynomial $X^3 - 2 \in \mathbb{F}_7[X]$ lie in a true extension field of $\mathbb{F}_7$.

With Theorem 2.22 the irreducibility of $X^t - a$, for $t \in \mathbb{N}$ and $a \in \mathbb{F}_q^*$, relies solely on the order of a. This is why Proposition 2.23 and Corollary 2.25 are the key components of the proofs in this section.

In the next proposition, we combine Corollary 2.25 and Theorem 2.22 to obtain a result which can be applied recursively. More precisely, given a prime p such that $p \mid (q-1)$ and an irreducible binomial $X^t - a$ over $\mathbb{F}_q$, we show that $X^{tp} - a$ is irreducible if and only if $p \cdot \operatorname{ord}(a) \nmid (q-1)$.

Proposition 2.27. *Let $a \in \mathbb{F}_q^*$, p be a prime such that $p \mid (q-1)$ and let $t \in \mathbb{N}$ be such that $X^t - a$ is irreducible over $\mathbb{F}_q$. Further, let $4 \nmid tp$ or $q \equiv 1 \pmod 4$. Then $X^{tp} - a$ is irreducible over $\mathbb{F}_q$ if and only if $p \cdot \operatorname{ord}(a) \nmid (q-1)$.*

Proof. We show that $X^{tp} - a$ is irreducible over $\mathbb{F}_q$ if and only if there does not exist $b \in \mathbb{F}_q$ such that $b^p = a$. Then Corollary 2.25 completes the statement.

Suppose that there exists $b \in \mathbb{F}_q$ such that $b^p = a$. Then the polynomial $X^{tp} - a$ in $\mathbb{F}_q[X]$ satisfies $X^{tp} - a = \prod_{j=0}^{p-1}(X^t - \zeta_p^j b)$ and it is reducible.

Suppose that there does not exist $b \in \mathbb{F}_q$ such that $b^p = a$. Then all roots of $X^p - a$ lie in proper extension fields of $\mathbb{F}_q$. Let $b \in \mathbb{F}_q(b)$ be a root of $X^p - a$. Then since $\zeta_p \in \mathbb{F}_q$, we have $\zeta_p^j b \in \mathbb{F}_q(b)$ for all $1 \leq j \leq p-1$. Vice versa, $b = \zeta_p^{-j} \cdot \zeta_p^j b \in \mathbb{F}_q(\zeta_p^j b)$. Thus, the degree of the minimal polynomial of every root of $X^p - a$ is equal to $[\mathbb{F}_q(b) : \mathbb{F}_q]$. Since $X^p - a$ is a product of the minimal polynomials of its roots over $\mathbb{F}_q$, there exists a positive integer m such that $p = m \cdot [\mathbb{F}_q(b) : \mathbb{F}_q]$. The two facts that p is prime and b is not an element of $\mathbb{F}_q$ imply that $[\mathbb{F}_q(b) : \mathbb{F}_q] = p$ and $X^p - a$ is irreducible over $\mathbb{F}_q$.

Theorem 2.22 yields that $p \mid \mathrm{ord}(a)$ and $\gcd(p, \frac{q-1}{\mathrm{ord}(a)}) = 1$. Furthermore, since $X^t - a$ is irreducible over $\mathbb{F}_q$, we have $\mathrm{rad}(t) \mid \mathrm{ord}(a)$, $\gcd(t, \frac{q-1}{\mathrm{ord}(a)}) = 1$. As a direct consequence we obtain that $\mathrm{rad}(tp) \mid \mathrm{ord}(a)$ and also $\gcd(tp, \frac{q-1}{\mathrm{ord}(a)}) = 1$. Since we assumed that $4 \nmid tp$ or $q \equiv 1 \pmod 4$, the third condition of Theorem 2.22 is satisfied and we can conclude that $X^{tp} - a$ is irreducible over $\mathbb{F}_q$. □

The following corollary states the useful fact that for a prime p such that $p \mid t$ the polynomial $X^{tp} - a$ is irreducible if $X^t - a$ is irreducible and ($4 \nmid tp$ or $q \equiv 1 \pmod 4$).

Corollary 2.28. *Let $a \in \mathbb{F}_q^*$, p be a prime such that $p \mid (q-1)$ and let $t \in \mathbb{N}$ such that $X^t - a$ is irreducible over $\mathbb{F}_q$. Further, let $4 \nmid tp$ or $q \equiv 1 \pmod 4$. Then if p divides t the polynomial $X^{tp} - a$ is irreducible over $\mathbb{F}_q$.*

Proof. Since $X^t - a$ is irreducible over $\mathbb{F}_q$, Theorem 2.22 implies that $\gcd(t, \frac{q-1}{\mathrm{ord}(a)}) = 1$. If $p \mid t$, then also $\gcd(p, \frac{q-1}{\mathrm{ord}(a)}) = 1$ and $p \cdot \mathrm{ord}(a)$ cannot divide $(q-1)$. Proposition 2.27 completes our proof. □

Corollary 2.28 simplifies the check for irreducibility significantly, because it is not necessary to compare the properties of t and $\mathrm{ord}(a)$ as specified in Theorem 2.22 (i) and (ii), if p is a prime factor of t.

Example 2.29. Let $a = 2 \in \mathbb{F}_7$ and $t = 3$. From Example 2.24 we know that the binomial $X^3 - 2$ is irreducible over $\mathbb{F}_7$. Let $p = 3$, then $3 \cdot \mathrm{ord}(2) = 9$ does not divide $7 - 1$. Thus, with Proposition 2.27 the polynomial $X^{3 \cdot 3} - 2$ is irreducible. However, with Corollary 2.28 we could have seen directly that $X^9 - 2$ is irreducible, because $p \mid t$.

$$\boxed{X - 2} \xrightarrow{p=3} \boxed{X^3 - 2} \xrightarrow{p=3} \boxed{X^9 - 2}$$

From Example 2.24 we know that the order of the roots of $X^3 - 2$ is $9 = 3 \cdot \mathrm{ord}(a)$. Here, all nine roots of $X^9 - 2$ are proper elements of F_{7^9} and their order equals $27 = 9 \cdot \mathrm{ord}(a)$ which follows directly from Theorem 2.22.

The following proposition shows that even if $X^{tp} - a$ is reducible, under certain conditions the factorization of $X^{tp} - a$ is determined by the factorization of $X^p - a$ over $\mathbb{F}_q$.

Proposition 2.30. *Let $a \in \mathbb{F}_q^*$, p be a prime such that $\gcd(p, q) = 1$ and let t be a positive integer such that $X^t - a$ is irreducible over $\mathbb{F}_q$. Suppose that the factorization of $X^p - a$ into monic irreducible factors over $\mathbb{F}_q$ is given by $\prod_{R \in \mathcal{R}} R$. Then the factorization of $X^{tp} - a$ into monic irreducible factors over $\mathbb{F}_q$ is given by $\prod_{R \in \mathcal{R}} R(X^t)$ if and only if $\mathrm{ord}_{p \cdot \mathrm{ord}(a)}(q) = 1$ or $\gcd(t, \frac{q^{\mathrm{ord}_{p \cdot \mathrm{ord}(a)}(q)} - 1}{(q-1) \cdot p}) = 1$.*

Proof. If $t = 1$, then $X^{tp} - a = X^p - a$ and the factorizations are obviously the same. Since $\gcd(1, \frac{q^{\mathrm{ord}_{p \cdot \mathrm{ord}(a)}(q)} - 1}{(q-1) \cdot p}) = 1$, the statement of Proposition 2.30 holds for $t = 1$.

Let $t \geq 2$. Since $X^t - a$ is irreducible, Theorem 2.22 implies that

(I) $\operatorname{rad}(t) \mid \operatorname{ord}(a)$,

(II) $\gcd(t, \frac{q-1}{\operatorname{ord}(a)}) = 1$,

(III) and $4 \nmid t$ or $q \equiv 1 \pmod 4$.

The factorization of $X^{tp} - a = (X^t)^p - a$ over $\mathbb{F}_q$ is given by $\prod_{R\in\mathcal{R}} R(X^t)$ if and only if $R(X^t)$ is irreducible for every $R \in \mathcal{R}$. Let $R \in \mathcal{R}$ be a monic irreducible factor of $X^p - a$ over $\mathbb{F}_q$. By Theorem 2.2 $R(X^t)$ is irreducible over $\mathbb{F}_q$ if and only if

(i) $\operatorname{rad}(t) \mid \operatorname{ord}(R)$,

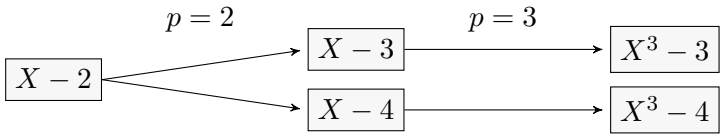
(ii) $\gcd(t, \frac{q^{\deg(R)}-1}{\operatorname{ord}(R)}) = 1$,

(iii) and $4 \nmid t$ or $q^{\deg(R)} - 1 \equiv 1 \pmod 4$.

If $4 \mid t$ then $q \equiv 1 \pmod 4$ (with (III)), which implies that $q^{\deg(R)} \equiv 1 \pmod 4$ and (iii) holds. Furthermore, from Proposition 2.23 follows that $\operatorname{ord}(R) \in \{\operatorname{ord}(a), p \cdot \operatorname{ord}(a)\}$. Consequently, (I) implies (i).

Moreover, if $\deg(R) = 1$, then $\frac{q-1}{\operatorname{ord}(R)}$ divides $\frac{q-1}{\operatorname{ord}(a)}$. Thus, (II) implies that $\gcd(t, \frac{q-1}{\operatorname{ord}(R)}) = 1$ and $R(X^t)$ is irreducible. If $\operatorname{ord}_{p\cdot\operatorname{ord}(a)}(q) = 1$ then all irreducible factors of $X^p - a$ are of degree 1. If $\operatorname{ord}_{p\cdot\operatorname{ord}(a)}(q) > 1$, then all irreducible factors $R \in \mathcal{R}$ such that $\deg(R) > 1$ satisfy $\operatorname{ord}(R) = \operatorname{ord}(a) \cdot p$ and $k := \deg(R) = \operatorname{ord}_{p\cdot\operatorname{ord}(a)}(q)$. We can write $\frac{q^k-1}{\operatorname{ord}(R)} = \frac{q-1}{\operatorname{ord}(a)} \cdot \frac{q^k-1}{(q-1)\cdot p}$ and (ii) does not hold if and only if $\gcd(t, \frac{q^k-1}{(q-1)\cdot p}) > 1$. □

Example 2.31. We consider again the irreducible binomial $X^3 - 2 \in \mathbb{F}_7[X]$. Let $p = 2$ and recall that $X^2 - 2$ factored as $(X - 4) \cdot (X - 3)$ over $\mathbb{F}_7$ (see Example 2.24). Furthermore, $p \cdot \operatorname{ord}(a) = 2 \cdot 3 \mid 7 - 1$ and $\operatorname{ord}_6(7) = 1$. Thus, with Proposition 2.30 the binomial $X^{3\cdot 2} - 2$ factors as $(X^3 - 4) \cdot (X^3 - 3)$ over $\mathbb{F}_7$.



Since the orders of 4 and 3 are $\operatorname{ord}(4) = 3$ and $\operatorname{ord}(3) = 6$, Theorem 2.22 implies that the order of $X^3 - 4$ equals $3 \cdot \operatorname{ord}(4) = 9$ and the order of $X^3 - 3$ equals $3 \cdot \operatorname{ord}(3) = 18$.

However, the factorization of $X^{tp} - a$ is not always given by $\prod_{R\in\mathcal{R}} R(X^t)$ if $X^p - a$ factors as $\prod_{R\in\mathcal{R}} R$ over $\mathbb{F}_q$. For example let $p = 19$. Then $\operatorname{ord}_{19\cdot 3}(7) = 3 > 1$ and $\gcd(3, \frac{7^3-1}{(7-1)\cdot 19}) = \gcd(3, \frac{2\cdot 3^2\cdot 19}{2\cdot 3\cdot 19}) = 3 > 1$. Thus, the condition in Proposition 2.30 is not satisfied. With Example 2.24 we know that $X^{19} - 2$ factors as

$$\begin{aligned}\prod_{R\in\mathcal{R}} R = {} & (X - 2) \cdot (X^3 + X + 6) \cdot (X^3 + X^2 + 2X + 6) \cdot (X^3 + X^2 + 4X + 6) \\ & \cdot (X^3 + 3X^2 + 6) \cdot (X^3 + 5X^2 + 5X + 6) \cdot (X^3 + 6X^2 + 5X + 6)\end{aligned}$$

over $\mathbb{F}_7$ and the factorization of $X^{3\cdot 19} - 2 = X^{57} - 2$ over $\mathbb{F}_7$ is

$$\begin{aligned}& (X^3 - 2) \cdot (X^3 + 3X + 5) \cdot (X^3 + 5X + 5) \cdot (X^3 + 6X + 5) \\ & \cdot (X^3 + X^2 + X + 5) \cdot (X^3 + X^2 + 3X + 5) \cdot (X^3 + X^2 + 6X + 5) \\ & \cdot (X^3 + 2X^2 + 3X + 5) \cdot (X^3 + 2X^2 + 4X + 5) \cdot (X^3 + 2X^2 + 5X + 5) \\ & \cdot (X^3 + 3X^2 + 2X + 5) \cdot (X^3 + 3X^2 + 3X + 5) \cdot (X^3 + 4X^2 + 2X + 5) \\ & \cdot (X^3 + 4X^2 + 5X + 5) \cdot (X^3 + 4X^2 + 6X + 5) \cdot (X^3 + 5X^2 + 4X + 5) \\ & \cdot (X^3 + 5X^2 + 6X + 5) \cdot (X^3 + 6X^2 + X + 5) \cdot (X^3 + 6X^2 + 5X + 5),\end{aligned}$$

which is not equal to $\prod_{R\in\mathcal{R}} R(X^3)$.

The following theorem gives the factorization of $X^{tp} - a$ for an irreducible binomial $X^t - a$ and all prime factors p of $q - 1$ such that $X^{tp} - a$ is reducible over $\mathbb{F}_q$.

Theorem 2.32. *Let $a \in \mathbb{F}_q^*$, p be a prime such that $p \cdot \mathrm{ord}(a) \mid (q-1)$ and let $t \in \mathbb{N}$ be such that $X^t - a$ is irreducible over $\mathbb{F}_q$. Further, let $4 \nmid tp$ or $q \equiv 1 \pmod 4$. Then there exists $b \in \mathbb{F}_q$ such that $b^p = a$ and the factorization of $X^{tp} - a$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{j=0}^{p-1} (X^t - \zeta_p^j b).$$

(i) If $p \nmid \mathrm{ord}(a)$, then $b = a^r$ and $\mathrm{ord}(b) = \mathrm{ord}(a)$ for a positive integer r satisfying $r = 1$ if $a = 1$ or $rp \equiv 1 \pmod{\mathrm{ord}(a)}$ otherwise.

(ii) If $p \mid \mathrm{ord}(a)$, $\mathrm{ord}(b) = p \cdot \mathrm{ord}(a)$.

Furthermore, $\mathrm{ord}(\zeta_p^j b) = p \cdot \mathrm{ord}(a)$ for all $1 \leq j \leq p-1$.

Proof. Since $p \cdot \mathrm{ord}(a) \mid (q-1)$, Proposition 2.27 implies that the polynomial $X^{tp} - a$ is reducible. Furthermore, $\mathrm{ord}_{p \cdot \mathrm{ord}(a)}(q) = 1$ and from Proposition 2.30 follows that the factorization of $X^{tp} - a$ is given by $\prod_{R \in \mathcal{R}} R(X^t)$, where $\prod_{R \in \mathcal{R}} R$ is the factorization of $X^p - a$ over $\mathbb{F}_q$. With Corollary 2.25 there exists an element $b \in \mathbb{F}_q$ such that $b^p = a$ and since $p \mid (q-1)$, the p-th primitive root of unity ζ_p is an element of $\mathbb{F}_q$. Thus, $X^p - a = \prod_{j=0}^{p-1}(X - \zeta_p^j b)$ and the factorization of $X^{tp} - a$ over $\mathbb{F}_q$ is $\prod_{j=0}^{p-1}(X^t - \zeta_p^j b)$. Statements (i) and (ii) follow directly from Proposition 2.23. □

Note that Theorems 2.27 and 2.32 allow us to factor the polynomial $X^{tp} - a$ simply by examining the order of the element a and comparing it with p and $q - 1$. This is why statements (i) and (ii) in Theorem 2.32, which specify the orders of the new appearing coefficients, are of great importance.

Theorems 2.27 and 2.32 can be applied recursively and we use them for the induction step in the proofs of the next two theorems, Theorem 2.33 and Theorem 2.35. These give the factorization of the polynomial $X^{tn} - a$ for the two cases $\mathrm{rad}(n) \mid \mathrm{ord}(a)$ and $\gcd(n, \mathrm{ord}(a)) = 1$, where $X^t - a$ is irreducible over $\mathbb{F}_q$. Additionally, they specify the order of all the appearing coefficients.

Theorem 2.33. *Let $a \in \mathbb{F}_q^*$, $t \in \mathbb{N}$ such that $X^t - a$ is irreducible over $\mathbb{F}_q$ and let $n \in \mathbb{N}$ such that $\mathrm{rad}(n) \mid \mathrm{ord}(a)$. Further, let $4 \nmid tn$ or $q \equiv 1 \pmod 4$. Set $d := \gcd(n, \frac{q-1}{\mathrm{ord}(a)})$. Then there exists $b \in \mathbb{F}_q$ such that $b^d = a$ and the factorization of $X^{tn} - a$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{j=0}^{d-1} (X^{t \cdot \frac{n}{d}} - \zeta_d^j b),$$

and $\mathrm{ord}(\zeta_d^j b) = \mathrm{ord}(a) \cdot d$ for all $0 \leq j \leq d-1$.

Proof. We use the fact that every positive integer n can be written as a product of prime factors and prove the statement by induction on the number of prime factors of n. Base case: If $n = p$ for a prime p such that $p \mid (q-1)$ and $p \mid \mathrm{ord}(a)$, the statement follows directly from Theorem 2.32 (i).

Induction hypothesis: Suppose that the statement of Theorem 2.35 holds for n, d, b such that $\mathrm{rad}(n) \mid \mathrm{ord}(a), d = \gcd(n, \frac{q-1}{\mathrm{ord}(a)})$ and $b^d = a$.

Induction step ($n \to np$): Let p be a prime such that $p \mid \mathrm{ord}(a)$, then

$$X^{tnp} - a = \prod_{j=0}^{d-1} (X^{t \cdot \frac{n}{d} \cdot p} - \zeta_d^j b). \tag{2.7}$$

If dp does not divide $\frac{q-1}{\mathrm{ord}(a)}$, then with Proposition 2.27 the polynomial $(X^{t \cdot \frac{n}{d} \cdot p} - \zeta_d^j b)$ is irreducible over $\mathbb{F}_q$ for every $0 \leq j \leq d-1$. Furthermore, the greatest common divisor of np and $\frac{q-1}{\mathrm{ord}(a)}$ is also d. Consequently, (2.7) is the factorization of $X^{tnp} - a$ into monic irreducible factors over $\mathbb{F}_q$ and the statement of Theorem 2.33 holds.

If dp divides $\frac{q-1}{\mathrm{ord}(a)}$, then $\tilde{d} := \gcd(np, \frac{q-1}{\mathrm{ord}(a)}) = dp$ and there exists $\zeta_{\tilde{d}} \in \mathbb{F}_q$. Furthermore, for every $0 \leq j \leq d-1$ holds $\mathrm{ord}(\zeta_d^j b) \cdot p = \mathrm{ord}(a) \cdot d \cdot p \mid (q-1)$ and the polynomial $(X^{t \cdot \frac{n}{d}} - \zeta_d^j b)$ is reducible. Theorem 2.32 yields that there exists an element $c_j \in \mathbb{F}_q$ such that $c_j^p = \zeta_d^j b$ and that the factorization of $(X^{t \cdot \frac{n}{d} \cdot p} - \zeta_d^j b)$ into monic irreducible factors over $\mathbb{F}_q$ is given by $\prod_{i=0}^{p-1} (X^{t \cdot \frac{n}{d}} - \zeta_p^i c_j)$, where $\mathrm{ord}(\zeta_p^i c_j) = \mathrm{ord}(\zeta_d^j b) \cdot p = \mathrm{ord}(a) \cdot d \cdot p = \mathrm{ord}(a) \cdot \tilde{d}$. Thus, the factorization of $X^{tnp} - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$X^{tnp} - a = \prod_{j=0}^{d-1} \prod_{i=0}^{p-1} (X^{t \cdot \frac{n}{d}} - \zeta_p^i c_j) = \prod_{j=0}^{d-1} \prod_{i=0}^{p-1} (X^{t \cdot \frac{np}{d}} - \zeta_p^i c_j). \tag{2.8}$$

Moreover, $c := c_0$ satisfies $c^{dp} = (c^p)^d = b^d = a$ and the factorization of $X^{tnp} - a$ into monic irreducible factors over $\mathbb{F}_q$ can also be written as:

$$X^{tnp} - a = \prod_{j=0}^{\tilde{d}} (X^{t \cdot \frac{np}{\tilde{d}}} - \zeta_{\tilde{d}}^j c). \tag{2.9}$$

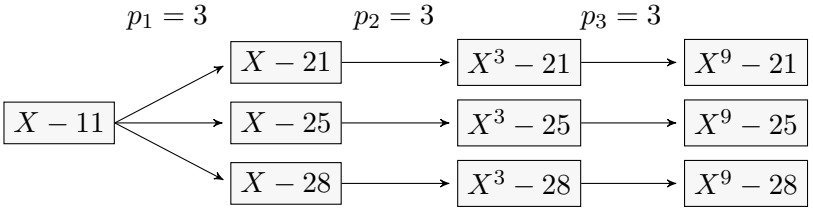
Then for every $0 \leq j \leq \tilde{d} - 1$ holds $\mathrm{ord}(\zeta_{\tilde{d}}^j c) = \mathrm{ord}(a) \cdot \tilde{d}$, because every factor in (2.9) corresponds to a factor in (2.8). □

Example 2.34. Let $q = 37$, then $q - 1 = 36 = 2^2 \cdot 3^3$ and $q \equiv 1 \pmod 4$. We select $a = 11$ with $\mathrm{ord}(a) = 6$ and $n = 108 = 2^2 \cdot 3^2$. In this example we present the inductive character of our proof. Thus, let $p_1 = p_2 = p_3 = 3$ and $p_4 = p_5 = 2$ and we factor $((((X^{p_5})^{p_4})^{p_3})^{p_2})^{p_1} - 11$ recursively.

(p_1) The factorization starts with the binomial $X^3 - 11 \in \mathbb{F}_{37}[X]$. Then $t = 1$ and since $p_1 \cdot \mathrm{ord}(a) = 18$ divides $q - 1$, the factorization is given with Theorem 2.32 as $\prod_{j=0}^{2} (X - \zeta_3^j b)$, where $b \in \mathbb{F}_{37}$ satisfies $b^3 = 11$. A primitive 3rd-root of unity is given by $\zeta_3 = 10$ and $b = 21 \in \mathbb{F}_{37}$ satisfies $21^3 = 11$. Thus, $\prod_{j=0}^{2} (X - \zeta_3^j b) = (X - 21) \cdot (X - 25) \cdot (X - 28)$. All three roots have order $3 \cdot \mathrm{ord}(a) = 18$.

(p_2) Next, we factor $(X^3 - 21) \cdot (X^3 - 25) \cdot (X^3 - 28)$. Again $t = 1$ and for all factors $X^3 - c$ the order $p_2 \cdot \mathrm{ord}(c) = 3 \cdot 18 = 54$ does not divide $q - 1 = 36$. Thus, the factors are all irreducible over $\mathbb{F}_{37}$ by Proposition 2.27.

(p_3) We consider $(X^{3 \cdot 3} - 21) \cdot (X^{3 \cdot 3} - 25) \cdot (X^{3 \cdot 3} - 28)$. Here we have $t = 3$ and $p_3 \mid t$. Thus, with Corollary 2.28 follows that all factors are irreducible over $\mathbb{F}_{37}$.



(p_4) We factor $(X^{9\cdot 2} - 21) \cdot (X^{9\cdot 2} - 25) \cdot (X^{9\cdot 2} - 28)$. Recall that $\operatorname{ord}(21) = \operatorname{ord}(25) = \operatorname{ord}(28) = 18$ and $p_4 \cdot 18 = 36$ divides $q-1$. Consequently, all three binomials $X^{9\cdot 2} - c$ for $c \in \{21, 25, 28\}$ are reducible and the factorization is given by Theorem 2.32: $\prod_{j=0}^{1}(X^9 - \zeta_2^j b)$, where $b \in \mathbb{F}_{37}$ satisfies $b^2 = c$. A primitive 2nd-root of unity is given by $\zeta_2 = -1 = 36$. Furthermore, $13^2 = 21, 5^2 = 25, 18^2 = 28$. Thus, the factorization of $X^{3^3\cdot 2} - 11$ is

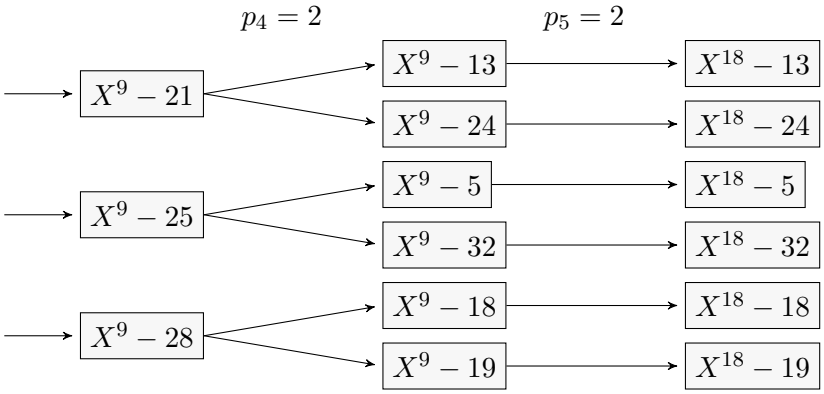
$$(X^9 - 13) \cdot (X^9 - 24) \cdot (X^9 - 5) \cdot (X^9 - 32) \cdot (X^9 - 18) \cdot (X^9 - 19),$$

where $\operatorname{ord}(13) = \operatorname{ord}(24) = \operatorname{ord}(5) = \operatorname{ord}(32) = \operatorname{ord}(18) = \operatorname{ord}(19) = 36$.

(p_5) Since $p_5 \cdot 36 = 72$ does not divide $q-1$, all binomials $X^{9\cdot 2} - c$ for $c \in \{13, 24, 5, 32, 18, 19\}$ are irreducible and the complete factorization of $X^n - 11$ is given by:

$$(X^{18} - 13) \cdot (X^{18} - 24) \cdot (X^{18} - 5) \cdot (X^{18} - 32) \cdot (X^{18} - 18) \cdot (X^{18} - 19).$$

With Theorem 2.22 the order of all the irreducible binomials is $18 \cdot 36 = 648$.



Now lets apply Theorem 2.33 directly. The parameter $d = \gcd(n, \frac{q-1}{\operatorname{ord}(a)}) = \gcd(2^2 \cdot 3^3, \frac{36}{6})$ equals 6. The element $5 \in \mathbb{F}_{37}$ satisfies $5^6 = 11$ and a primitive 6-th root of unity is given by 11 itself. With Theorem 2.33 the factorization of $X^{108} - 11$ over $\mathbb{F}_{37}$ is given by

$$\begin{aligned}&\prod_{j=0}^{5}(X^{18} - 11^j \cdot 5)\\ &= (X^{18} - 5) \cdot (X^{18} - 18) \cdot (X^{18} - 13) \cdot (X^{18} - 32) \cdot (X^{18} - 19) \cdot (X^{18} - 24).\end{aligned}$$

This is obviously the same factorization as above. Furthermore, the application of Theorem 2.33 is a lot easier than a recursive factorization by hand.

As in the proof of Theorem 2.33, we obtain the factorization of $X^{tn} - a$ for the positive integers n such that $\gcd(n, \operatorname{ord}(a)) = 1$ by induction on the number of prime factors of n. Additionally, we impose the condition $\gcd(n, t) = 1$ on n because otherwise the polynomials would not be reducible in every induction step (see Corollary 2.28). Note that for $a = 1$ holds $\gcd(n, \operatorname{ord}(a)) = 1$ for every positive integer n and therefore, the structure of the factorization of $X^n - 1$ in Theorem 2.35 resembles the factorization given in Theorem 2.12.

Theorem 2.35. *Let $a \in \mathbb{F}_q^*$, $t \in \mathbb{N}$ such that $X^t - a$ is irreducible over $\mathbb{F}_q$ and let $n \in \mathbb{N}$ such that $\operatorname{rad}(n) \mid (q-1)$, $\gcd(n, \operatorname{ord}(a) \cdot t) = 1$ and $(4 \nmid tn$ or $q \equiv 1 \pmod 4)$. Set $d := \gcd(n, \frac{q-1}{\operatorname{ord}(a)}) = \gcd(n, q-1)$ and select $r \in \mathbb{N}$ such that $r = 1$ if $a = 1$ or $rn \equiv 1 \pmod{\operatorname{ord}(a)}$ otherwise. Then the factorization of $X^{tn} - a$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{v \mid \frac{n}{d}} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{d-1} (X^{tv} - \zeta_d^j a^{rv}), \tag{2.10}$$

and for all applicable v and j holds

(i) $(\zeta_d^j a^{rv})^{\frac{n}{v}} = a$,

(ii) $\operatorname{ord}(\zeta_d^j a^{rv}) = \operatorname{ord}(a) \cdot \frac{d}{\gcd(j,d)}$.

Proof. Base case: If $n = p$, then the statement follows directly from Theorem 2.32 (ii).

Induction hypothesis: Suppose that the statement of Theorem 2.35 holds for n, d, r such that $\operatorname{rad}(n) \mid (q-1), \gcd(n, \operatorname{ord}(a) \cdot t) = 1, d = \gcd(n, q-1)$ and either $a = 1$ and $r = 1$ or $rn \equiv 1 \pmod{\operatorname{ord}(a)}$.

Induction step $(n \to np)$: Let p be a prime such that $p \mid (q-1)$ and $p \nmid \operatorname{ord}(a) \cdot t$. If $a = 1$ then set $\tilde{r} = 1$ so that $(a^{\tilde{r}})^p = a$. Otherwise there exists a positive integer r' such that $r'p \equiv 1 \pmod{\operatorname{ord}(a)}$. Since $\operatorname{ord}(a^r) = \operatorname{ord}(a)$, this implies that $a^{rr'p} = (a^r)^{r'p} = a^r$. We set $\tilde{r} := rr'$ and together with (2.10) we obtain:

$$X^{tnp} - a = \prod_{v \mid \frac{n}{d}} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{d-1} (X^{tvp} - \zeta_d^j a^{\tilde{r}pv}), \tag{2.11}$$

where $\tilde{r}$ satisfies $r = 1$ if $a = 1$ or $\tilde{r} \cdot np = rn \cdot r'p \equiv 1 \pmod{\operatorname{ord}(a)}$ otherwise. Furthermore, we have $\operatorname{ord}(a^{\tilde{r}pv}) = \operatorname{ord}(a)$, because $(a^{\tilde{r}pv})^{\frac{n}{v}} = a$. It remains to discuss which of the factors $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ are reducible. For this we need to discern the two cases $\gcd(np, q-1) = d$ and $\gcd(np, q-1) = d \cdot p$.

Case $\gcd(np, q-1) = d$: If $\gcd(np, q-1) = d$, then $p \mid n$ but $dp \nmid q-1$ and also $\operatorname{ord}(a) \cdot d \cdot p \nmid (q-1)$. Let $v \mid \frac{n}{d}$ and $0 \le j \le d-1$ such that $\gcd(j, v) = 1$. Then if $p \mid v$, the polynomial $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ is irreducible over $\mathbb{F}_q$ with Corollary 2.28. Thus, the following expression is a factorization into irreducible factors over $\mathbb{F}_q$, where we set $\tilde{v} := vp$:

$$\prod_{\substack{v \mid \frac{n}{d} \\ p \mid v}} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{d-1} (X^{tvp} - \zeta_d^j a^{\tilde{r}vp}) = \prod_{\substack{\tilde{v} \mid \frac{np}{d} \\ p^2 \mid \tilde{v}}} \prod_{\substack{j=0 \\ \gcd(j,\tilde{v})=1}}^{d-1} (X^{t\tilde{v}} - \zeta_d^j a^{\tilde{r}\tilde{v}}), \tag{2.12}$$

where $\gcd(j, v) = 1$ if and only if $\gcd(j, \tilde{v}) = \gcd(j, vp) = 1$, because $p \mid v$. If $p \nmid v$, then with Proposition 2.27 the polynomial $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ is irreducible if and only if $\operatorname{ord}(\zeta_d^j a^{\tilde{r}vp}) \cdot p = \operatorname{ord}(a) \cdot \frac{d}{\gcd(j,d)} \cdot p$ does not divide $(q-1)$. This is the case if and only if $p \nmid j$, because $\operatorname{ord}(a) \cdot d$ divides $(q-1)$ but $\operatorname{ord}(a) \cdot d \cdot p$ does not. Consequently, the following expression is a factorization into irreducible factors over $\mathbb{F}_q$, where we set $\tilde{v} := vp$:

$$\prod_{\substack{v \mid \frac{n}{d} \\ p \nmid v}} \prod_{\substack{j=0 \\ \gcd(j,v)=1 \\ p \mid j}}^{d-1} (X^{tvp} - \zeta_d^j a^{\tilde{r}vp}) = \prod_{\substack{\tilde{v} \mid \frac{np}{d} \\ \nu_p(\tilde{v})=1}} \prod_{\substack{j=0 \\ \gcd(j,\tilde{v})=1}}^{d-1} (X^{t\tilde{v}} - \zeta_d^j a^{\tilde{r}\tilde{v}}). \tag{2.13}$$

Let $0 \le j \le d-1$ such that $p \mid j$. Then the polynomial $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ is reducible and $\zeta_d^{\frac{j}{p}} a^{\tilde{r}v}$ is an element in $\mathbb{F}_q$ satisfying $(\zeta_d^{\frac{j}{p}} a^{\tilde{r}v})^p = \zeta_d^j a^{\tilde{r}vp}$. Let $\zeta_p = \zeta_d^{\frac{d}{p}}$, then from Theorem 2.32 follows that its factorization into irreducible factors over $\mathbb{F}_q$ is

$$\prod_{i=0}^{p-1} (X^{tv} - \zeta_p^i \zeta_d^{\frac{j}{p}} a^{\tilde{r}v}) = \prod_{i=0}^{p-1} (X^{tv} - \zeta_d^{\frac{d}{p} \cdot i + \frac{j}{p}} a^{\tilde{r}v}).$$

With the fact that $\{0 \le j \le d-1 : p \mid j\} = \{p\tilde{j} : 0 \le \tilde{j} \le \frac{d}{p} - 1\}$, the following expression is a factorization into irreducible factors over $\mathbb{F}_q$:

$$\prod_{\substack{v \mid \frac{n}{d} \\ p \nmid v}} \prod_{\tilde{j}=0}^{\frac{d}{p}-1} \prod_{i=0}^{p-1} (X^{tv} - \zeta_d^{\frac{d}{p} \cdot i + \tilde{j}} a^{\tilde{r}v}) = \prod_{\substack{v \mid \frac{np}{d} \\ p \nmid v}} \prod_{j=0}^{d-1} (X^{tv} - \zeta_d^j a^{\tilde{r}v}), \tag{2.14}$$

where $\mathrm{ord}(a^{\tilde{r}v}) = \mathrm{ord}(a)$, because $(a^{\tilde{r}v})^{p\frac{n}{v}} = a$. Equations (2.12), (2.13) and (2.14) combined yield that if $\gcd(np, q-1) = d$ the factorization of $X^{tnp} - a$ into irreducible factors over $\mathbb{F}_q$ is given by:

$$\prod_{v|\frac{np}{d}} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{d-1} (X^{tv} - \zeta_d^j a^{\tilde{r}v}),$$

where for all applicable v and j holds:

(i) $(\zeta_d^j a^{\tilde{r}v})^{\frac{np}{v}} = \zeta_d^{j\frac{np}{v}} a^{\tilde{r}np} = a$, because $v \mid \frac{np}{d}$ implies $d \mid \frac{np}{v}$,

(ii) $\mathrm{ord}(\zeta_d^j a^{\tilde{r}v}) = \frac{d}{\gcd(j,d)} \cdot \mathrm{ord}(a)$, because $\gcd(d, \mathrm{ord}(a)) = 1$.

Case $\gcd(np, q-1) = d \cdot p$: If $\gcd(np, q-1) = dp$, then $d \cdot p \mid (q-1)$ and there exists $\zeta_{dp} \in \mathbb{F}_q$ such that $\zeta_{dp}^p = \zeta_d$. Additionally, we set $\zeta_p := \zeta_{dp}^d$. We consider the polynomial $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ from eq. (2.11), where $v \mid \frac{n}{d}$ and $0 \le j \le d-1$ such that $\gcd(j, v) = 1$. Since $\gcd(d \cdot p, \mathrm{ord}(a)) = 1$ and $dp \mid (q-1)$, the product $\mathrm{ord}(a) \cdot d \cdot p$ divides $(q-1)$. Consequently, also $\mathrm{ord}(\zeta_d^j a^{\tilde{r}vp}) \cdot p$ divides $(q-1)$ and with Theorem 2.32 the factorization of $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{i=0}^{p-1} (X^{tv} - \zeta_p^i \zeta_{dp}^j a^{\tilde{r}v}) = \prod_{i=0}^{p-1} (X^{tv} - \zeta_{dp}^{d\cdot i + j} a^{\tilde{r}v}),$$

because $\zeta_{dp}^j a^{\tilde{r}v}$ is an element of $\mathbb{F}_q$ satisfiying $(\zeta_{dp}^j a^{\tilde{r}v})^p = \zeta_d^j a^{\tilde{r}vp}$.

Note that $\{di + j : 0 \le i \le p-1, 0 \le j \le d-1\} = \{0 \le j \le dp-1\}$. Therefore, the factorization of $X^{tnp} - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{v|\frac{n}{d}} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{d-1} \prod_{i=0}^{p-1} (X^{tv} - \zeta_{dp}^{di+j} a^{\tilde{r}v}) = \prod_{v|\frac{np}{dp}} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{dp-1} (X^{tv} - \zeta_{dp}^j a^{\tilde{r}v}),$$

where $\gcd(j, v) = 1$ if and only if $\gcd(di + j, v) = 1$, because every prime w that divides v must divide n. Since $\mathrm{rad}(n) \mid q-1$, w also divides d. Furthermore, for all applicable v and j, we have

(i) $(\zeta_{dp}^j a^{\tilde{r}v})^{\frac{np}{v}} = (\zeta_{dp}^j a^{\tilde{r}v})^{p\cdot\frac{n}{v}}$, because p does not divide v. Indeed, suppose that $p \mid v$, then with Corollary 2.28 the polynomial $(X^{tvp} - \zeta_d^j a^{\tilde{r}vp})$ would not have been reducible. Thus, $(\zeta_{dp}^j a^{\tilde{r}v})^{\frac{np}{v}} = (\zeta_d^j a^{\tilde{r}vp})^{\frac{n}{v}} = a$ by induction hypothesis.

(ii) $\mathrm{ord}(\zeta_{dp}^j a^{\tilde{r}v}) = \frac{dp}{\gcd(j,dp)} \cdot \mathrm{ord}(a)$, because $\mathrm{ord}(a^{\tilde{r}v}) = \mathrm{ord}(a)$ and $\gcd(dp, \mathrm{ord}(a)) = 1$. □

Example 2.36. Again let $q = 37$ and we consider $a = 6 \in \mathbb{F}_{37}$ with $\mathrm{ord}(a) = 4$. Further, let $n = 81 = 3^4$ and $p_1 = p_2 = p_3 = p_4 = 3$. As in Example 2.34, we factor $X^{81} - 6$ recursively to show the structure of the proof of Theorem 2.35.

(p_1) We begin with $t = 1$ and the binomial $X^3 - 6 \in \mathbb{F}_{37}[X]$. Since $p_1 \cdot \mathrm{ord}(a) = 12$ divides $q - 1 = 36$, the binomial is reducible and the factorization is given by Theorem 2.32 as $\prod_{j=0}^{2}(X - \zeta_3^j b)$, where $b = a^r$ for a positive integer r such that $p_1 \cdot r \equiv 1 \pmod{\mathrm{ord}(a)}$. The integer $r = 3$ satisfies $3 \cdot 3 \equiv 1 \pmod 4$ and therefore $b = a^3 = 31$. Furthermore, a primitive 3rd root of unity was given by $\zeta_3 = 10$ and the factorization is $(X - 31) \cdot (X - 14) \cdot (X - 29)$, where $\mathrm{ord}(31) = \mathrm{ord}(a) = 4$ and $\mathrm{ord}(14) = \mathrm{ord}(29) = p_1 \cdot \mathrm{ord}(a) = 12$.

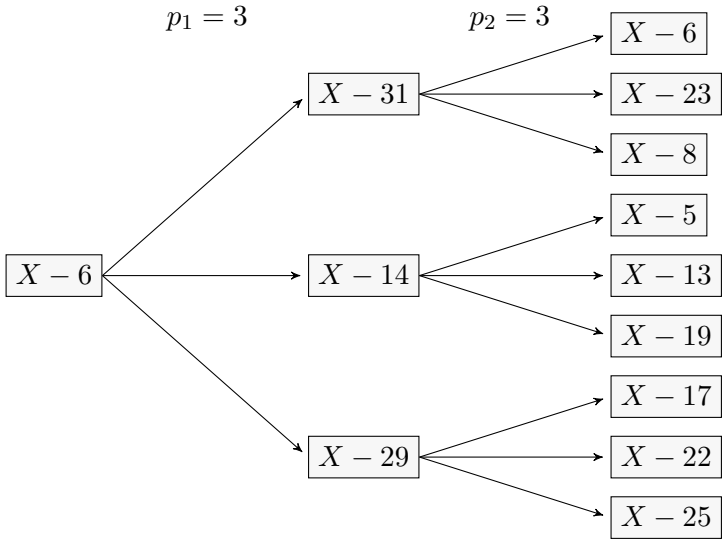
(p_2) Next, we consider the product $(X^3 - 31) \cdot (X^3 - 14) \cdot (X^3 - 29)$ with $t = 1$ and $p_2 = 3$ in all factors. For $X^3 - 31$ holds $p_2 \cdot \mathrm{ord}(31) = 12$ and since 12 divides 36, the factorization is given by Theorem 2.32 as $\prod_{j=0}^{2}(X - \zeta_3^j 31^3)$, because $r = 3$ satisfies $r \cdot p_2 \equiv 1 \pmod 4$. Thus, $b = 31^3 = 6$ and $X^3 - 31 = (X - 6) \cdot (X - 23) \cdot (X - 8)$, where $\mathrm{ord}(6) = \mathrm{ord}(31) = 4$ and $\mathrm{ord}(23) = \mathrm{ord}(8) = 12$.

For $X^3 - 14$ the order $p_2 \cdot \mathrm{ord}(14) = 36$ also divides 36 and the factorization is given by $\prod_{j=0}^{2}(X - 10^j b)$. However, since $p_2 \mid \mathrm{ord}(14)$, the element b has order 36 and needs to be looked up in $\mathbb{F}_{37}$. Indeed, $b = 5$ satisfies $b^3 = 14$. Thus, the factorization is $(X - 5) \cdot (X - 13) \cdot (X - 19)$, where $\mathrm{ord}(5) = \mathrm{ord}(13) = \mathrm{ord}(19) = 36$.

For $X^3 - 29$ we have $p_2 \cdot \mathrm{ord}(29) = 36$, which divides $q - 1$ and $p_2 \mid \mathrm{ord}(29)$. The element $b = 17$ satisfies $b^3 = 29$ and the factorization is given by $(X - 17) \cdot (X - 22) \cdot (X - 35)$, where $\mathrm{ord}(17) = \mathrm{ord}(22) = \mathrm{ord}(35) = 36$. Consequently, the factorization of $X^9 - 6$ over $\mathbb{F}_{37}$ is given by

$$(X-6)\cdot(X-23)\cdot(X-8)\cdot(X-5)\cdot(X-13)\cdot(X-19)\cdot(X-17)\cdot(X-22)\cdot(X-35),$$

where $\mathrm{ord}(6) = 4$, $\mathrm{ord}(23) = \mathrm{ord}(8) = 12$ and $\mathrm{ord}(5) = \mathrm{ord}(13) = \mathrm{ord}(19) = \mathrm{ord}(17) = \mathrm{ord}(22) = \mathrm{ord}(35) = 36$.



(p_3) In this step, we factor the product $(X^3 - 6) \cdot (X^3 - 23) \cdot (X^3 - 8) \cdot (X^3 - 5) \cdot (X^3 - 13) \cdot (X^3 - 19) \cdot (X^3 - 17) \cdot (X^3 - 22) \cdot (X^3 - 35)$. The factorization of $X^3 - 6$ can be taken from (p_1): $(X - 31) \cdot (X - 14) \cdot (X - 29)$, where $\mathrm{ord}(31) = \mathrm{ord}(a) = 4$ and $\mathrm{ord}(14) = \mathrm{ord}(29) = p_1 \cdot \mathrm{ord}(a) = 12$.

For $c \in \{23, 8\}$ holds $p_3 \cdot \mathrm{ord}(c) = 36$, which divides 36. Thus, both binomials $X^3 - c$ factor as $\prod_{j=0}^{2}(X - 10^j \cdot b)$, where $b \in \mathbb{F}_{37}$ satisfies $b^3 = c$. With $18^3 = 23$ and $2^3 = 8$ we obtain that $X^3 - 23 = (X - 18) \cdot (X - 32) \cdot (X - 24)$ and $X^3 - 8 = (X - 2) \cdot (X - 20) \cdot (X - 15)$, where $\mathrm{ord}(b) = 36$ for all $b \in \{18, 32, 24, 2, 20, 15\}$.

For $c \in \{5, 13, 19, 17, 22, 35\}$ the order $p_3 \cdot \mathrm{ord}(c) = 108$ does not divide 36 and the binomial $X^3 - c$ is irreducible with Proposition 2.27.

(p_4) In the last step, we consider the product

$$\begin{aligned}&(X^3 - 31) \cdot (X^3 - 14) \cdot (X^3 - 29) \cdot (X^3 - 18) \cdot (X^3 - 32) \cdot (X^3 - 24)\\ &\cdot (X^3 - 2) \cdot (X^3 - 20) \cdot (X^3 - 15) \cdot (X^{3\cdot 3} - 5) \cdot (X^{3\cdot 3} - 13)\\ &\cdot (X^{3\cdot 3} - 19) \cdot (X^{3\cdot 3} - 17) \cdot (X^{3\cdot 3} - 22) \cdot (X^{3\cdot 3} - 35).\end{aligned}$$

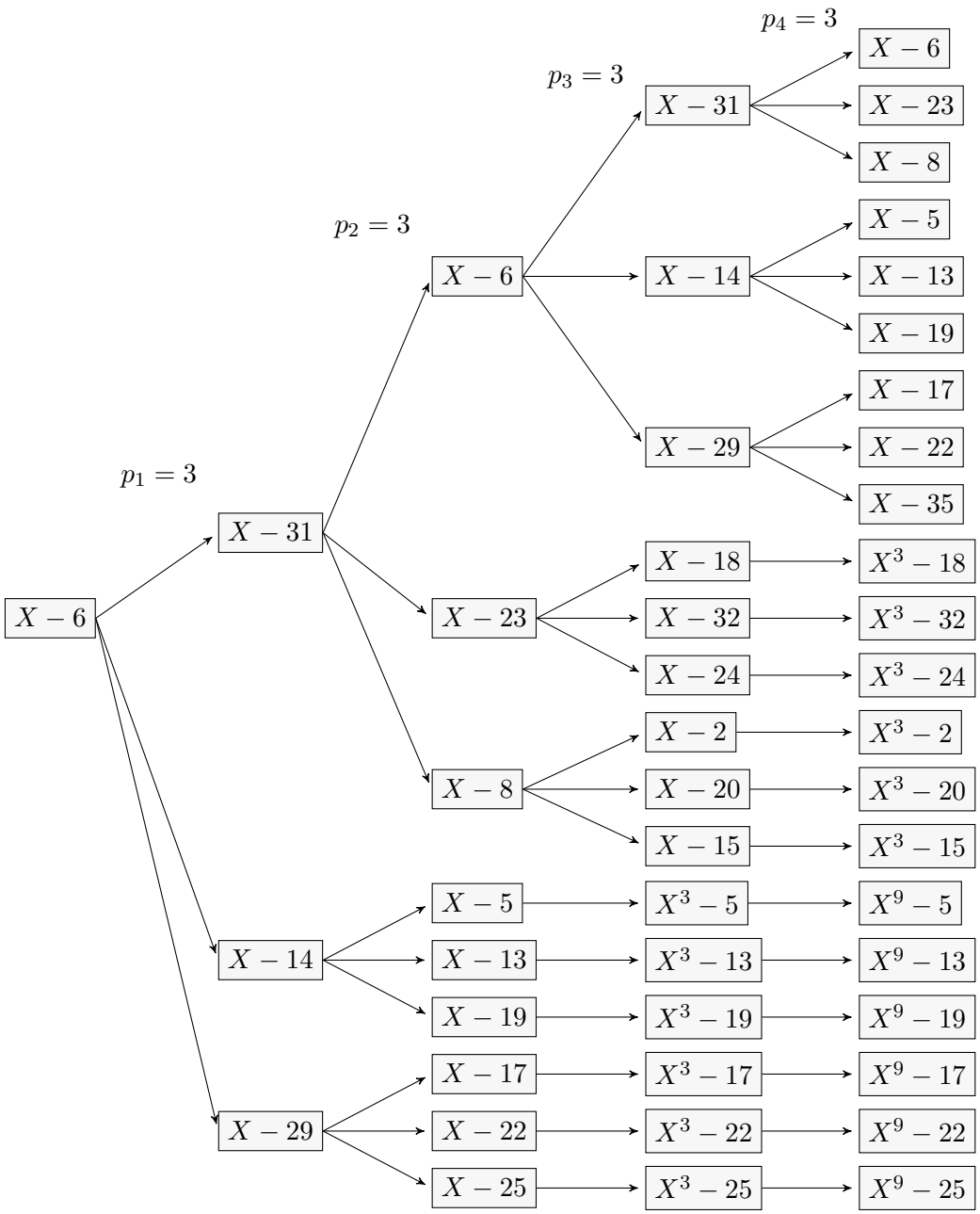
For $c \in \{31, 14, 29\}$ we obtain the factorization from above: $X^3 - 31 = (X - 6) \cdot (X - 23) \cdot (X - 8)$, $X^3 - 14 = (X - 5) \cdot (X - 13) \cdot (X - 19)$ and $X^3 - 29 = (X - 17) \cdot (X - 22) \cdot (X - 35)$.

For $c \in \{18, 32, 24, 2, 20, 15\}$ we have $p_4 \cdot \mathrm{ord}(c) = 108$, which does not divide 36 so that all binomials $X^3 - c$ are irreducible with Proposition 2.27.

For $c \in \{5, 13, 19, 17, 22, 35\}$ holds $t = 3$ and the prime p_4 divides t so that $X^{t \cdot p_4} - c = X^9 - c$ is irreducible with Corollary 2.28. Thus, the factorization of $X^{81} - 6$ over $\mathbb{F}_{37}$ is given by

$$\begin{aligned}&(X - 6) \cdot (X - 23) \cdot (X - 8) \cdot (X - 5) \cdot (X - 13) \cdot (X - 19) \cdot (X - 17)\\ &\cdot (X - 22) \cdot (X - 35)\\ &\cdot (X^3 - 18) \cdot (X^3 - 32) \cdot (X^3 - 24) \cdot (X^3 - 2) \cdot (X^3 - 20) \cdot (X^3 - 15)\\ &\cdot (X^9 - 5) \cdot (X^9 - 13) \cdot (X^9 - 19) \cdot (X^9 - 17) \cdot (X^9 - 22) \cdot (X^9 - 35),\end{aligned}$$

where $\mathrm{ord}(6) = 4$, $\mathrm{ord}(23) = \mathrm{ord}(8) = 12$ and all other coefficients have order 36.



Using Theorem 2.35 instead, we determine $d = \gcd(n, q-1) = 9$ and $\zeta_9 = 7$, which is a primitive d-th root of unity in $\mathbb{F}_{37}$. Furthermore, $r \cdot 81 \equiv 1 \pmod 4$ holds for $r = 1$. Thus, $6^n = 6$ and the factorization of $X^{81} - 6$ over $\mathbb{F}_{37}$ is given by:

$$\prod_{v|9} \prod_{\substack{j=0 \\ \gcd(j,v)=1}}^{8} (X^v - 7^j \cdot 6^v).$$

The formula yields the same irreducible factors as our factorization above and the order of the coefficients $7^j 6^v$ is given by $\mathrm{ord}(a) \cdot \frac{d}{\gcd(d,j)} = 4 \cdot \frac{9}{\gcd(9,j)}$.

The next theorem combines Theorems 2.33 and 2.35. It gives the complete factorization of $X^n - a$ and the order of all appearing coefficients for every positive integer n satisfying $\mathrm{rad}(n) \mid (q-1)$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$). Note that the result covers Theorem 2.12, Theorem 2.19 (1), Theorem 2.20 (1) and Theorem 2.21(1).

Theorem 2.37. *Let $a \in \mathbb{F}_q^*$ and $n \in \mathbb{N}$ such that $\mathrm{rad}(n) \mid (q-1)$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$). We write $n = n_1 \cdot n_2$, where $\mathrm{rad}(n_1) \mid \mathrm{ord}(a)$ and $\gcd(n_2, \mathrm{ord}(a)) = 1$. We select $r \in \mathbb{N}$ such that $r = 1$ if $a = 1$ and $rn_2 \equiv 1 \pmod{\mathrm{ord}(a) \cdot d_1}$ otherwise. Further, we set $d_1 := \gcd(n_1, \frac{q-1}{\mathrm{ord}(a)})$ and $d_2 := \gcd(n_2, q-1)$. Then there exists $b \in \mathbb{F}_q$ such that $b^{d_1} = a$ and the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{j=0}^{d_1-1} \prod_{v|\frac{n_2}{d_2}} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{d_2-1} (X^{\frac{n_1}{d_1} \cdot v} - \zeta_{d_2}^i \zeta_{d_1}^j b^{rv}),$$

and for all applicable j, v, i holds:

(i) $(\zeta_{d_2}^i \zeta_{d_1}^j b^{rv})^{\frac{n_2}{v} \cdot d_1} = a$,

(ii) $\mathrm{ord}(\zeta_{d_2}^i \zeta_{d_1}^j b^{rv}) = \mathrm{ord}(a) \cdot d_1 \cdot \frac{d_2}{\gcd(i,d_2)}$.

Proof. From Theorem 2.33 follows that the factorization of $X^{n_1} - a$ is $\prod_{j=0}^{d_1-1} (X^{\frac{n_1}{d_1}} - \zeta_{d_1}^j b)$, where $b \in \mathbb{F}_q$ such that $b^{d_1} = a$ and $\mathrm{ord}(\zeta_{d_1}^j b) = \mathrm{ord}(a) \cdot d_1$ for all $0 \le j \le d_1 - 1$. Since $\gcd(n_2, \mathrm{ord}(a) \cdot d_1 \cdot \frac{n_1}{d_1}) = 1$, application of Theorem 2.35 yields the factorization of $(X^{\frac{n_1}{d_1}} - \zeta_{d_1}^j b)$:

$$\prod_{v|\frac{n_2}{d_2}} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{d_2-1} (X^{\frac{n_1}{d_1} \cdot v} - \zeta_{d_2}^i (\zeta_{d_1}^j b)^{rv}),$$

where $(\zeta_{d_2}^i (\zeta_{d_1}^j b)^{rv})^{\frac{n_2}{v}} = (\zeta_{d_1}^j b)$ and $\mathrm{ord}(\zeta_{d_2}^i (\zeta_{d_1}^j b)^{rv}) = \mathrm{ord}(\zeta_{d_1}^j b) \cdot \frac{d_2}{\gcd(i,d_2)} = \mathrm{ord}(a) \cdot d_1 \cdot \frac{d_2}{\gcd(i,d_2)}$. The element $\zeta_{d_1}^{rv}$ has order d_1, because $\gcd(rv, d_1) = 1$. Consequently, $\{\zeta_{d_1}^{jrv} : 0 \le j \le d_1 - 1\} = \{\zeta_{d_1}^j : 0 \le j \le d_1 - 1\}$ and instead of $(\zeta_{d_1}^j b)^{rv}$ we can write $\zeta_{d_1}^j b^{rv}$. □

2.3.2 A new closed formula for the factorization of $X^n - a$ for every positive integer n such that $\gcd(n, q) = 1$

With Theorem 2.37 we completed step (0) from our sketch of proof in Section 2.3. We use steps (1) to (5) to prove our main theorem, Theorem 2.41, and therefore state them again:

(1) Let $w = \mathrm{ord}_{\mathrm{rad}(n)}(q)$. If $4 \nmid n$ or $q^w \equiv 1 \pmod 4$, then set $s := w$. Otherwise, set $s := 2w$.

Then Theorem 2.37 yields the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_{q^s}$: $X^n - a = \prod_{R \in \mathcal{R}} R$.

(2) All $R \in \mathcal{R}$ are monic binomials. Thus, $R = X^t - \beta$, for a positive integer t and an element β of $\mathbb{F}_{q^s}$, and $\operatorname{coeffdeg}_q(R)$ equals $\operatorname{ord}_{\operatorname{ord}(\beta)}(q)$. We determine $\operatorname{ord}_{\operatorname{ord}(\beta)}(q)$ explicitly.

(3) We define an equivalence relation $\sim$ on $\mathcal{R}$ as follows: $R \sim R'$ if and only if $R' = R^{(j)}$ for an integer $0 \leq j \leq \operatorname{coeffdeg}_q(R) - 1$.

(4) We determine $\mathcal{R}/\sim$.

(5) With Corollary 2.11 the factorization of $X^n - a$ over $\mathbb{F}_q$ is given by

$$X^n - a = \prod_{R \in \mathcal{R}/\sim} \operatorname{spin}_q[R].$$

In Step (1) we set $s = 2w$ if $4 \mid n$ and $q^w \equiv 3 \pmod 4$, because then $q^{2w} \equiv 1 \pmod 4$ and the factorization of $X^n - a$ over $\mathbb{F}_{q^s}$ is indeed given by Theorem 2.37.

For our proof of Theorem 2.41 it remains to determine $\operatorname{coeffdeg}_q(R)$ for all irreducible factors $R \in \mathcal{R}$ of $X^n - a$ over $\mathbb{F}_{q^s}$ (see Step (2)) and to find a representative system of the equivalence classes of the relation $\sim$ as specified in (3).

From Theorem 2.37 we know that the irreducible factors of $X^n - a$ over $\mathbb{F}_{q^s}$ are binomials defined by three parameters (j, v, i). In the proof of Theorem 2.41 we use the following fact to split the equivalence relation $\sim$ on the irreducible factors of $X^n - a$ in the extension field $\mathbb{F}_{q^s}$ into three separate equivalence relations for j, v and i instead. This reduces the complexity of finding the representative system $\mathcal{R}/\sim$ significantly.

Fact 2.38. *Let $(G, \cdot)$ be an abelian finite group and $a, b \in G$.*

(i) Let $\gcd(\operatorname{ord}(a), \operatorname{ord}(b)) = 1$ and $m_1, m_2, n_1, n_2 \in \mathbb{N}$. Then $a^{m_1} b^{n_1} = a^{m_2} b^{n_2}$ if and only if $a^{m_1} = a^{m_2}$ and $b^{n_1} = b^{n_2}$.

(ii) Let $m \in \mathbb{N}$ such that $\gcd(m, \operatorname{ord}(a) \cdot \operatorname{ord}(b)) = 1$. Then $a^m = b^m$ if and only if $a = b$.

Proof. (i) The equation $a^{m_1} b^{n_1} = a^{m_2} b^{n_2}$ is equivalent to $a^{m_1 - m_2} = b^{n_2 - n_1}$. Since $\operatorname{ord}(a)$ and $\operatorname{ord}(b)$ are coprime, this equation holds if and only if $a^{m_1 - m_2} = 1 = b^{n_2 - n_1}$ and the statement holds.

(ii) If $a = b$, then obviously $a^m = b^m$. If $a^m = b^m$, then $1 = (\frac{a}{b})^m$ and therefore $\operatorname{ord}(\frac{a}{b}) \mid m$. Since $\operatorname{ord}(b^{-1}) = \operatorname{ord}(b)$, we know that $(\frac{a}{b})^{\operatorname{ord}(a) \cdot \operatorname{ord}(b)} = 1$ and thus $\operatorname{ord}(\frac{a}{b})$ divides $\operatorname{ord}(a) \cdot \operatorname{ord}(b)$. Then $\operatorname{ord}(\frac{a}{b}) = 1$, because $\gcd(m, \operatorname{ord}(a) \cdot \operatorname{ord}(b)) = 1$. □

The following fact is well known. We use it in the proof of the next lemma, Lemma 2.40 and in the proof of the closed formula for the factorization of Φ_n, Theorem 2.49.

Fact 2.39. *Let $q \equiv 3 \pmod 4$ and n be a positive integer such that $\operatorname{rad}(n) \mid (q-1)$ and $4 \mid n$.*

(i) Then $\gcd(n, q^2 - 1) = \gcd(n, q - 1) \cdot 2^{1 + \min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}$.

(ii) Let e be a divisor of $q - 1$. Then $\gcd(n, \frac{q^2-1}{e}) = \gcd(n, \frac{q-1}{e}) \cdot 2^{1 + \min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}$

Proof. (i) Since $q \equiv 3 \pmod 4$, we have $2 \mid q - 1$ but $4 \nmid q - 1$. Furthermore, $q^2 - 1 = (q-1) \cdot (q+1)$ and $2 \mid (q+1)$, which implies that $q^2 - 1$ is divisible by 4. Then 4 also divides $\gcd(n, q^2 - 1)$, because $4 \mid n$. Note that $q^2 - 1$ and $q - 1$ do not have any other common prime factors apart from 2. Consequently, $\gcd(n, q^2 - 1) = 2^{\min\{\frac{n}{2}, \nu_2(q+1)\}} \cdot \gcd(n, q - 1)$.

(ii) Since e divides $q-1$, we have $\nu_2(e) \leq 1$. If $2 \nmid e$, then $4 \mid n$ and ((i)) imply that $\nu_2(\gcd(n, \frac{q-1}{e})) = 1$ and $\nu_2(\gcd(n, \frac{q^2-1}{e})) = 2 + \min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}$. If $2 \mid e$, then $\nu_2(\gcd(n, \frac{q-1}{e})) = 0$ and $\nu_2(\gcd(n, \frac{q-1}{e})) = 1 + \min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}$. In both cases, the statement is true, because there are no common prime factors of q^2-1 and $q-1$ other than 2. □

The following lemma specifies the order and the degree of the irreducible factors of $X^n - a$ for every positive integer n such that $\operatorname{rad}(n) \mid \operatorname{ord}(a)$, a special case of $\operatorname{rad}(n) \mid (q-1)$. We use it in the proof of Theorem 2.41 to determine one of the parameters (namely s_1) explicitly. The information about the degree of the irreducible factors could be derived from [WY18, Theorem 8]. However, to present the mechanics behind the case $4 \mid n$ and $q \equiv 3 \pmod 4$, we give an elementary proof of Lemma 2.40 here.

Lemma 2.40. *Let $a \in \mathbb{F}_q^*$ and $n \in \mathbb{N}$ such that $\operatorname{rad}(n) \mid \operatorname{ord}(a)$. Set $d := \gcd(n, \frac{q-1}{\operatorname{ord}(a)})$. Then every root of the polynomial $X^n - a$ has order $\operatorname{ord}(a) \cdot n$ and is a proper element of $\mathbb{F}_{q^k}$, where*

$$k = \begin{cases} \frac{n}{d} & \text{if } 4 \nmid n \text{ or } q \equiv 1 \pmod 4, \\ \frac{n}{d \cdot 2^{\min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}} & \text{otherwise.} \end{cases}$$

Proof. Set $d^{(s)} := \gcd(n, \frac{q^s-1}{\operatorname{ord}(a)})$ for $s \in \{1, 2\}$. It suffices to determine the order and the degree of the irreducible factors of $X^n - a$. We set $s := 1$ if $4 \nmid n$ or $q \equiv 1 \pmod 4$, otherwise we set $s := 2$. Then the factorization of $X^n - a$ over $\mathbb{F}_{q^s}$ is given by Theorem 2.33 for $t = 1$:

$$\prod_{j=0}^{d^{(s)}-1} (X^{t \cdot \frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^j b), \tag{2.15}$$

where $b \in \mathbb{F}_{q^s}$ such that $b^{d^{(s)}} = a$ and $\operatorname{ord}(\zeta_{d^{(s)}}^j b) = \operatorname{ord}(a) \cdot d^{(s)}$ for every $0 \leq j \leq d^{(s)} - 1$. Theorem 2.22 implies that for every $0 \leq j \leq d^{(s)} - 1$ the order of $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^j b$ is $\operatorname{ord}(a) \cdot n$. Consequently, the order of every root of the polynomial $X^n - a$ equals $\operatorname{ord}(a) \cdot n$.

If $4 \nmid n$ or $q \equiv 1 \pmod 4$, then (2.15) is the factorization of $X^n - a$ over $\mathbb{F}_q$ and every irreducible factor has degree $\frac{n}{d^{(1)}}$.

If $4 \mid n$ and $q \equiv 3 \pmod 4$, then (2.15) is the factorization of $X^n - a$ over $\mathbb{F}_{q^2}$ and with Corollary 2.11, the factorization of $X^n - a$ over $\mathbb{F}_q$ is given by $\prod_{j \in \mathcal{J}} \operatorname{spin}_q \left[X^{\frac{n}{d^{(2)}}} - \zeta_{d^{(2)}}^j b \right]$, where $\mathcal{J}$ is a representative system of $\{0, \ldots, d^{(2)} - 1\} / \sim$. The equivalence relation $\sim$ is defined as $j \sim \tilde{j}$ if and only $\zeta_{d^{(2)}}^{\tilde{j}} b = (\zeta_{d^{(2)}}^j b)^{q^m}$ for an integer $m \in \{0, 1\}$. For every $j \in \mathcal{J}$ the order of $\zeta_{d^{(2)}}^j b$ equals $\operatorname{ord}(a) \cdot d^{(2)}$. Since $\operatorname{rad}(n) \mid q-1$, $4 \mid n$ and $q \equiv 3 \pmod 4$, Fact 2.39 ((ii)) implies that $2 \mid \frac{d^{(2)}}{d^{(1)}}$ and $d^{(2)} \cdot \operatorname{ord}(a)$ does not divide $q-1$. Consequently, for all $j \in \mathcal{J}$ holds $\operatorname{coeffdeg}_q \left(X^{\frac{n}{d^{(2)}}} - \zeta_{d^{(2)}}^j b \right) = 2$ and every irreducible factor of $X^n - a$ over $\mathbb{F}_q$ has degree $2 \cdot \frac{n}{d^{(2)}} = 2 \cdot \frac{n}{d^{(1)} \cdot 2^{1 + \min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}} = \frac{n}{d^{(1)} \cdot 2^{\min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}}$. □

The following theorem is our main result in Chapter 2. It gives a closed formula for the factorization of $X^n - a$ into monic irreducible polynomials over $\mathbb{F}_q$ for every $a \in \mathbb{F}_q^*$ and every positive integer n such that $\gcd(n, q) = 1$. Recall that with Remark 1.2 the condition $\gcd(n, q) = 1$ is no restriction on the selection of the positive integer n.

Theorem 2.41. *Let $a \in \mathbb{F}_q^*$ and $n \in \mathbb{N}$ such that $\gcd(n, q) = 1$. We define:*

n_1, n_2	*s.t.* $n = n_1 \cdot n_2$, *where* $\mathrm{rad}(n_1) \mid \mathrm{ord}(a)$ *and* $\gcd(n_2, \mathrm{ord}(a)) = 1$
w	$= \mathrm{ord}_{\mathrm{rad}(n)}(q)$
s	$= \begin{cases} w & \text{if } 4 \nmid n \text{ or } q^w \equiv 1 \pmod 4, \\ 2w & \text{otherwise.} \end{cases}$
${d_1}^{(t)}$	$= \gcd(n_1, \frac{q^t-1}{\mathrm{ord}(a)})$ *for* $t \in \{1, s\}$
${d_2}^{(t)}$	$= \gcd(n_2, q^t - 1)$ *for* $t \in \mathbb{N}$
s_1	$= \begin{cases} \frac{{d_1}^{(s)}}{{d_1}^{(1)}} & \text{if } 4 \nmid {d_1}^{(s)} \text{ or } q \equiv 1 \pmod 4, \\ \frac{{d_1}^{(s)}}{{d_1}^{(1)} \cdot 2^{\min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}} & \text{otherwise.} \end{cases}$
$\sim_b$	*s.t. for* $j, \tilde{j} \in \{0, \ldots, {d_1}^{(s)} - 1\}$ *holds* $j \sim_b \tilde{j}$ *if and only if* $\zeta_{{d_1}^{(s)}}^{\tilde{j}} = \zeta_{{d_1}^{(s)}}^{j \cdot q^u} b^{q^u - 1}$ *for an integer* $0 \le u \le s_1 - 1$, *for* $b \in \mathbb{F}_q$
t_i	$= \min\{t \in \mathbb{N} : \frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i\}$ *for every* $i \in \mathrm{CR}_q({d_2}^{(s)})$
c_i	$= \mathrm{lcm}(t_i, s_1)$ *for every* $i \in \mathrm{CR}_q({d_2}^{(s)})$
r	*s.t.* $\begin{cases} r = 1 & \text{if } a = 1, \\ r n_2 \equiv 1 \pmod{\mathrm{ord}(a) \cdot {d_1}^{(s)}} & \text{otherwise.} \end{cases}$

Then there exists $b \in \mathbb{F}_{q^s}$ such that $b^{{d_1}^{(s)}} = a$ and the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{j \in \{0, \ldots, {d_1}^{(s)} - 1\}/\sim_b} \prod_{v \mid \frac{n_2}{{d_2}^{(s)}}} \prod_{\substack{i \in \mathrm{CR}_q({d_2}^{(s)}) \\ \gcd(i,v)=1}} \prod_{m=0}^{\gcd(t_i, s_1)-1} S_{(j,v,i,m)},$$

where for all applicable (j, v, i, m) the polynomial $S_{(j,v,i,m)}$ is defined as

$$\begin{aligned} S_{(j,v,i,m)} &= \mathrm{spin}_q \left[X^{\frac{n_1}{{d_1}^{(s)}} \cdot v} - \zeta_{{d_2}^{(s)}}^{i \cdot q^m} (\zeta_{{d_1}^{(s)}}^{j} b)^{rv} \right] \\ &= \prod_{u=0}^{c_i - 1} \left(X^{\frac{n_1}{{d_1}^{(s)}} \cdot v} - (\zeta_{{d_2}^{(s)}}^{i \cdot q^m} (\zeta_{{d_1}^{(s)}}^{j} b)^{rv})^{q^u} \right) \\ &= \sum_{l=0}^{c_i} X^{\frac{n_1}{{d_1}^{(s)}} \cdot v \cdot (c_i - l)} \cdot (-1)^l \sum_{\substack{\mathcal{U} \subseteq \{0, \ldots, c_i - 1\} \\ |\mathcal{U}| = l}} \prod_{u \in \mathcal{U}} (\zeta_{{d_2}^{(s)}}^{i \cdot q^m} (\zeta_{{d_1}^{(s)}}^{j} b)^{rv})^{q^u}. \end{aligned}$$

$S_{(j,v,i,m)}$ has degree $\frac{n_1}{{d_1}^{(s)}} \cdot v \cdot c_i$ and order $\mathrm{ord}(a) \cdot n_1 \cdot v \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}$.

Proof. Since $\mathrm{rad}(n) \mid q^s - 1$ and $4 \nmid n$ or $q^s \equiv 1 \pmod 4$, with the proof of Theorem 2.37 the factorization of $X^n - a$ over $\mathbb{F}_{q^s}$ is given by:

$$X^n - a = \prod_{j=0}^{{d_1}^{(s)} - 1} \prod_{v \mid \frac{n_2}{{d_2}^{(s)}}} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{{d_2}^{(s)} - 1} (X^{\frac{n_1}{{d_1}^{(s)}} \cdot v} - \zeta_{{d_2}^{(s)}}^{i} (\zeta_{{d_1}^{(s)}}^{j} b)^{rv}), \tag{2.16}$$

where $b \in \mathbb{F}_{q^s}$ such that ${b^{d_1}}^{(s)} = a$, $r \cdot n_2 \equiv 1 \pmod{\operatorname{ord}(a) \cdot {d_1}^{(s)}}$ and $\operatorname{ord}(\zeta^i_{{d_2}^{(s)}}(\zeta^j_{{d_1}^{(s)}} b)^{rv}) = \operatorname{ord}(a) \cdot {d_1}^{(s)} \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}$ for all applicable (j, v, i). Note that we use the formulation $(\zeta^j_{{d_1}^{(s)}} b)^{rv}$ in (2.16) instead of $\zeta^j_{{d_1}^{(s)}} b^{rv}$ because this allows us to split the equivalence relation $\approx$ (see below) into three separate equivalence relations for j, v and i. We define

$$\mathcal{J} = \{(j, v, i) : j \in \{0, \ldots, {d_1}^{(s)} - 1\}, v \mid \frac{n_2}{{d_2}^{(s)}}, i \in \{0, \ldots, {d_2}^{(s)} - 1\}, \ \gcd(i, v) = 1\}$$

to be the set of all applicable (j, v, i). For all $(j, v, i) \in \mathcal{J}$ we set

$$R_{j,v,i} := (X^{\frac{n_1}{{d_1}^{(s)}} \cdot v} - \zeta^i_{{d_2}^{(s)}}(\zeta^j_{{d_1}^{(s)}} b)^{rv})$$

and $c_i := \operatorname{coeffdeg}_q(R_{j,v,i})$. With Theorem 2.37 holds

$$c_i = \operatorname{ord}_{\operatorname{ord}(\zeta^i_{{d_2}^{(s)}}(\zeta^j_{{d_1}^{(s)}} b)^{rv})}(q) = \operatorname{ord}_{\operatorname{ord}(a) \cdot {d_1}^{(s)} \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}}(q),$$

which only depends on i. Additionally, on $\mathcal{J}$ we define an equivalence relation $\approx$ as follows: $(j, v, i) \approx (\tilde{j}, \tilde{v}, \tilde{i})$ if and only if $R_{(\tilde{j},\tilde{v},\tilde{i})} = {R_{(j,v,i)}}^{(m)}$ for an integer $0 \leq m \leq c_i - 1$. Meaning that $R_{(\tilde{j},\tilde{v},\tilde{i})}$ is a factor of the q-spin of $R_{(j,v,i)}$. Then from Corollary 2.11 follows that the factorization of $X^n - a$ over $\mathbb{F}_q$ is given by: $\prod_{(j,v,i) \in \mathcal{J}/\approx} \operatorname{spin}_q\left[R_{(j,v,i)}\right]$, where $\operatorname{spin}_q\left[R_{(j,v,i)}\right] = \prod_{m=0}^{c_i - 1} {R_{(j,v,i)}}^{(m)} =: S_{(j,v,i)}$. Since $R_{(j,v,i)}$ is a binomial, we can easily compute $S_{(j,v,i)}$:

$$S_{(j,v,i)} = \sum_{l=0}^{c_i} X^{\frac{n_1}{{d_1}^{(s)}} \cdot v \cdot l} \cdot (-1)^{c_i - l} \cdot \sum_{\substack{\mathcal{U} \subseteq \{0, \ldots, c_i - 1\} \\ |\mathcal{U}| = c_i - l}} \prod_{u \in \mathcal{U}} (\zeta^i_{{d_2}^{(s)}}(\zeta^j_{{d_1}^{(s)}} b)^{rv})^{q^u},$$

and from Theorem 2.22 follows that

$$\begin{aligned} \operatorname{ord}(S_{(j,v,i)}) = \operatorname{ord}(R_{(j,v,i)}) &= \frac{n_1}{{d_1}^{(s)}} \cdot v \cdot \operatorname{ord}(\zeta^i_{{d_2}^{(s)}}(\zeta^j_{{d_1}^{(s)}} b)^{rv}) \\ &= \frac{n_1}{{d_1}^{(s)}} \cdot v \cdot \operatorname{ord}(a) \cdot {d_1}^{(s)} \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})} = n_1 \cdot v \cdot \operatorname{ord}(a) \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}. \end{aligned}$$

It remains to determine a representative system $\mathcal{J}/\approx$. For $R_{(j,v,i)}$ and $R_{(\tilde{j},\tilde{v},\tilde{i})}$ to be of the same q-spin, the two polynomials need to be of the same degree, which implies that $v = \tilde{v}$. Furthermore, $\zeta^{\tilde{i}}_{{d_2}^{(s)}}(\zeta^{\tilde{j}}_{{d_1}^{(s)}} b)^{rv} = \zeta^{iq^m}_{{d_2}^{(s)}}(\zeta^j_{{d_1}^{(s)}} b)^{rvq^m}$ for an integer $0 \leq m \leq c_i - 1$. Since $\gcd({d_2}^{(s)}, \operatorname{ord}(a) \cdot {d_1}^{(s)}) = 1$, with Fact 2.38 (i) this is equivalent to

$$\zeta^{\tilde{i}}_{{d_2}^{(s)}} = \zeta^{iq^m}_{{d_2}^{(s)}} \tag{2.17}$$

$$\text{and} \quad (\zeta^{\tilde{j}}_{{d_1}^{(s)}} b)^{rv} = (\zeta^j_{{d_1}^{(s)}} b)^{rvq^m} \tag{2.18}$$

for an integer $0 \leq m \leq c_i - 1$. Condition (2.17) is equivalent to $\tilde{i} = i \cdot q^m \pmod{{d_2}^{(s)}}$. We define $i^{(m)} := i \cdot q^m \pmod{{d_2}^{(s)}}$ for all nonnegative integers m. This is in fact an enumeration of the elements in $\mathrm{C}_{q,{d_2}^{(s)}}(i)$, the q-cyclotomic coset of i modulo ${d_2}^{(s)}$. Since $\zeta^i_{{d_2}^{(s)}}$ is a proper element of $\mathbb{F}_{q^{t_i}}$, $\mathrm{C}_{q,{d_2}^{(s)}}(i)$ contains exactly $t_i := \operatorname{ord}_{\frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}}(q)$ elements. Consequently, $i^{(t_i)} = i^{(0)} = i$ and the elements $i^{(0)}, i^{(1)}, \ldots, i^{(t_i - 1)}$ are all distinct.

Concerning equation (2.18), v divides n_2 and $rn_2 \equiv 1 \pmod{\operatorname{ord}(a) \cdot {d_1}^{(s)}}$, which implies that $\gcd(rv, \operatorname{ord}(a) \cdot {d_1}^{(s)}) = 1$. Then with Fact 2.38 (ii) equation (2.18) is equivalent to $\zeta^{\tilde{j}}_{{d_1}^{(s)}} b = (\zeta^j_{{d_1}^{(s)}} b)^{q^m}$. For any nonnegative integer m we define

$$j^{(m)} := \tilde{j} \quad \Leftrightarrow \quad \zeta^{\tilde{j}}_{{d_1}^{(s)}} b = (\zeta^j_{{d_1}^{(s)}} b)^{q^m}.$$

Note that the elements $\{\zeta_{{d_1}^{(s)}}^j b : 0 \leq j \leq {d_1}^{(s)}\}$ are the distinct roots of the polynomial $X^{{d_1}^{(s)}} - a \in \mathbb{F}_q[X]$ and our definition of $j^{(m)}$ is well-defined. Since $\mathrm{rad}({d_1}^{(s)}) \mid \mathrm{ord}(a)$, we can apply Lemma 2.40 and the order of $\zeta_{{d_1}^{(s)}}^j b$ is $\mathrm{ord}(a) \cdot {d_1}^{(s)}$ for all $0 \leq j \leq {d_1}^{(s)}$. We set $s_1 := \mathrm{ord}_{\mathrm{ord}(a)\cdot {d_1}^{(s)}}(q)$ and every element $\zeta_{{d_1}^{(s)}}^j b$ is a proper element of $\mathbb{F}_{q^{s_1}}$. Consequently, $j^{(s_1)} = j^{(0)} = j$ and the integers $j^{(0)}, j^{(1)}, \ldots, j^{(s_1-1)}$ are all distinct. Additionally, Lemma 2.40 yields that

$$s_1 = \begin{cases} \frac{{d_1}^{(s)}}{{d_1}^{(1)}} & \text{if } 4 \nmid {d_1}^{(s)} \text{ or } q \equiv 1 \pmod 4, \\ \frac{2{d_1}^{(s)}}{{d_1}^{(2)}} & \text{otherwise.} \end{cases}$$

We define an equivalence relation $\sim$ on the set of integers $\{0, \ldots, {d_1}^{(s)}\}$, which follows naturally from our observations above: For $j, \tilde{j} \in \{0, \ldots, {d_1}^{(s)} - 1\}$ holds $j \sim \tilde{j}$ if and only if $\tilde{j} = j^{(m)}$ for an integer $0 \leq m \leq s_1 - 1$.

Using our definitions, $(j, v, i) \approx (\tilde{j}, \tilde{v}, \tilde{i})$ holds if and only if $(\tilde{j}, \tilde{v}, \tilde{i}) = (j^{(m)}, v, i^{(m)})$ for an integer $0 \leq m \leq c_i - 1$. Thus, the equivalence class of every $(j, v, i) \in \mathcal{J}$ is:

$$[(j, v, i)] = \{(j^{(0)}, v, i^{(0)}), (j^{(1)}, v, i^{(1)}), \ldots, (j^{(c_i-1)}, v, i^{(c_i-1)})\}.$$

It is our goal to find a unique representative $(\hat{j}, v, \hat{i})$ for every equivalence class $[(j, v, i)]$ in $\mathcal{J}/\approx$. The fact that $s_1 = \mathrm{ord}_{\mathrm{ord}(a)\cdot {d_1}^{(s)}}(q)$ and $t_i = \mathrm{ord}_{\frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}}(q)$ implies that $c_i = \mathrm{ord}_{\mathrm{ord}(a)\cdot {d_1}^{(s)} \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}} = \mathrm{lcm}(s_2, t_i)$. Therefore the sequence $j^{(0)}, \ldots, j^{(c_i-1)}$ runs exactly $\frac{c_i}{s_1}$ times through the distinct elements $j^{(0)}, \ldots, j^{(s_1-1)}$. By definition, exactly one of these s_1 elements is an element of $\{0, \ldots, {d_1}^{(s)} - 1\}/\sim$. Thus, for the representant $(\hat{j}, v, \hat{i})$ of the equivalence class $[(j, v, i)]$ we can assume that $\hat{j} \in \{0, \ldots, {d_1}^{(s)} - 1\}/\sim$. Without loss of generality, we assume that $\hat{j} = j^{(0)}$.

Now we examine for which $i, \tilde{i} \in \{0, \ldots, {d_2}^{(s)} - 1\}$ the equivalence classes $[(\hat{j}, v, i)]$ and $[(\hat{j}, v, \tilde{i})]$ are equal. Obviously, $\tilde{i}$ needs to be an element of $\mathrm{C}_{q,{d_2}^{(s)}}(i)$. If all $\tilde{i} \in \mathrm{C}_{q,{d_2}^{(s)}}(i)$ satisfy $[(\hat{j}, v, \tilde{i})] = [(\hat{j}, v, i)]$, then we can select $i \in \mathrm{CR}_q({d_2}^{(s)})$ and uniquely describe every equivalence class. However, not all $\tilde{i} \in \mathrm{C}_{q,{d_2}^{(s)}}(i)$ satisfy $[(\hat{j}, v, \tilde{i})] = [(\hat{j}, v, i)]$. In fact, since $j^{(s_1)} = j$ and $j^{(l)} \neq j$ for all $1 \leq l < s_1$, the equivalence classes of $[(j, v, i)]$ and $[(j, v, i^{(m)})]$ are equal if and only if $m = s_1 \cdot l \pmod{t_i}$ for a nonnegative integer l. Since $\{s_1 \cdot l \pmod{t_i} : l \geq 0\} = \{\gcd(s_1, t_i) \cdot l \pmod{t_i} : l \geq 0\}$, this is equivalent to $\tilde{i} = i^{(m)}$ for an integer $m \in \{\gcd(s_1, t_i) \cdot l \pmod{t_i} : l \geq 0\}$. From this fact follows directly, that if $\gcd(s_1, t_i) = 1$ then $[(\hat{j}, v, i)] = [(\hat{j}, v, i^{(m)})]$ for all $0 \leq m \leq t_i - 1$. Otherwise

$$\begin{aligned} &[(\hat{j}, v, i)] = [(\hat{j}, v, i^{(\gcd(s_1,t_i))})] = \ldots = [(\hat{j}, v, i^{(\gcd(s_1,t_i)\cdot(\frac{c_i}{\gcd(s_1,t_i)}-1))})], \\ &[(\hat{j}, v, i^{(1)})] = [(\hat{j}, v, i^{(\gcd(s_1,t_i)+1)})] = \ldots, \\ &\vdots \\ &[(\hat{j}, v, i^{(\gcd(s_1,t_i)-1)})] = [(\hat{j}, v, i^{(2\gcd(s_1,t_i)-1)})] = \ldots. \end{aligned}$$

The equivalence classes $[((\hat{j}, v, i)], [(\hat{j}, v, i^{(1)})], \ldots, [(\hat{j}, v, i^{(\gcd(s_1,t_i)-1)})]$ are all distinct. Thus, for every $i \in \mathrm{CR}_q({d_2}^{(s)})$ such that $\gcd(v, i) = 1$ we put

$$(\hat{j}, v, i), (\hat{j}, v, i^{(1)}), \ldots, (\hat{j}, v, i^{(\gcd(s_1,t_i)-1)})$$

in our representative system $\mathcal{J}/\approx$, which we can describe as the following cartesian product:

$$\{0, \ldots, {d_1}^{(s)} - 1\}/\sim$$

$$\times \{(v, i^{(m)}) : v \mid \frac{n_2}{{d_2}^{(s)}}, i \in \mathrm{CR}_q({d_2}^{(s)}), \gcd(i, v) = 1, 0 \leq m \leq \gcd(s_1, t_i) - 1\}.$$

For any $i \in \{0, \ldots, {d_2}^{(s)} - 1\}$ and any positive integer t holds $t_i = t$ if and only if t is the smallest positive integer such that $\frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}$ divides $q^t - 1$. This is true only if $\frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})} \mid \gcd({d_2}^{(s)}, q^t - 1) = {d_2}^{(t)}$. Since ${d_2}^{(t)}$ is a divisor of ${d_2}^{(s)}$, this is equivalent to $\frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i$. Thus, $t_i = \min\{t \in \mathbb{N} : \frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i\}$. Furthermore, for all $\tilde{i} \in \mathrm{C}_{q, {d_2}^{(s)}}(i)$ holds $\gcd(i, {d_2}^{(s)}) = \gcd(\tilde{i}, {d_2}^{(s)})$. Consequently, $\mathrm{ord}(S_{(j,v,\tilde{i})}) = \mathrm{ord}(S_{(j,v,i)})$ and Theorem 2.41 holds. □

We illustrate the steps of our proof with the following example:

Example 2.42. Let $q = 5$ and $a = 2 \in \mathbb{F}_5$. We select $n = 2^4 \cdot 3 \cdot 7^2 = 2352$ and factor the polynomial $X^{2352} - a$ over $\mathbb{F}_5[X]$. The order of a is $\mathrm{ord}(a) = 2^2 = 4$ and we can write $n = n_1 \cdot n_2$ for $n_1 = 2^4$ and $n_2 = 3 \cdot 7^2$. Since $w = \mathrm{ord}_{\mathrm{rad}(n)}(5) = \mathrm{ord}_{21}(5) = 6$ and $5^6 \equiv 1 \pmod 4$, we can set $s = w = 6$. Note that since w is not prime, the factorization of $X^{2352} - a$ over $\mathbb{F}_5$ was not given by any existing result before.

The factorization of $X^{2352} - 2$ over $\mathbb{F}_{5^6}$ is given by Theorem 2.37. Let $\mathbb{F}_{5^6} = \mathbb{F}_5(\beta)$ where β is a root of the monic irreducible polynomial $X^6 + X^4 + 4X^3 + X^2 + 2 \in \mathbb{F}_5[X]$. The multiplicative group of $\mathbb{F}_{5^6}$ has order $q^s - 1 = 5^6 - 1 = 2^3 \cdot 3^2 \cdot 7 \cdot 31$ and ${d_1}^{(6)} = \gcd(n_1, \frac{5^6-1}{\mathrm{ord}(a)}) = 2$ and ${d_2}^{(6)} = \gcd(n_2, 5^6 - 1) = 3 \cdot 7 = 21$. The element $b = \beta^5 + 3\beta^4 + \beta^2 + 4\beta + 3 \in \mathbb{F}_{5^6}$ satisfies $b^{{d_1}^{(6)}} = b^2 = 2 = a$ and $r \cdot n_2 = r \cdot 3 \cdot 7^2 \equiv 1 \pmod{2 \cdot \mathrm{ord}(a)}$ holds for $r = 3$, because $3 \cdot 7^2 \equiv 3 \pmod 8$. Then $b^r = b^3 = 2\beta^5 + \beta^4 + 2\beta^2 + 3\beta + 1$ satisfies $(b^r)^{n_2 \cdot {d_1}^{(s)}} = (b^r)^{2 \cdot 3 \cdot 7^2} = 2 = a$. Furthermore, a primitive ${d_1}^{(s)}$-th root of unity is $\zeta_2 = -1$ and a primitive ${d_2}^{(s)}$-th root of unity is $\zeta_{21} = 3\beta^5 + 3\beta^3 + 2$. Then the factorization of $X^{2352} - 2$ over $\mathbb{F}_{5^6}$ is

$$\prod_{j=0}^{1} \prod_{v \mid 7} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{20} (X^{2^3 \cdot v} - \zeta_{21}^i (\zeta_2^j b)^{3 \cdot v}) = \prod_{j \in \{0,1\}} \prod_{v \in \{1,7\}} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{20} (X^{2^3 \cdot v} - \zeta_{21}^i (\zeta_2^j b)^{3 \cdot v}),$$

where $\mathrm{ord}(\zeta_{21}^i (\zeta_2^j b)^{3 \cdot v}) = \mathrm{ord}(a) \cdot {d_1}^{(s)} \cdot \frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})} = 4 \cdot 2 \cdot \frac{21}{\gcd(21, i)} \in \{8, 24, 56, 168\}$. We define $\mathcal{J}$ to be the set of all applicable (j, v, i). For all tuples $(j, v, i) \in \mathcal{J}$ we set $R_{(j,v,i)} = X^{2^3 \cdot v} - \zeta_{21}^i (\zeta_2^j b)^{3 \cdot v}$ and $c_i = \mathrm{coeffdeg}_q\left(R_{(j,v,i)}\right) = \mathrm{ord}_{2^3 \cdot \frac{21}{\gcd(21,i)}}(5)$. Then with Corollary 2.11 the factorization of $X^{2352} - 2$ over $\mathbb{F}_5$ into monic irreducible factors is given by

$$\prod_{(j,v,i) \in \mathcal{J}/\approx} \mathrm{spin}_q\left[R_{(j,v,i)}\right],$$

where $(j, v, i) \approx (\tilde{j}, \tilde{v}, \tilde{i})$ if and only if there exists an integer $0 \leq m \leq c_i - 1$ such that $R_{(\tilde{j},\tilde{v},\tilde{i})} = {R_{(j,v,i)}}^{(m)}$. Note that this is true only if $v = \tilde{v}$. Furthermore, $(j, v, i) \approx (\tilde{j}, v, \tilde{i})$ holds if and only if $(\zeta_{21}^i (\zeta_2^j b)^3)^5 = \zeta_{21}^{\tilde{i}} (\zeta_2^{\tilde{j}} b)^3$. Since $\gcd(21, \mathrm{ord}(\zeta_2^j b)) = \gcd(21, 8) = 1$, this is true if and only if $\zeta_{21}^{i5} = \zeta_{21}^{\tilde{i}}$ and $(\zeta_2^j b)^{15} = (\zeta_2^{\tilde{j}} b)^3$ (see Fact 2.38(i)). Thus, $\tilde{i} \in \mathrm{C}_{5,21}(i)$ and $(\zeta_2^j b)^5 = \zeta_2^{\tilde{j}} b$ with Fact 2.38(ii). The elements $\zeta_2^j b$ are roots of the polynomial $X^2 - a$ and $X^2 - a$ is irreducible with Corollary 2.28. Thus, $(\zeta_2^j b)^5 = \zeta_2^{\tilde{j}} b$ if and only if $j \neq \tilde{j}$.

We set $s_1 := \mathrm{ord}_{\mathrm{ord}(a) \cdot {d_1}^{(6)}}(5) = \mathrm{ord}_8(5)$ and Lemma 2.40 yields that $s_1 = \frac{{d_1}^{(6)}}{{d_1}^{(1)}} = \frac{2}{1} = 2$. For every $i \in \{0, \ldots, 20\}$ the coefficient degree c_i is the least common multiple of $t_i := \mathrm{ord}_{\frac{{d_2}^{(s)}}{\gcd(i, {d_2}^{(s)})}}(q)$ and $s_1 = 2$. Furthermore, t_i is a divisor of $s_2 := \mathrm{ord}_{{d_2}^{(s)}}(q) = \mathrm{ord}_{21}(5) = 6$ and satisfies $t_i = \min\{t \in \mathbb{N} : \frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i\}$. Since ${d_2}^{(1)} = 1, {d_2}^{(2)} = 3, {d_2}^{(3)} = 1$ and ${d_2}^{(6)} = 21$,

we obtain that for $i = 0$ holds $t_0 = 1$, for $i \in \{7, 14\}$ holds $t_i = 2$ and for $i \notin \{0, 1, 7, 14\}$ holds $t_i = 6$.

Let $i = 0$, then $v = 1$, because for $v = 7$ holds $\gcd(0, 7) = 7 > 1$. Thus only the elements $(0, 1, 0)$ and $(1, 1, 0)$ in $\mathcal{J}$ satisfy $i = 0$. With our observations above holds $(0, 1, 0) \approx (1, 1, 0)$ and $[(0, 1, 0)]$ yields the irreducible factor $(X^8 - b)(X^8 - (-1)b) = X^{16} + 2$.

Let $i \in \{7, 14\}$, then also $v = 1$, because for $v = 7$ holds $\gcd(i, v) > 1$. There are the elements $(0, 1, 7), (1, 1, 7), (0, 1, 14), (1, 1, 14) \in \mathcal{J}$ and $(0, 1, 7) \approx (1, 1, 14)$ and $(0, 1, 14) \approx (1, 1, 7)$, because $\mathrm{C}_{5,21}(7) = \{7, 14\}$. Then $[(0, 1, 7)]$ yields the irreducible trinomial $X^{16} + X^8 + 2$ and $[(0, 1, 14)]$ the irreducible trinomial $X^{16} + 4X^8 + 2$.

For $i \notin \{0, 7, 14\}$ both $v \in \{1, 7\}$ satisfy $\gcd(i, v) = 1$ and we have $t_i = |\mathrm{C}_{5,21}(i)| = 6$. Since $\gcd(t_i, s_1) = \gcd(6, 2) = 2$, every cyclotomic coset yields 2 distinct equivalence classes for every $v \in \{1, 7\}$. Take for example $i = 1$ and $v = 1$. Then $\mathrm{C}_{5,21}(1) = \{1 \cdot 5^m \pmod{21} : 0 \leq m \leq 6 - 1\} = \{1, 5, 4, 20, 16, 17\}$ and

$$\begin{aligned}[(0, 1, 1)] &= \{(0, 1, 1), (1, 1, 5), (0, 1, 4), (1, 1, 20), (0, 1, 16), (1, 1, 17)\} \\ &= [(0, 1, 4)] = [(0, 1, 16)],\end{aligned}$$

which gives the irreducible factor $X^{48} + 3X^{40} + 2X^{32} + 4X^{24} + 4X^{16} + 2X^8 + 3$. Additionally,

$$\begin{aligned}[(0, 1, 5)] &= \{(0, 1, 5), (1, 1, 4), (0, 1, 20), (1, 1, 16), (0, 1, 17)\} \\ &= [(0, 1, 20)] = [(0, 1, 17)]\end{aligned}$$

gives the irreducible factor $X^{48} + 2X^{40} + 2X^{32} + X^{24} + 4X^{16} + 3X^8 + 3$. Since there exist exactly 18 elements in $\{0 \leq i \leq 20\} \setminus \{0, 7, 14\}$, these integers define $\frac{18}{|\mathrm{C}_{5,21}(i)|} \cdot \gcd(t_i, s_1) = \frac{18}{6} \cdot 2 = 6$ irreducible factors of degree $8v$ of $X^{2352} - 2$ over F_5 for every $v \in \{1, 7\}$. The complete factorization of $X^{2352} - 2$ over $\mathbb{F}_5$ is

$$\begin{aligned}&(X^{16} + 2) \cdot (X^{16} + X^8 + 2) \cdot (X^{16} + 4X^8 + 2) \\ &\cdot (X^{48} + X^{40} + 4X^{32} + 3X^{16} + 4X^8 + 3) \\ &\cdot (X^{48} + 2X^{40} + X^{32} + 4X^{24} + 2X^{16} + 3X^8 + 3) \\ &\cdot (X^{48} + 2X^{40} + 2X^{32} + X^{24} + 4X^{16} + 3X^8 + 3) \\ &\cdot (X^{48} + 3X^{40} + X^{32} + X^{24} + 2X^{16} + 2X^8 + 3) \\ &\cdot (X^{48} + 3X^{40} + 2X^{32} + 4X^{24} + 4X^{16} + 2X^8 + 3) \\ &\cdot (X^{48} + 4X^{40} + 4X^{32} + 3X^{16} + X^8 + 3) \\ &\cdot (X^{336} + X^{280} + 3X^{224} + 2X^{168} + 4X^{112} + 4X^{56} + 2) \\ &\cdot (X^{336} + X^{280} + 4X^{224} + 3X^{168} + 2X^{112} + 4X^{56} + 2) \\ &\cdot (X^{336} + 2X^{280} + X^{224} + 3X^{112} + 3X^{56} + 2) \\ &\cdot (X^{336} + 3X^{280} + X^{224} + 3X^{112} + 2X^{56} + 2) \\ &\cdot (X^{336} + 4X^{280} + 3X^{224} + 3X^{168} + 4X^{112} + X^{56} + 2) \\ &\cdot (X^{336} + 4X^{280} + 4X^{224} + 2X^{168} + 2X^{112} + X^{56} + 2).\end{aligned}$$

The following lemma shows that $\mathbb{F}_{q^s}$ is actually the smallest extension of $\mathbb{F}_q$ such that $\zeta_{{d_1}^{(s)}}$ and $\zeta_{{d_2}^{(s)}}$ are both elements of this extension:

Lemma 2.43. *Let $a \in \mathbb{F}_q^*$ and $n \in \mathbb{N}$ such that $\gcd(n, q) = 1$. We define the positive integers $s, {d_1}^{(s)}$ and ${d_2}^{(s)}$ as in Theorem 2.41. Then*

$$\mathbb{F}_{q^s} = \mathbb{F}_q(\zeta_{{d_1}^{(s)}}, \zeta_{{d_2}^{(s)}}).$$

Proof. Since ${d_1}^{(s)}$ and ${d_2}^{(s)}$ divide $q^s - 1$, the elements $\zeta_{{d_1}^{(s)}}$ and $\zeta_{{d_2}^{(s)}}$ are elements of $\mathbb{F}_{q^s}$. Furthermore, $\mathbb{F}_{q^w}$ is the smallest extension of $\mathbb{F}_q$ such that $\mathrm{rad}(n) \mid q^w - 1$. From the fact

that $\mathrm{rad}({d_1}^{(s)} \cdot {d_2}^{(s)}) = \mathrm{rad}(n)$ follows that the extension $\mathbb{F}_q(\zeta_{{d_1}^{(s)}}, \zeta_{{d_2}^{(s)}})$ equals either $\mathbb{F}_{q^w}$ or $\mathbb{F}_{q^s}$. If $s = w$ this distinction is irrelevant. If $s = 2w$, then $4 \mid n$ and $q^w \equiv 3 \pmod 4$ and Fact 2.39 implies that

$$ {d_1}^{(s)} = {d_1}^{(2w)} = {d_1}^{(w)} \cdot 2^{1+\min\{\nu_2(\frac{{d_1}^{(s)}}{r}, \nu_2(\frac{q^w+1}{s}))\}}. $$

Thus, ${d_1}^{(s)}$ does not divide $q^w - 1$ and $\zeta_{{d_1}^{(s)}}$ is not an element of $\mathbb{F}_{q^w}$. Consequently, $\mathbb{F}_q(\zeta_{{d_1}^{(s)}}, \zeta_{{d_2}^{(s)}} = \mathbb{F}_{q^s}$. □

Remark 2.44. The finite field $\mathbb{F}_{q^s}$ is the smallest extension of $\mathbb{F}_q$ such that $\mathrm{rad}(n) \mid q^s - 1$ and ($4 \nmid n$ or $q^s \equiv 1 \pmod 4$) so that we can apply Theorem 2.37. Lemma 2.43 shows that it can be constructed by adjoining $\zeta_{{d_1}^{(s)}}$ and $\zeta_{{d_2}^{(s)}}$ to $\mathbb{F}_q$.

Not only s but also the parameters s_1, s_2, t_i, c_i in Theorem 2.41 are actually degrees of extension fields over $\mathbb{F}_q$.

From the proof of Theorem 2.41 follows that $s_1 = \mathrm{ord}_{{d_1}^{(s)} \cdot \mathrm{ord}(a)}(q)$. Since the order of $\zeta^j_{{d_1}^{(s)}} \cdot b$ and also of $(\zeta^j_{{d_1}^{(s)}} b)^{rv}$ equals ${d_1}^{(s)} \cdot \mathrm{ord}(a)$ for all applicable j, r, v, the extension $\mathbb{F}_{q^{s_1}}$ equals

$$ \mathbb{F}_{q^{s_1}} = \mathbb{F}_q(\zeta_{{d_1}^{(s)}} b) = \mathbb{F}_q((\zeta^j_{{d_1}^{(s)}} b)^{rv}). $$

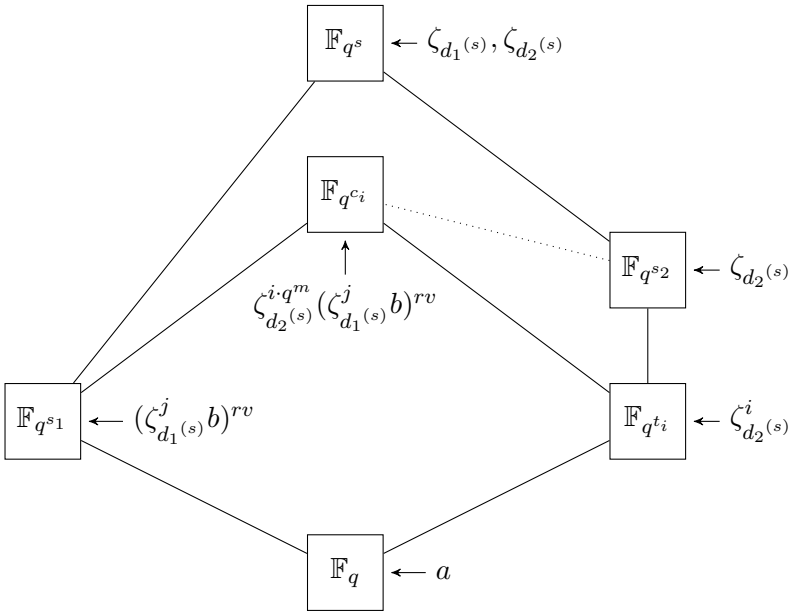
The parameter s_2, which only appears in the proof of Theorem 2.41 describes the finite extension

$$ \mathbb{F}_{q^{s_2}} = \mathbb{F}_q(\zeta_{{d_2}^{(s)}}). $$

Consequently, $\mathbb{F}_{q^s}$ is the smallest extension of $\mathbb{F}_q$ which contains $\mathbb{F}_{q^{s_1}}$ and $\mathbb{F}_{q^{s_2}}$ as subfields. The finite fields $\mathbb{F}_{q^{t_i}}$ for $i \in \mathrm{CR}_q({d_2}^{(s)})$ are the subfields of $\mathbb{F}_{q^{s_2}}$ defined as $\mathbb{F}_{q^{t_i}} = \mathbb{F}_q(\zeta^i_{{d_2}^{(s)}})$. Therefore, $c_i = \mathrm{lcm}(s_1, t_i)$ defines the extension $\mathbb{F}_q(\zeta^i_{{d_2}^{(s)}}, (\zeta^j_{{d_1}^{(s)}} b)^{rv})$ for all applicable j, v, r. For all $0 \le m \le t_i$ we have $\mathrm{ord}(\zeta^i_{{d_2}^{(s)}}) = \mathrm{ord}(\zeta^{i \cdot q^m}_{{d_2}^{(s)}})$ and this order and the order ${d_1}^{(s)} \cdot \mathrm{ord}(a)$ of $(\zeta^j_{{d_1}^{(s)}} b)^{rv}$ are coprime. Thus,

$$ \mathbb{F}_{q^{c_i}} = \mathbb{F}_q(\zeta^i_{{d_2}^{(s)}}, (\zeta^j_{{d_1}^{(s)}} b)^{rv}) = \mathbb{F}_q(\zeta^{i \cdot q^m}_{{d_2}^{(s)}} \cdot (\zeta^j_{{d_1}^{(s)}} b)^{rv}), $$

where $\zeta^{i \cdot q^m}_{{d_2}^{(s)}} \cdot (\zeta^j_{{d_1}^{(s)}} b)^{rv}$ are the elements which define the monic irreducible factors of $X^n - a$ over $\mathbb{F}_q$. The following picture shows the connection between the mentioned extensions of $\mathbb{F}_q$:



2.3.3 The generating polynomials of all constacyclic codes

We obtain a closed explicit formula for the generating polynomials of all constacyclic codes with Theorem 2.41 and Remark 1.2:

Corollary 2.45. *Let n be a positive integer and $\mathcal{C} = \langle g \rangle$ be a λ-constacyclic code $\mathcal{C} = \langle g \rangle$ of length $\tilde{n} = n \cdot \mathrm{char}(\mathbb{F}_q)^l$ for $\lambda \in \mathbb{F}_q^*$ and positive integers n and l such that $\gcd(n, q) = 1$. Furthermore, let $a \in \mathbb{F}_q$ be an element satisfying $a^{\mathrm{char}(\mathbb{F}_q)^l} = \lambda$. We define $n_1, n_2, {d_1}^{(s)}, {d_2}^{(s)}, b, \sim_b, r, s_1, t_i, c_i$ as in Theorem 2.41. Then the generating polynomial g of $\mathcal{C}$ is of the form*

$$g = \prod_{(j,v,i,m)\in\mathcal{I}} \left[\prod_{u=0}^{c_i-1} \left(X^{\frac{n_1}{{d_1}^{(s)}} \cdot v} - (\zeta_{{d_2}^{(s)}}^{i \cdot q^m} (\zeta_{{d_1}^{(s)}}^{j} b)^{rv})^{q^u} \right) \right]^{\mathrm{char}(\mathbb{F}_q)^{l_{(j,v,i,m)}}}$$

$$= \prod_{(j,v,i,m)\in\mathcal{I}} \left[\sum_{l=0}^{c_i} X^{\frac{n_1}{{d_1}^{(s)}} v(c_i - l)} (-1)^l \sum_{\substack{\mathcal{U} \subseteq \{0,\ldots,c_i-1\} \\ |\mathcal{U}| = l}} \prod_{u \in \mathcal{U}} (\zeta_{{d_2}^{(s)}}^{i \cdot q^m} (\zeta_{{d_1}^{(s)}}^{j} b)^{rv})^{q^u} \right]^{\mathrm{char}(\mathbb{F}_q)^{l_{(j,v,i,m)}}}$$

where $l_{(j,v,i,m)} \leq l$ for all $(j, v, i, m) \in \mathcal{I}$ and the index set $\mathcal{I}$ is a subset of

$$\{(j, v, i, m) : \ j \in \{0, \ldots, {d_1}^{(s)} - 1\} / \sim_b, \ v \mid \frac{n_2}{{d_2}^{(s)}}, \ i \in \mathrm{CR}_q({d_2}^{(s)}) \\ \text{such that } \gcd(i, v) = 1, \ 0 \leq m \leq \gcd(t_i, s_1) - 1\}.$$

2.4 A new algorithm for the factorization of $X^n - a$

Our main theorem, Theorem 2.41, is not only a theoretical result, but can be used to compute the factorization of $X^n - a$. The following algorithm is a detailed description of this computation. It is designed so that the number of computations is reduced.

Algorithm 1 (Factorization of $X^n - a$).
Input: Field size q, $n \in \mathbb{N}$ such that $\gcd(n, q) = 1$, $a \in \mathbb{F}_q^*$.
Output: Factorization of $X^n - a$ into monic irreducible polynomials over $\mathbb{F}_q$.

Preliminaries in $\mathbb{Z}$ and $\mathbb{F}_q$:

1.1. Compute $w = \mathrm{ord}_{\mathrm{rad}(n)}(q)$.

1.2. Set $s := \begin{cases} w & \text{if } 4 \nmid n \text{ or } q^w \equiv 1 \pmod 4, \\ 2w & \text{otherwise.} \end{cases}$

1.3. Compute $\mathrm{ord}(a)$.

1.4. Compute $n_1, n_2 \in \mathbb{N}$ s.t. $n = n_1 \cdot n_2$, $\mathrm{rad}(n_1) \mid \mathrm{ord}(a)$ and $\gcd(n_2, \mathrm{ord}(a)) = 1$.

1.5. Compute ${d_1}^{(s)} = \gcd(n_1, \frac{q^s - 1}{\mathrm{ord}(a)})$ and ${d_1}^{(1)} = \gcd(n_1, \frac{q-1}{\mathrm{ord}(a)})$.

1.6. Set $s_1 := \begin{cases} \frac{{d_1}^{(s)}}{{d_1}^{(1)}} & \text{if } 4 \nmid {d_1}^{(s)} \text{ or } q \equiv 1 \pmod 4, \\ \frac{{d_1}^{(s)}}{{d_1}^{(1)} \cdot 2^{\min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}} & \text{otherwise.} \end{cases}$

1.7. Compute ${d_2}^{(s)} = \gcd(n, q^s - 1)$.

1.8. Compute $s_2 = \mathrm{ord}_{{d_2}^{(s)}}(q)$.

1.9. Let $\mathcal{T}$ be the list of all divisors of s_2 (sorted from lowest to highest).
For all $t \in \mathcal{T}$:
  a. Compute ${d_2}^{(t)} = \gcd(n_2, q^t - 1)$.
  b. If ${d_2}^{(t)} = {d_2}^{(\tilde{t})}$ for a $\tilde{t} \in \mathcal{T}$ such that $\tilde{t} < t$, then remove t from $\mathcal{T}$.

1.10. **If** $a = 1$, set $r := 1$.
Else compute $r \in \mathbb{N}$ such that $r \cdot n_2 \equiv 1 \pmod{\mathrm{ord}(a) \cdot {d_1}^{(s)}}$.

Preliminaries in $\mathbb{F}_{q^s}$:

1.11. **If** $\mathbb{F}_{q^s} = \mathbb{F}_q[X]/\langle f\rangle$ for a primitive polynomial $f \in \mathbb{F}_q[X]$, then let γ be a root of f.
Else find a primitive element $\gamma \in \mathbb{F}_{q^s}$.

1.12. Set $\zeta_{{d_1}^{(s)}} := \gamma^{\frac{q^s-1}{{d_1}^{(s)}}}$, $\zeta_{{d_2}^{(s)}} := \gamma^{\frac{q^s-1}{{d_2}^{(s)}}}$ and $\zeta_{{d_1}^{(s)}\cdot \mathrm{ord}(a)} := \gamma^{\frac{q^s-1}{{d_1}^{(s)}\cdot \mathrm{ord}(a)}}$.

1.13. Find $b \in \{\zeta^l_{{d_1}^{(s)}\cdot \mathrm{ord}(a)} : 0 \le l \le {d_1}^{(s)} \cdot \mathrm{ord}(a) - 1, \gcd(l, {d_1}^{(s)} \cdot \mathrm{ord}(a)) = 1\}$ such that $b^{{d_1}^{(s)}} = a$.

1.14. **For** all $j \in \{0, \dots, {d_1}^{(s)} - 1\}$ compute $b_j := (\zeta^j_{{d_1}^{(s)}} \cdot b)^r$.

Representative system $\{0, \dots, {d_1}^{(s)} - 1\}/\sim$:

1.15. Set $R := \{\}$ and $J := \{0, \dots, {d_1}^{(s)} - 1\}$
While $|J| > 0$ do:
a. Select $j \in J$.
b. Set $R \to R \cup \{j\}$.
c. Compute $J_j = \{\tilde{j} \in J : b_{\tilde{j}} = b_j^{q^m}, 0 \le m \le s_1\}$.
d. Set $J \to J \setminus J_j$.
After the while-loop set $\{0, \dots, {d_1}^{(s)} - 1\}/\sim := J$.

Representative system $\mathrm{CR}_q({d_2}^{(s)})$ and t_i, c_i for every $i \in \mathrm{CR}_q({d_2}^{(s)})$:

1.16. Set $C := \{\}$, $I := \{0, \dots, {d_2}^{(s)} - 1\}$
While $|I| > 0$ do:
a. Select $i \in I$.
b. Set $C \to C \cup \{i\}$
c. Compute $t_i = \min\{t \in \mathcal{T} : \frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i\}$.
d. Compute $c_i = \mathrm{lcm}(t_i, s_1)$.
e. Compute $\mathrm{C}_{q,{d_2}^{(s)}}(i) = \{i \cdot q^m \pmod{{d_2}^{(s)}} : 0 \le m \le t_i - 1\}$.
f. Set $I \to I \setminus \mathrm{C}_{q,{d_2}^{(s)}}(i)$.
After the while-loop set $\mathrm{CR}_q({d_2}^{(s)}) := C$.

Computation of the irreducible factors of $X^n - a$:

1.17. **For** all $j \in \{0, \dots, {d_1}^{(s)} - 1\}/\sim$,
for all $v \mid \frac{n_2}{{d_2}^{(s)}}$,
for all $i \in \mathrm{CR}_q({d_2}^{(s)})$ such that $\gcd(i, v) = 1$,
for all $m \in \{0, \dots, \gcd(t_i, s_1) - 1\}$:
Compute

$$\prod_{u=0}^{c_i-1} (X^{\frac{n_1}{{d_1}^{(s)}}\cdot v} - (\zeta^{i\cdot q^m}_{{d_2}^{(s)}} \cdot b_j^v)^{q^u}).$$

Proof of Correctness. We only discuss the correctness of the algorithm steps which differ from the formulation in Theorem 2.41.

1.9. The set $\mathcal{T}$ does not appear in Theorem 2.41. Note that we use it in Step **1.16. c.** for the computation of t_i for every $i \in \mathrm{CR}_q({d_2}^{(s)})$. In Theorem 2.41 the parameter t_i is defined as $\min\{t \mid s_2 : \frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i\}$. If ${d_2}^{(\tilde{t})} = {d_2}^{(t)}$ for two divisors $t, \tilde{t}$ of s_2 such that $\tilde{t} < t$, then $\frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i$ if and only if $\frac{{d_2}^{(s)}}{{d_2}^{(\tilde{t})}} \mid i$. Thus, t is never going to be considered as a candidate for t_i and we can remove it beforehand to reduce the number of computations.

1.12. If $\gamma \in \mathbb{F}_{q^s}$ is a primitive element, then $\gamma^{\frac{q^s-1}{d}}$ has order $\frac{q^s-1}{\gcd(\frac{q^s-1}{d}, q^s-1)} = d$ for every divisor d of $q^s - 1$.

1.13. In Theorem 2.41 the element b is defined as $b \in \mathbb{F}_{q^s}$ such that $b^{{d_1}^{(s)}} = a$. However, if q^s is large, then looping through all elements of $\mathbb{F}_{q^s}$ takes a very long time. Instead we can use the fact that with Theorem 2.33 (which is used in the proof of Theorem 2.37) the order of b equals $\mathrm{ord}(a) \cdot {d_1}^{(s)}$ and restrict our search to the elements in $\mathbb{F}_{q^s}$ of this order.

1.14. In Theorem 2.41 the equivalence relation $\sim$ is defined for the element $\zeta^j_{{d_1}^{(s)}} b$ and not $b_j = (\zeta^j_{{d_1}^{(s)}} b)^r$. However, in the proof of Theorem 2.41 we show that with Fact 2.38(ii) the elements $\zeta^j_{{d_1}^{(s)}} b$ and $\zeta^{\tilde{j}}_{{d_1}^{(s)}} b$ are $\mathbb{F}_q$-conjugates if and only if the elements $(\zeta^j_{{d_1}^{(s)}} b)^r$ and $(\zeta^{\tilde{j}}_{{d_1}^{(s)}} b)^r$ are $\mathbb{F}_q$-conjugates. By computing $b_j := (\zeta^j_{{d_1}^{(s)}} b)^r$ beforehand we omit the repeated computation of this power in the formula of $S_{(j,v,i,m)}$.

1.15. The representative system $\{0, \ldots, {d_1}^{(s)} - 1\}/\sim$ is determined by computing all equivalence classes using the b_js (see **1.14.**) and selecting one representative each (in R).

1.16. The representative system $\mathrm{CR}_q({d_2}^{(s)})$ is determined by computing all cyclotomic cosets modulo ${d_2}^{(s)}$ and selecting one representative each (in C). For every representative we also compute t_i (see **1.9.**) and c_i. □

We illustrate the application of Algorithm 1 with the following example.

Example 2.46. We factor the polynomial $X^n - 4 \in \mathbb{F}_5[X]$ over $\mathbb{F}_5$ for $n = 2^5 \cdot 7^2 \cdot 13 = 20\,384$ with Algorithm 1.

1.1. For any two positive integers k and d such that $\gcd(k, d) = 1$, the order of k in the multiplicative group $(\mathbb{Z}/d\mathbb{Z})^*$ is a divisor of the group order, which is $\varphi(d)$. Thus, $\mathrm{ord}_d(k)$ equals $\min\{m \mid \varphi(d) : d \mid k^m - 1\}$.

Since $\mathrm{rad}(n) = 2 \cdot 7 \cdot 13$, we have $\varphi(\mathrm{rad}(n)) = 1 \cdot 6 \cdot 12 = 72$ and the divisors of $\varphi(\mathrm{rad}(n))$ are $\{1, 2, 3, 4, 6, 8, 9, 12, 18, 24, 36, 72\}$. The smallest divisor w of 72 such that $\mathrm{rad}(n) = 2 \cdot 7 \cdot 13 \mid q^w - 1$ is $w = 12$.

1.2. Since $q = 5 \equiv 1 \pmod 4$, we also have $q^{12} \equiv 1 \pmod 4$ and we set $s := w$.

1.3. Since a is an element of $\mathbb{F}_q$, its multiplicative order e is a divisor of the group order $|\mathbb{F}_q^*| = q - 1$. The smallest divisor e of $q - 1 = 4$ such that $a^e = 1$ is $\mathrm{ord}(a) = 2$.

1.4. Since $\mathrm{ord}(a) = 2$, we have $n_1 = 2^5$ and $n_2 = 7^2 \cdot 13$.

1.5. Since $q^s - 1 = 5^{12} - 1 = 2^4 \cdot 3^2 \cdot 7 \cdot 13 \cdot 31 \cdot 601$, we have

$$\begin{aligned} {d_1}^{(s)} &= \gcd\left(n_1, \frac{2^4 \cdot 3^2 \cdot 7 \cdot 13 \cdot 31 \cdot 601}{2}\right) = \gcd(2^5, 2^3 \cdot 3^2 \cdot 7 \cdot 13 \cdot 31 \cdot 601) \\ &= 2^3 = 8 \\ {d_1}^{(1)} &= \gcd\left(2^5, \frac{4}{2}\right) = 2. \end{aligned}$$

1.6. Since $5 \equiv 1 \pmod 4$, we set $s_1 := \frac{{d_1}^{(s)}}{{d_1}^{(1)}} = \frac{2^3}{2} = 4$.

Recall that $s_1 = \mathrm{ord}_{\mathrm{ord}(a) \cdot {d_1}^{(s)}}(q) = \mathrm{ord}_{2^4}(5)$. *Since* $5^2 - 1 = 2^3 \cdot 3, 5^3 - 1 = 2^2 \cdot 31$ *and* $5^4 - 1 = 2^4 \cdot 3 \cdot 13$ *we have indeed* $s_1 = 4$. *Thus, even though Step* ***1.6.*** *looks complicated, it reduces the computational complexity significantly.*

1.7. Since $q^s - 1 = 5^{12} - 1 = 2^4 \cdot 3^2 \cdot 7 \cdot 13 \cdot 31 \cdot 601$, we have

$${d_2}^{(s)} = \gcd(n_2, q^s - 1) = \gcd(7^2 \cdot 13, 2^4 \cdot 3^2 \cdot 7 \cdot 13 \cdot 31 \cdot 601) = 7 \cdot 13 = 91.$$

1.8. As in **1.1.** we use the fact that $\mathrm{ord}_{{d_2}^{(s)}}(q) = \min\{m \mid \varphi({d_2}^{(s)}) : {d_2}^{(s)} \mid (q^m - 1)\}$. Euler's totient function of ${d_2}^{(s)}$ is $\varphi(7 \cdot 13) = 6 \cdot 12 = 72$ and the smallest divisor m of 72 such that $7 \cdot 13 \mid (q^m - 1)$ is $s_2 = 12$.

1.9. The set of all divisors of $s_2 = 12$ is $\mathcal{T} = \{1, 2, 3, 4, 6, 12\}$. We determine ${d_2}^{(t)}$ for every $t \in \mathcal{T}$:

$$
\begin{aligned}
{d_2}^{(1)} &= \gcd(n_2, q-1) = \gcd(7^2 \cdot 13, 4) = 1, \\
{d_2}^{(2)} &= \gcd(7^2 \cdot 13, 2^3 \cdot 3) = 1, && \mathcal{T} \to \{1, 3, 4, 6, 12\} \\
{d_2}^{(3)} &= \gcd(7^2 \cdot 13, 2^2 \cdot 31) = 1, && \mathcal{T} \to \{1, 4, 6, 12\} \\
{d_2}^{(4)} &= \gcd(7^2 \cdot 13, 2^4 \cdot 3 \cdot 13) = 13, \\
{d_2}^{(6)} &= \gcd(7^2 \cdot 13, 2^3 \cdot 3^2 \cdot 7 \cdot 31) = 7, \\
{d_2}^{(12)} &= 7 \cdot 13 && \text{(see } \mathbf{1.7.}\text{)}.
\end{aligned}
$$

Thus, $t_i \in \mathcal{T} = \{1, 4, 6, 12\}$ for all $i \in \mathrm{CR}_q({d_2}^{(s)})$.

1.10. Since $a \neq 1$, we look for a positive integer r satisfying $r \cdot 7^2 \cdot 13 \equiv 1 \pmod{8 \cdot 2}$. Since $7^2 \cdot 13 \equiv 13 \pmod{16}$, we determine the multiplicative inverse of 13 in $\mathbb{Z}/16\mathbb{Z}$. This is $r = 5$.

1.11. Let $\mathbb{F}_{q^s} = \mathbb{F}_{5^{12}} = \mathbb{F}_5(\beta)$, where β is a root of the monic irreducible polynomial $X^{12} + X^7 + X^6 + 4X^4 + 4X^3 + 3X^2 + 2X + 2 \in \mathbb{F}_5[X]$. The polynomial is a primitive polynomial and β is a primitive element of $\mathbb{F}_{5^{12}}$.

In general, finding a primitive element in $\mathbb{F}_{q^s}$ is no trivial task. However, in SageMath we can construct the finite field $\mathbb{F}_{q^s}$ with a primitive modulus:

```
Fqs=GF(5^(12), modulus="primitive")
```

or select a primitive element with

```
Fqs.multiplicative_generator()
```

1.12. We compute

$$
\begin{aligned}
\zeta_{{d_1}^{(s)}} &= \zeta_8 = \beta^{\frac{5^{12}-1}{8}} \\
&= \beta^{11} + \beta^{10} + 3\beta^9 + 2\beta^7 + \beta^6 + \beta^5 + 2\beta^4 + 2\beta^3 + \beta^2 + \beta \\
\zeta_{{d_2}^{(s)}} &= \zeta_{7\cdot 13} = \beta^{\frac{5^{12}-1}{7\cdot 13}} = \beta^{11} + 3\beta^{10} + 3\beta^9 + \beta^8 + \beta^6 + \beta^5 + 2\beta + 2 \\
\zeta_{\mathrm{ord}(a)\cdot {d_1}^{(s)}} &= \zeta_{2\cdot 8} = \zeta_{16} \\
&= 4\beta^{11} + \beta^{10} + 4\beta^9 + 3\beta^7 + \beta^6 + 4\beta^5 + 4\beta^4 + 4\beta^3 + 3\beta^2 + 3\beta + 2.
\end{aligned}
$$

1.13. For $0 \leq l \leq 15$ such that $\gcd(l, 16) = 1$ we compute $\zeta_{16}^{l \cdot {d_1}^{(s)}} = \zeta_{16}^{l \cdot 8}$ until we find an integer l such that $\zeta_{16}^{8l} = a$. Since $b^{{d_1}^{(s)}}$ has order $\mathrm{ord}(a) = 2$ and there exists only one element of order 2 in $\mathbb{F}_5$, we can select any $0 \leq l \leq 15$ such that $\gcd(l, 16) = 1$. We select $l = 1$ and $b := \zeta_{16}$.

1.14. For $0 \leq j \leq 7 = {d_1}^{(s)} - 1$ we compute $b_j = (\zeta_{{d_1}^{(s)}}^j b)^r$. We omit the detailed results here but they can be found in Example A.1.

1.15. Every equivalence class in $\{0, \ldots, 7\}$ with respect to $\sim$ contains exactly $s_1 = 4$ elements. Thus, there exist two equivalence classes. For $j = 0$ we compute

$$b_0^{q^0} = b_0,\ b_0^{q^1} = b_2,\ b_0^{q^2} = b_4,\ b_0^{q^3} = b_6,$$

which implies $[0] = \{0, 2, 4, 6\}$. For $j = 1$ holds

$$b_1^{q^0} = b_1,\ b_1^{q^1} = b_7,\ b_1^{q^2} = b_5,\ b_1^{q^3} = b_3$$

and $[1] = \{1, 3, 5, 7\}$. We set $\{0, \ldots, 7\}/\sim = \{0, 1\}$.

1.16. We determine $\mathrm{CR}_5(7 \cdot 13) = \mathrm{CR}_5(91)$ and for every $i \in \mathrm{CR}_5(91)$ we determine $t_i = \min\{t \in \mathcal{T} : \frac{7\cdot 13}{{d_2}^{(t)}} \mid i\}$ and $c_i = \mathrm{lcm}(t_i, s_1)$:

i	t_i	c_i	$\mathrm{C}_{5,91}(i)$
0	1	4	$\{0\}$
1	12	12	$\{1, 5, 25, 34, 79, 31, 64, 47, 53, 83, 51, 73\}$
2	12	12	$\{2, 10, 50, 68, 67, 62, 37, 3, 15, 75, 11, 55\}$
4	12	12	$\{4, 20, 9, 45, 43, 33, 74, 6, 30, 59, 22, 19\}$
7	4	4	$\{7, 35, 84, 56\}$
8	12	12	$\{8, 40, 18, 90, 86, 66, 57, 12, 60, 27, 44, 38\}$
13	6	12	$\{13, 65, 52, 78, 26, 39\}$
14	4	4	$\{14, 70, 77, 21\}$
16	12	12	$\{16, 80, 36, 89, 81, 41, 23, 24, 29, 54, 88, 76\}$
17	12	12	$\{17, 85, 61, 32, 69, 72, 87, 71, 82, 46, 48, 58\}$
28	4	4	$\{28, 49, 63, 42\}$

Thus $\mathrm{CR}_5(91) = \{0, 1, 2, 4, 7, 8, 13, 14, 16, 17, 28\}$.

1.17. The divisors of $\frac{n_2}{{d_2}^{(s)}} = \frac{7^2\cdot 13}{7\cdot 13} = 7$ are 1 and 7 and $\frac{n_1}{{d_1}^{(s)}} = \frac{2^5}{2^3} = 4$. Thus, for all $j \in \{0, 1\}$, $v \in \{1, 7\}$, $i \in \{0, 1, 2, 4, 7, 8, 13, 14, 16, 17, 28\}$ such that $\gcd(i, v) = 1$ and all $m \in \{0, \ldots, \gcd(t_i, 4) - 1\}$ we compute

$$\prod_{u=0}^{c_i-1} (X^{4\cdot v} - (\zeta_{91}^{i\cdot q^m} \cdot b_j^v)^{q^u}).$$

Note that the computation of the explicit formula given in Theorem 2.41

$$\sum_{l=0}^{c_i} X^{4\cdot v\cdot l} \cdot (-1)^{c_i - l} \sum_{\substack{\mathcal{U}\subseteq\{0,\ldots,c_i-1\} \\ |\mathcal{U}|=c_i-l}} \prod_{u\in\mathcal{U}} (\zeta_{91}^{i\cdot q^m} \cdot b_j^v)^{q^u}$$

would take much longer because it is costly to determine all $c_i - l$ subsets of $\{0, \ldots, c_i - 1\}$ for all $0 \le l \le c_i$ and for all elements of $\{c_i : i \in \mathrm{CR}_5(91)\} = \{4, 12\}$.

For $i = 0$ we have $\gcd(t_0, 4) = 1$ and $\gcd(0, 7) = 7 > 1$. Thus, for $i = 0$ we consider the tuples $(j, v, i, m) = (0, 1, 0, 0)$ and $(1, 1, 0, 0)$ which both yield one monic irreducible factor of degree $4 \cdot c_i = 16$. For $i \in \{7, 14, 28\}$ we have $\gcd(t_i, 4) = 4$ and $\gcd(i, 7) = 7 > 1$ and we need to consider the tuples $(0, 1, i, 0), (0, 1, i, 1), (0, 1, i, 2)$, $(0, 1, i, 3), (1, 1, i, 0), (1, 1, i, 1), (1, 1, i, 2), (1, 1, i, 3)$ which all yield one factor of degree $4 \cdot 4 = 16$. In total $i \in \{7, 14, 28\}$ yield exactly 24 factors of degree 16. For $i \in \{1, 2, 4, 8, 16, 17\}$ we consider all tuples (j, v, i, m) with $j \in \{0, 1\}, v \in \{1, 7\}, m \in \{0, 1, 2, 3\}$. For $v = 1$ each of these i yields exactly $2 \cdot 4 = 8$ irreducible factors of degree $4 \cdot 12 = 48$ and for $v = 7$ exactly 8 irreducible factors of degree $4 \cdot 7 \cdot 12 = 336$. For $i = 13$ we consider all tuples $(j, v, 13, m)$ with $j \in \{0, 1\}, v \in \{1, 7\}, m \in \{0, 1\}$ because $\gcd(t_{13}, 4) = \gcd(6, 4) = 2$. These give $2 \cdot 2 = 4$ factors of degree 48 and 4 factors of degree 336.

In total, the factorization of $X^{20\,384} - 4$ consists of 26 factors of degree 16, 52 factors of degree 48 and 52 factors of degree 336. For the complete explicit factorization see Example A.1.

From Example 2.46 we can see that the preparation steps **1.1.** to **1.14.** are simple computations. A little more work is necessary to determine the representative systems $\{0, \ldots, {d_1}^{(s)} - 1\}/\sim$ and $\mathrm{CR}_q({d_2}^{(s)})$ and the computation of the irreducible factors themselves is heavy work for the computer.

In Section 2.9 we present an implementation of Algorithm 1 and compare its performance with the existing functions `factor` from SageMath and `Factorization` from Magma.

2.5 New results on the factorization of $X^n - 1$ and Φ_n

Recall that there exist explicit formulas for the factorization of $X^n - 1$ over $\mathbb{F}_q$ for all positive integers n satisfying $\mathrm{rad}(n) \mid (q^w - 1)$ where $w = 1$ [MGO15, Corollaries 1 and 2], w is prime [WYF18, Theorems 3.2, 3.4 and 3.6] or w is the product of two primes [WY21, Theorems 3.3, 3.5, 3.8, 4.2, 4.4 and 4.7]. No closed formula for arbitrary positive integers n was known.

With the choice $a = 1$ we derive the following closed formula for the factorization of $X^n - 1$ for every positive integer n such that $\gcd(n, q) = 1$ from Theorem 2.41. Recall that with Remark 1.2 the condition $\gcd(n, q) = 1$ is no restriction on the selection of the positive integer n. Theorem 2.47 covers all factorizations given in [MGO15; WYF18] and [WY21].

Theorem 2.47. *Let $n \in \mathbb{N}$ such that $\gcd(n, q) = 1$. Let $w := \mathrm{ord}_{\mathrm{rad}(n)}(q)$ and set $s := w$ if $4 \nmid n$ or $q^w \equiv 1 \pmod 4$, else set $s := 2w$. Furthermore, for all positive integers t we define $d^{(t)} := \gcd(n, q^t - 1)$ and for every $i \in \mathrm{CR}_q(d^{(s)})$ we set $c_i := \min\{t \in \mathbb{N} : \frac{d^{(s)}}{d^{(t)}} \mid i\}$.*
Then the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{v \mid \frac{n}{d^{(s)}}} \prod_{\substack{i \in \mathrm{CR}_q(d^{(s)}) \\ \gcd(i,v)=1}} \left[\prod_{u=0}^{c_i - 1} \left(X^v - \zeta_{d^{(s)}}^{i q^u} \right) \right]$$

$$= \prod_{v \mid \frac{n}{d^{(s)}}} \prod_{\substack{i \in \mathrm{CR}_q(d^{(s)}) \\ \gcd(i,v)=1}} \left[\sum_{l=0}^{c_i} X^{v \cdot l} \cdot (-1)^{c_i - l} \sum_{\substack{\mathcal{U} \subseteq \{0, \ldots, c_i - 1\} \\ |\mathcal{U}| = c_i - l}} \prod_{u \in \mathcal{U}} \zeta_{d^{(s)}}^{i \cdot q^u} \right],$$

where for all $v \mid \frac{n}{d^{(s)}}$ and all $i \in \mathrm{CR}_q(d^{(s)})$ such that $\gcd(i, v) = 1$, the monic irreducible factor belonging to (v, i) has degree $v c_i$ and order $v \cdot \frac{d^{(s)}}{\gcd(i, d^{(s)})}$.

Proof. We apply Theorem 2.41 for $a = 1$. Then $\mathrm{ord}(a) = 1$ implies that $1 = n_1 = {d_1}^{(t)}$ for all positive integers t and $n = n_2$. Furthermore, $\zeta_{{d_1}^{(s)}} = 1$ and the element $b \in \mathbb{F}_q$ such that $b^{{d_1}^{(s)}} = a$ can be set to 1. Its order is obviously 1, which implies that $s_1 = 1$. The representative system $\{0, \ldots, {d_1}^{(s)} - 1\} / \sim = \{0\} / \sim$ is $\{0\}$ and we can set $j = 0$ for all irreducible factors. Furthermore, for all $i \in \mathrm{CR}_q({d_2}^{(s)})$ holds $\gcd(t_i, s_1) = 1$. Since $a = 1$ the positive integer r satisfies $r = 1$. Set $d^{(s)} := {d_2}^{(s)}$. Then the factorization of $X^n - 1$ over $\mathbb{F}_q$ is given by

$$\prod_{v \mid \frac{n}{d^{(s)}}} \prod_{\substack{i \in \mathrm{CR}_q(d^{(s)}) \\ \gcd(i,v)=1}} S_{(0,v,i,0)} = \prod_{v \mid \frac{n}{d^{(s)}}} \prod_{\substack{i \in \mathrm{CR}_q(d^{(s)}) \\ \gcd(i,v)=1}} = \sum_{l=0}^{c_i} X^{vl} \cdot (-1)^{c_i - l} \sum_{\substack{\mathcal{U} \subseteq \{0, \ldots, c_i - 1\} \\ |\mathcal{U}| = c_i - l}} \prod_{u \in \mathcal{U}} (\zeta_{d^{(s)}}^i)^{q^u}.$$

$S_{(0,v,i,0)}$ has degree $v \cdot c_i$ and order $v \cdot \frac{d^{(s)}}{\gcd(i, d^{(s)})}$. □

2.5.1 The generating polynomials of all cyclic codes

With Theorem 2.47 and Remark 1.2 we obtain a closed explicit formula for the generating polynomials of all cyclic codes.

Corollary 2.48. *Let n be a positive integer and $\mathcal{C} = \langle g \rangle$ be a cyclic code of length $\tilde{n} = n \cdot \mathrm{char}(\mathbb{F}_q)^l$, for positive integers n and l such that $\gcd(n, q) = 1$. We define $s, d^{(s)}, c_i$ as in Theorem 2.47. Then the generating polynomial g of $\mathcal{C}$ is of the form*

$$\begin{aligned} g &= \prod_{(v,i)\in\mathcal{I}} \left[\prod_{u=0}^{c_i-1} \left(X^v - \zeta_{d^{(s)}}^{iq^u} \right) \right]^{\mathrm{char}(\mathbb{F}_q)^{l_{(j,v,i,m)}}} \\ &= \prod_{(v,i)\in\mathcal{I}} \left[\sum_{l=0}^{c_i} X^{v\cdot l} \cdot (-1)^{c_i - l} \sum_{\substack{\mathcal{U}\subseteq\{0,\ldots,c_i-1\} \\ |\mathcal{U}|=c_i-l}} \prod_{u\in\mathcal{U}} \zeta_{d^{(s)}}^{i\cdot q^u} \right]^{\mathrm{char}(\mathbb{F}_q)^{l_{(j,v,i,m)}}} \end{aligned}$$

where $l_{(j,v,i,m)} \leq l$ for all $(j, v, i, m) \in \mathcal{I}$ and the index set $\mathcal{I}$ is a subset of

$$\{(v, i) : v \mid \frac{n}{d^{(s)}}, i \in \mathrm{CR}_q({d_2}^{(s)}) \text{ such that } \gcd(i, v) = 1\}.$$

2.5.2 A new closed formula for the explicit factorization of the n-th cyclotomic polynomial Φ_n

Recall that for any positive integer n such that $\gcd(n, q) = 1$ the n-th cyclotomic polynomial Φ_n is the product of all primitive n-th roots of unity and that $X^n - 1$ is the product $\prod_{d|n} \Phi_d$. Consequently, the explicit factorization of the n-th cyclotomic polynomial Φ_n is given by the product of all monic irreducible factors of order n in the factorization of $X^n - 1$ from Theorem 2.47.

We present the resulting new closed formula for the factorization of Φ_n for every positive integer such that $\gcd(n, q) = 1$ in the next theorem, Theorem 2.49. It covers all explicit factorizations given in [FY07; WW12; TW13; Wu+17; Als18]. Even though Theorem 2.49 is a corollary of Theorem 2.47, we give a direct proof for it here.

Theorem 2.49. *Let $n \in \mathbb{N}$ such that $\gcd(n, q) = 1$. Set $w := \mathrm{ord}_{\mathrm{rad}(n)}(q)$ and set $s := w$ if $4 \nmid n$ or $q^w \equiv 1 \pmod 4$, otherwise set $s := 2w$. Furthermore, we define $d^{(s)} := \gcd(n, q^s - 1)$.*

Then the factorization of the cyclotomic polynomial Φ_n into $\frac{\varphi(d^{(s)})}{s}$ monic irreducible factors of degree $\frac{n}{d^{(s)}} \cdot s$ over $\mathbb{F}_q$ is

$$\begin{aligned} &\prod_{\substack{i\in\mathrm{CR}_q(d^{(s)}) \\ \gcd(i,d^{(s)})=1}} \left[\prod_{u=0}^{s-1} \left(X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^{i\cdot q^u} \right) \right] \\ &= \prod_{\substack{i\in\mathrm{CR}_q(d^{(s)}) \\ \gcd(i,d^{(s)})=1}} \left[\sum_{l=0}^{s} X^{\frac{n}{d^{(s)}}\cdot l} \cdot (-1)^{s-l} \sum_{\substack{\mathcal{U}\subseteq\{0,\ldots,s-1\} \\ |\mathcal{U}|=s-l}} \prod_{u\in\mathcal{U}} \zeta_{d^{(s)}}^{i\cdot q^u} \right]. \end{aligned}$$

Proof. Let $\mathcal{I} := \{0 \leq i \leq d^{(s)} - 1 : \gcd(i, d^{(s)}) = 1\}$. Then every primitive n-th root of unity ζ_n is a root of a binomial of the form $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i \in \mathbb{F}_{q^s}[X]$ for a primitive $d^{(s)}$-th

root of unity $\zeta_{d^{(s)}}^i$ with $i \in \mathcal{I}$. Note that for two distinct elements i and $\tilde{i}$ of $\mathcal{I}$, there do not exist any common roots of $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i$ and $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^{\tilde{i}}$. Furthermore, there exist exactly $\varphi(d^{(s)})$ polynomials of this form.

Let $i \in \mathcal{I}$. We show that the binomial $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i$ is irreducible over $\mathbb{F}_{q^s}$. Every prime factor of $\frac{n}{d^{(s)}}$ is also a prime factor of $d^{(s)}$ because $\mathrm{rad}(n) \mid q^s - 1$ and $d^{(s)} = \gcd(n, q^s - 1)$. Since $\mathrm{ord}(\zeta_{d^{(s)}}^i) = d^{(s)}$, holds $\mathrm{rad}(\frac{n}{d^{(s)}}) \mid \mathrm{ord}(\zeta_{d^{(s)}}^i)$ and $\gcd(\frac{n}{d^{(s)}}, \frac{q^s-1}{\mathrm{ord}(\zeta_{d^{(s)}}^i)}) = 1$. The positive integer s was selected so that either $4 \nmid n$ or $q^s \equiv 1 \pmod 4$. With Theorem 2.22 follows that the polynomial $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i$ is irreducible over $\mathbb{F}_{q^s}$. Consequently, $\mathrm{ord}_n(q) = \frac{n}{d^{(s)}} \cdot s$. The roots of $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i$ are $\frac{n}{d^{(s)}}$ distinct primitive n-th roots of unity. Since there exist exactly $\varphi(d^{(s)})$ irreducible binomials of this form, we obtain that $\varphi(n) = \frac{n}{d^{(s)}} \cdot \varphi(d^{(s)})$. The factorization of Φ_n into monic irreducible factors over $\mathbb{F}_{q^s}$ is :

$$\prod_{\substack{i=0 \\ \gcd(i,d^{(s)})=1}}^{d^{(s)}-1} (X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i).$$

With Corollary 2.11 the factorization of Φ_n over $\mathbb{F}_q$ is given by

$$\prod_{\{0 \le i \le d^{(s)}-1:\ \gcd(i,d^{(s)})=1\}/\sim} \mathrm{spin}_q \left[X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i \right],$$

where $i \sim \tilde{i}$ if and only if $\tilde{i} \in \mathrm{C}_{q,d^{(s)}}(i)$. It remains to prove that for every $i \in \mathcal{I}$, the primitive $d^{(s)}$-th root of unity $\zeta_{d^{(s)}}^i$ is a proper element of $\mathbb{F}_{q^s}$. The smallest integer t such that $d^{(s)} \mid q^t - 1$ satisfies $t = \mathrm{ord}_{d^{(s)}}(q) \ge \mathrm{ord}_{\mathrm{rad}(d^{(s)})}(q) = \mathrm{ord}_{\mathrm{rad}(n)}(q) = w$ and from the fact that $d^{(s)}$ divides $q^s - 1$ follows that $t = \mathrm{ord}_{d^{(s)}} \le s$. If $s = w$, then $t = \mathrm{ord}_{d^{(s)}}(q) = s$. If $s = 2w$, then $4 \mid n$ and $q^w \equiv 3 \pmod 4$. Thus, with Fact 2.39 ((i)) holds $\gcd(n, q^s - 1) = \gcd(n, q^w - 1) \cdot 2^{1+\min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q+1}{2})\}}$. Consequently, $2 \mid \frac{d^{(s)}}{d^{(w)}}$ which implies that $d^{(s)} \nmid q^w - 1$ and $s = \mathrm{ord}_{d^{(s)}}(q)$. Thus, $\mathrm{coeffdeg}_q \left(X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i \right) = s$. Since $X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i$ is a binomial, its q-spin can easily be computed:

$$\mathrm{spin}_q \left[X^{\frac{n}{d^{(s)}}} - \zeta_{d^{(s)}}^i \right] = \sum_{l=0}^{s} X^{\frac{n}{d^{(s)}} \cdot l} \cdot (-1)^{c_i - l} \sum_{\substack{\mathcal{U} \subseteq \{0,\ldots,c_i-1\} \\ |\mathcal{U}| = c_i - l}} \prod_{u \in \mathcal{U}} \zeta_{d^{(s)}}^{i \cdot q^u}.$$

□

Remark 2.50. In Section 2.1 we explained that the degree of the irreducible factors of Φ_n is $\mathrm{ord}_n(q)$ and the number of irreducible factors is $\frac{\varphi(n)}{\mathrm{ord}_n(q)}$. In Theorem 2.49 the degree of the irreducible factors is given as $\frac{n}{d^{(s)}} \cdot s$ and the number of irreducible factors as $\frac{\varphi(d^{(s)})}{s}$. These two statements are equivalent. Indeed, from the proof of Theorem 2.49 follows that $\mathrm{ord}_n(q)$ equals $\frac{n}{d^{(s)}} \cdot s$. Furthermore, since $d^{(s)} \mid n$ and $\mathrm{rad}(n) = \mathrm{rad}(d^{(s)})$, for every i in

$$\begin{aligned}\{0, \ldots, n-1\} = \{&0, \ldots, d^{(s)} - 1, d^{(s)} + 0, \ldots, d^{(s)} + (d^{(s)} - 1), \\ &\ldots, (\frac{n}{d^{(s)}} - 1) \cdot d^{(s)} + 0, \ldots, (\frac{n}{d^{(s)}} - 1) \cdot d^{(s)} + (d^{(s)} - 1)\}\end{aligned}$$

holds $\gcd(i, n) = 1$ if and only if $\gcd(i, d^{(s)}) = 1$. Consequently, $\varphi(n) = \frac{n}{d^{(s)}} \cdot \varphi(d^{(s)})$ and $\frac{\varphi(n)}{\mathrm{ord}_n(q)} = \frac{\frac{n}{d^{(s)}} \cdot \varphi(d^{(s)})}{\frac{n}{d^{(s)}} \cdot s} = \frac{\varphi(d^{(s)})}{s}$.

2.6 Known results on the factorization of $f(X^n)$

In this section we consider again the factorization of the composition $f(X^n)$ for a positive integer n and a monic irreducible polynomial $f \in \mathbb{F}_q[X]$, $f \neq X$. Recall that with Remark 2.1 the polynomial $f = X$ is of no interest.

Let f be of degree k and $\alpha \in \mathbb{F}_{q^k}^*$ be a root of f. The explicit factorization of $f(X^n)$ was known before for all positive integers n such that $\gcd(n, \operatorname{ord}(f) \cdot \deg(f)) = 1$ and $\operatorname{rad}(n) \mid (q^w - 1)$ where $w = 1$ or w is an odd prime satisfying $\gcd(n, w) = 1$. It was given in [MRS19] in the following three separate theorems, Theorem 2.51, Theorem 2.55 and Theorem 2.57.

Theorem 2.51 gives the explicit factorization of $f(X^n)$ over $\mathbb{F}_q$ for all positive integers n such that $\operatorname{rad}(n) \mid (q-1)$ and ($q \equiv 1 \pmod 4$ or $8 \nmid n$). As in Section 2.1 and Section 2.2 we simplified the statement of Theorems 2.51, 2.55 and 2.57 by introducing the definition $d^{(t)} := \gcd(n, q^t - 1)$ for positive integers t.

Theorem 2.51 ([MRS19, Theorem 2.3])**.** *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and let $\alpha \in \mathbb{F}_{q^k}$ be a root of f. Furthermore, let $n \in \mathbb{N}$ such that $\operatorname{rad}(n) \mid (q-1)$, $\gcd(n, \operatorname{ord}(f) \cdot k) = 1$ and ($8 \nmid n$ or $q \equiv 1 \pmod 4$)). Let r be a positive integer satisfying $rn \equiv 1 \pmod{\operatorname{ord}(f)}$. Set $d^{(1)} := \gcd(n, q-1)$. Then the factorization of $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$f(X^n) = \prod_{t \mid \frac{n}{d^{(1)}}} \prod_{\substack{1 \leq u \leq d^{(1)} \\ \gcd(u,t)=1}} \zeta_{d^{(1)}}^{-uk} \, g_t(\zeta_{d^{(1)}}^u X^t),$$

where $g_t(X) = \prod_{j=0}^{k-1} (X - \alpha^{trq^j})$ for every divisor t of $\frac{n}{d^{(1)}}$.

Remark 2.52. In Theorem 2.51 the positive integers n and r satisfy $\gcd(n, \operatorname{ord}(f)) = \gcd(n, \operatorname{ord}(\alpha)) = 1$ and $rn \equiv 1 \pmod{\operatorname{ord}(\alpha)}$. Then the element α^r satisfies $(\alpha^r)^n = \alpha$. Consequently, with $\beta = \alpha^r$ there exists an element β in $\mathbb{F}_{q^k}^*$ such that $\beta^n = \alpha$. The authors of [MRS19] claim that their results are generalizations of the factorizations of $X^n - 1$ given in [MGO15] and [WYF18]. As they choose the positive integer r such that $rn \equiv 1 \pmod{\operatorname{ord}(\alpha)}$, their results technically do not hold for $\alpha = 1$ which corresponds to the polynomial $f = X - 1$. This little mistake could easily be corrected by choosing $r = 1$ if $f = X - 1$.

However, even then, the factorizations given in [MRS19] could easily be derived from the factorizations of $X^n - 1$ in [MGO15; WYF18] with our Theorem 2.10 because there exists an element $\beta \in \mathbb{F}_{q^k}^*$ such that $\beta^n = \alpha$. In fact, Theorem 2.10 combined with all results from [MGO15; WYF18; WY21] yields the factorization of $f(X^n)$ for every positive integer n such that there exists $\beta \in \mathbb{F}_{q^k}^*$ satisfying $\beta^n = \alpha$ and $\operatorname{rad}(n) \mid (q^w - 1)$, where $w = 1$ or w is prime or w is the product of two primes, without the condition $\gcd(n, \operatorname{ord}(f) \cdot \deg(f)) = 1$ on n. As a reminder, we state the result again:

Theorem 2.10. *Let $n \in \mathbb{N}$ and $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k. Further, let $\alpha \in \mathbb{F}_{q^k}^*$ be a root of f such that there exists $\beta \in \mathbb{F}_{q^k}^*$ with $\beta^n = \alpha$ and let $Q = \frac{X}{\beta} \in \mathbb{F}_{q^k}(X)$. If $\prod_{R \in \mathcal{R}} R$ is the factorization of $X^n - 1$ into monic irreducible factors over $\mathbb{F}_{q^k}$, then the factorization of $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$\prod_{R \in \mathcal{R}} \operatorname{spin}_q \left[R^Q \right],$$

where $\operatorname{coeffdeg}_q \left(R^Q \right) = k$ for all monic irreducible factors R of $X^n - 1$ over $\mathbb{F}_{q^k}$.

Theorem 2.51 is proved with the same top-down approach as the results in [MGO15] and with the use of the following lemma.

Lemma 2.53 ([Bey77, Proposition 1]). *Let p be a prime such that $p \mid (q-1)$ and let k be a positive integer. Then*

(i) If p is odd, then $\nu_p(q^k-1) = \nu_p(q-1) + \nu_p(k)$.

(ii) If $p = 2$, then $\nu_2(q^k-1) = \begin{cases} \nu_2(q-1) & \text{if } k \text{ is odd,} \\ \nu_2(q-1) + \nu_2(k) + \nu_2(q+1) - 1 & \text{if } k \text{ is even,} \end{cases}$

so that $\nu_p(q^k-1) = \nu_p(q-1) + \nu_p(k)$ if $q \equiv 1 \mod 4$.

From Lemma 2.53 we can derive the following corollary. It shows that the condition $\gcd(n,k) = 1$ in Theorem 2.51, where k is the degree of the monic irreducible polynomial $f \in \mathbb{F}_q[X]$, implies that $d^{(k)}$ equals $d^{(1)}$.

Corollary 2.54. *Let $n, k \in \mathbb{N}$ such that $\operatorname{rad}(n) \mid (q-1)$ and $\gcd(n,k) = 1$. For every positive integer t we set $d^{(t)} := \gcd(n, q^t-1)$. Then*

(i) $d^{(km)} = d^{(m)}$ for every positive integer m,

(ii) $d^{(2k)} = 2^l \cdot d^{(1)}$, where $l = \min\{\max\{0, \nu_2(n) - \nu_2(q-1)\}, \nu_2(q+1)\}$.

Proof. (i) Let p be a prime factor of n. If p is odd, then Lemma 2.53(i) implies that $\nu_p(q^{km}-1) = \nu_p(q^m-1) + \nu_p(k) = \nu_p(q^m-1)$. If $p = 2$, then k is odd, because $\gcd(n,k) = 1$, and $\nu_2(q^{km}-1) = \nu_2(q^m-1)$ with Lemma 2.53(ii).

(ii) Let p be a prime factor of n. If p is odd, then Lemma 2.53(i) implies that $\nu_p(q^{2k}-1) = \nu_p(q-1) + \nu_p(2k) = \nu_p(q-1)$, because $p \neq 2$ and $\gcd(n,k) = 1$. If $p = 2$, then $\nu_2(q^{2k}-1) = \nu_2(q-1) + \nu_2(2k) + \nu_2(q+1) - 1 = \nu_2(q-1) + 1 + \nu_2(q+1) - 1 = \nu_2(q-1) + \nu_2(q+1)$ with Lemma 2.53(ii). □

As discussed in Remark 2.52, the factorization of $f(X^n)$ presented in Theorem 2.51 is defined by the factorization of X^n-1 over $\mathbb{F}_{q^k}$, because $\gcd(n, \operatorname{ord}(f)) = 1$. Our alternative proof goes as follows:

Proof of Theorem 2.51. There exists a positive integer r such that $rn \equiv 1 \pmod{\operatorname{ord}(f)}$ and $\beta = \alpha^r$ satisfies $\beta^n = \alpha$. With the conditions $\operatorname{rad}(n) \mid (q-1)$ and ($8 \nmid n$ or $q \equiv 1 \pmod 4$) the factorization of $X^n - 1$ over $\mathbb{F}_{q^k}$ is given by Theorem 2.12 as:

$$\prod_{t \mid \frac{n}{d^{(k)}}} \prod_{\substack{0 \le u \le d^{(k)}-1 \\ \gcd(u,t)=1}} (X^t - \zeta_{d^{(k)}}^u).$$

Then Theorem 2.10 implies that the factorization of $f(X^n)$ is:

$$\prod_{t \mid \frac{n}{d^{(k)}}} \prod_{\substack{0 \le u \le d^{(k)}-1 \\ \gcd(u,t)=1}} \operatorname{spin}_q\left[(X^t - \zeta_{d^{(k)}}^u)^Q\right] = \prod_{t \mid \frac{n}{d^{(k)}}} \prod_{\substack{0 \le u \le d^{(k)}-1 \\ \gcd(u,t)=1}} \left[\prod_{j=0}^{k-1} (X^t - \beta^{t \cdot q^j} \zeta_{d^{(k)}}^{u \cdot q^j})\right],$$

where $Q = \frac{X}{\beta} \in \mathbb{F}_{q^k}(X)$. Since $\gcd(n,k) = 1$, Corollary 2.54 implies that $d^{(k)} = d^{(1)}$ and $\zeta_{d^{(k)}} = \zeta_{d^{(1)}}$ is an element of $\mathbb{F}_q$. Thus, the factorization of $X^n - 1$ over $\mathbb{F}_{q^k}$ is actually a factorization over $\mathbb{F}_q$ and $\zeta_{d^{(k)}}^{u \cdot q^j} = \zeta_{d^{(k)}}^u$ for all $0 \le j \le k-1$. □

For the proof of the next two theorems, Theorem 2.55 and Theorem 2.57, the authors of [MRS19] use the same method as [WYF18; WY18; WY21] and as we do in Theorem 2.41. They determine the least positive integer s such that $\operatorname{rad}(n) \mid (q^s-1)$ and ($8 \nmid n$ or $q^s \equiv 1 \pmod 4$), obtain the factorization of $f(X^n)$ over $\mathbb{F}_{q^s}$ with Theorem 2.51 and show that the q-spins of the irreducible factors over $\mathbb{F}_{q^s}$ are the irreducible factors of $f(X^n)$ over $\mathbb{F}_q$.

The next theorem, Theorem 2.55, details the factorization of $f(X^n)$ for all positive integers n such that $\gcd(n, \operatorname{ord}(f) \cdot \deg(f)) = 1$ and $\operatorname{rad}(n) \mid (q-1)$, $8 \mid n$ and $q \equiv 3 \pmod 4$.

Theorem 2.55 ([MRS19, Theorem 4.1]). *Let $q \equiv 3 \pmod 4$, $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of an odd degree k and let $\alpha \in \mathbb{F}_{q^k}^*$ be a root of f. Furthermore, let n be a positive integer such that $\operatorname{rad}(n) \mid (q-1)$, $\gcd(n, \operatorname{ord}(f) \cdot k) = 1$ and $8 \mid n$. Let r be a positive integer satisfying $rn \equiv 1 \pmod{\operatorname{ord}(f)}$. Set $d^{(t)} := \gcd(n, q^t - 1)$ for $t \in \{1, 2\}$. Then $d^{(2)} = 2^l \cdot d^{(1)}$, where $l = \min\{\nu_2(\frac{n}{2}), \nu_2(q+1)\}$. Set $\zeta_{d^{(1)}} = \zeta_{d^{(2)}}^{2^l}$. Then the factorization of $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$f(X^n) = \prod_{\substack{t \mid \frac{n}{d^{(2)}} \\ 2 \nmid t}} \prod_{\substack{1 \le v \le d^{(1)} \\ \gcd(v,t)=1}} \zeta_{d^{(1)}}^{-vk} \, g_t(\zeta_{d^{(1)}}^v X^t) \cdot \prod_{t \mid \frac{n}{d^{(2)}}} \prod_{u \in \mathcal{U}_t} \left[\zeta_{d^{(2)}}^{-uk(q+1)} g_t\left(\zeta_{d^{(2)}}^u X^t\right) g_t\left(\zeta_{d^{(2)}}^{uq} X^t\right) \right].$$

where $g_t(X) = \prod_{j=0}^{k-1} (X - \alpha^{trq^j})$ for every divisor t of $\frac{n}{d^{(2)}}$ and

$$\mathcal{U}_t = \{1 \le u \le d^{(2)} : \gcd(u, t) = 1, 2^l \nmid u\} / \sim_{d^{(2)}}.$$

The equivalence relation $\sim_{d^{(2)}}$ is defined as $u \sim_{d^{(2)}} \tilde{u}$ if and only if $\tilde{u} \equiv u \cdot q^i \pmod{d^{(2)}}$ for a positive integer i.

Remark 2.56. In the original statement of Theorem 2.55, [MRS19, Theorem 4.1], the result is stated without the condition $\gcd(n, \operatorname{ord}(f) \cdot \deg(f)) = 1$. However, without this condition the statement of the theorem would be false. For example with Lemma 2.53 the greatest common divisor $d^{(2)}$ of n and $q^2 - 1$ would equal $d^{(1)} \cdot \gcd(n, k) \cdot 2^l$, where $l := \min\{\nu_2(\frac{n}{2}), \nu_2(q+1)\}$, and not satisfy the equation $d^{(2)} = 2^l \cdot d^{(1)}$. Furthermore, the authors prove this theorem using results from their Section 3 and the condition $\gcd(n, \operatorname{ord}(f) \cdot \deg(f))$ is set throughout this section.

Theorem 2.55 could alternatively be proved with Theorem 2.10, Corollary 2.54 and Theorem 2.13.

The following theorem gives the factorization of $f(X^n)$ for every positive integer such that $\gcd(n, \operatorname{ord}(f) \cdot \deg(f)) = 1$ and $\operatorname{rad}(n)$ does not divide $q - 1$ but divides $q^w - 1$ for an odd prime w such that $\gcd(n, w) = 1$.

Theorem 2.57 ([MRS19, Theorem 4.2]). *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and let $\alpha \in \mathbb{F}_{q^k}^*$ be a root of f. Furthermore, let n be a positive integer such that $\operatorname{ord}_{\operatorname{rad}(n)}(q) = w$ is an odd prime, $\gcd(n, \operatorname{ord}(f) \cdot k) = \gcd(n, w) = 1$ and $(8 \nmid n$ or $q \equiv 1 \pmod 4)$. For every positive integer t we set $d^{(t)} = \gcd(n, q^t - 1)$ and we define the equivalence relation $\sim_{d^{(w)}}$ on $\{1, \dots, d^{(w)}\}$ as follows: $u \sim_{d^{(w)}} \tilde{u}$ if and only if $\tilde{u} \equiv u \cdot q^i \pmod{d^{(w)}}$ for a positive integer i. Then the factorization of $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is*

$$f(X^n) = \prod_{\substack{t \mid \frac{n}{d^{(w)}} \\ \operatorname{rad}(t) \mid q-1}} \prod_{\substack{1 \le v \le d' \\ \gcd(v,t)=1}} \zeta_{d'}^{-vk} \, g_t(\zeta_{d'}^v X^t) \cdot \prod_{\substack{t \mid \frac{n}{d^{(w)}} \\ \operatorname{rad}(t) \nmid q-1}} \prod_{u \in \mathcal{U}_t} \left[\zeta_{d^{(w)}}^{-uk(1+\dots+q^{w-1})} g_t(\zeta_{d^{(w)}}^u X^t) \cdots g_t(\zeta_{d^{(w)}}^{uq^{w-1}} X^t) \right],$$

where

(1) if $w \nmid n$ or $w \nmid q - 1$ or $\nu_w(n) > \nu_w(q-1) \ge 1$, $d' = d^{(1)}$ and

$$\mathcal{U}_t = \left\{ 1 \le u \le d^{(w)} : \gcd(u, t) = 1, \gcd\left(n, \frac{q^w - 1}{q - 1}\right) \nmid u \right\} / \sim_{d^{(w)}}.$$

(2) if $w \mid n$ *and* $w \mid q-1$ *and* $\nu_w(n) \geq \nu_w(q-1)$, $d' = w \cdot \gcd(\frac{n}{w}, q-1)$ *and*

$$\mathcal{U}_t = \left\{ 1 \leq u \leq d^{(w)} : \gcd(u,t) = 1, \gcd\left(\frac{n}{w}, \frac{q^w-1}{w\cdot(q-1)}\right) \nmid u \right\} / \sim_{d^{(w)}} .$$

Theorem 2.57 could alternatively be proved with Theorem 2.10, Corollary 2.54 and Theorem 2.17.

Remark 2.58. Note that for a prime w the equivalence classes of the relation $\sim_{d^{(w)}}$ as defined in Theorem 2.55 and Theorem 2.57 on $\{1, \ldots, d^{(w)}\}$ are in fact the q-cyclotomic cosets modulo $d^{(w)}$. Furthermore, if $\mathcal{U}_t = \{1 \leq u \leq d^{(w)} : \gcd(u,t) = 1, m \nmid u\}/\sim_{d^{(w)}}$ for two positive integers t and m such that $\mathrm{rad}(t) \mid \mathrm{rad}(n)$ and $m \mid d^{(w)}$ then Lemma 2.14 implies that

$$\mathcal{U}_t = \{u \in \mathrm{CR}_q(d^{(w)}) : \gcd(u,t) = 1, m \nmid u\}.$$

In Theorem 2.55 we have $d^{(2)} = 2^l \cdot d^{(1)}$ and $m = 2^l$ divides $d^{(2)}$. In Theorem 2.57 both positive integers $m = \gcd\left(n, \frac{q^w-1}{q-1}\right)$ and $m = \gcd\left(\frac{n}{w}, \frac{q^w-1}{w\cdot(q-1)}\right)$ are divisors of $\gcd(n, q^w - 1) = d^{(w)}$. Thus, the index sets $\mathcal{U}_t$ are actually subsets of the representative system $\mathrm{CR}_q(d^{(w)})$ in both theorems, Theorem 2.55 and Theorem 2.57.

2.7 A new closed formula for the factorization of $f(X^n)$

In this section we derive a closed formula for the factorization of $f(X^n)$ for every monic irreducible polynomial $f \in \mathbb{F}_q[X]$, $f \neq X$ and every positive integer n such that $\gcd(n,q) = 1$ from our main theorem, Theorem 2.41. Let f be of degree k and $\alpha \in \mathbb{F}_{q^k}^*$ a root of f.

Remark 2.59. As in Remark 1.2, we can assume $\gcd(n,q) = 1$ without loss of generality. Indeed, if $\gcd(n,q) > 1$, then $n = \tilde{n} \cdot \mathrm{char}(\mathbb{F}_q)^l$ and there exists an element $\gamma \in \mathbb{F}_{q^k}$ such that $\gamma^{\mathrm{char}(\mathbb{F}_q)^l} = \alpha$. Let $\prod_R R$ be the factorization of $X^{\tilde{n}} - \gamma$ over $\mathbb{F}_{q^k}$, then $(\prod_R R)^{\mathrm{char}(\mathbb{F}_q)^l}$ is the factorization of $X^n - \alpha$ over $\mathbb{F}_{q^k}$. With Theorem 2.8 the factorization of $f(X^n)$ over $\mathbb{F}_q$ then is given by $(\prod_R \mathrm{spin}_q[R])^{\mathrm{char}(\mathbb{F}_q)^l}$.

Additionally, without loss of generality, we can restrict ourselves to the monic polynomials f by Remark 2.5.

Recall that $f(X^n)$ is a rational transformation f^Q with the rational function $Q = \frac{X^n}{1} \in \mathbb{F}_q(X)$. By Theorem 2.8, the factorization of any rational transformation f^Q with $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ is given by the q-spins of the irreducible factors of $g - \alpha h$ over $\mathbb{F}_{q^k}$. As a reminder we state the result again:

Theorem 2.8 ([MYM10, Lemma 13])**.** *Let* $f \in \mathbb{F}_q[X]$ *be a monic irreducible polynomial of degree* k *and* $\alpha \in \mathbb{F}_{q^k}$ *be a root of* f*. Further, let* $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ *and* $\prod_{R\in\mathcal{R}} R$ *be the factorization of* $g - \alpha h$ *into irreducible factors over* $\mathbb{F}_{q^k}$*. Then the factorization of* f^Q *into monic irreducible factors over* $\mathbb{F}_q$ *is given by*

$$\prod_{R\in\mathcal{R}} \mathrm{spin}_q[R],$$

where $\mathrm{coeffdeg}_q(R) = k$ *for all irreducible factors* R *of* $g - \alpha h$ *over* $\mathbb{F}_{q^k}$*.*

Thus, with Theorem 2.8 the factorization of $f(X^n)$ over $\mathbb{F}_q$ is given by $\prod_{R\in\mathcal{R}} \mathrm{spin}_q[R]$, where $\prod_{R\in\mathcal{R}} R$ is the factorization of $X^n - \alpha$ into monic irreducible factors over $\mathbb{F}_{q^k}$. The factorization of $X^n - \alpha$ over $\mathbb{F}_{q^k}$ is given by Theorem 2.41 and consists of q^k-spins of the monic irreducible factors of $X^n - \alpha$ over $\mathbb{F}_{q^{ks}}$, where s is either $\mathrm{ord}_{\mathrm{rad}(n)}(q^k)$ or $2 \cdot \mathrm{ord}_{\mathrm{rad}(n)}(q^k)$.

The following fact shows that the q-spin is transitive. We use it in the proof of our next theorem to simplify the factorization of $f(X^n)$ obtained from the combination of Theorem 2.41 and Theorem 2.8.

Fact 2.60. *Let k, s be positive integers and g be an irreducible polynomial over $\mathbb{F}_{q^{ks}}$ such that $k \mid \text{coeffdeg}_q(g)$, then the q-spin of g satisfies*

$$\text{spin}_q[g] = \text{spin}_q\left[\text{spin}_{q^k}[g]\right].$$

Proof. Since $k \mid \text{coeffdeg}_q(g)$, there exists a divisor d of s such that $\text{coeffdeg}_q(g) = kd$ and $\text{coeffdeg}_{q^k}(g) = d$. Then if $g = \sum_{i=0}^{m} a_i X^i$,

$$\begin{aligned}\text{spin}_q\left[\text{spin}_{q^k}[g]\right] &= \text{spin}_q\left[\prod_{j=0}^{d-1}\sum_{i=0}^{m} a_i^{q^{kj}} X^i\right] = \prod_{l=0}^{k-1}\prod_{j=0}^{d-1}\sum_{i=0}^{m} a_i^{q^{kj+l}} X^i \\ &= \prod_{j=0}^{kd-1}\sum_{i=0}^{m} a_i^{q^j} X^i = \text{spin}_q[g],\end{aligned}$$

because q^l is a power of $\text{char}(\mathbb{F}_q)$ and for the coefficients of the polynomial $\prod_{j=0}^{d-1}\sum_{i=0}^{m} a_i^{q^{kj+l}} X^i$ we can write i.e. $(a_i^{q^{kj}} + a_{\tilde{i}}^{q^{k\tilde{j}}})^{q^l} = a_i^{q^{kj+l}} + a_{\tilde{i}}^{q^{k\tilde{j}+l}}$. □

In the next theorem, we give a closed formula for the factorization of $f(X^n)$ for every monic irreducible polynomial $f \in \mathbb{F}_q[X]$, $f \neq X$, and every positive integer n such that $\gcd(n, q) = 1$. It covers and extends Theorem 2.51, Theorem 2.55 and Theorem 2.57.

Theorem 2.61. *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and let n be a positive integer such that $\gcd(n, q) = 1$. We define:*

α	$\in \mathbb{F}_{q^k}^*$, *a root of* f
n_1, n_2	$n = n_1 \cdot n_2$, *where* $\text{rad}(n_1) \mid \text{ord}(\alpha)$ *and* $\gcd(n_2, \text{ord}(\alpha)) = 1$
w	$= \text{ord}_{\text{rad}(n)}(q^k)$
s	$= \begin{cases} w & \text{if } 4 \nmid n \text{ or } q^{kw} \equiv 1 \pmod 4, \\ 2w & \text{otherwise.} \end{cases}$
${d_1}^{(t)}$	$= \gcd(n_1, \frac{q^{kt}-1}{\text{ord}(\alpha)})$ *for* $t \in \{1, s\}$
${d_2}^{(t)}$	$= \gcd(n_2, q^{kt} - 1)$ *for* $t \in \mathbb{N}$
$\sim_\beta$	$\beta \in \mathbb{F}_{q^s}$ *and for* $j, \tilde{j} \in \{0, \ldots, {d_1}^{(s)} - 1\}$ *holds* $j \sim_\beta \tilde{j}$ *if and only if* $\zeta_{{d_1}^{(s)}}^{\tilde{j}} = \zeta_{{d_1}^{(s)}}^{j \cdot q^{ku}} \beta^{q^{ku}-1}$ *for an integer* $0 \leq u \leq s_1 - 1$,
s_1	$= \begin{cases} \frac{{d_1}^{(s)}}{{d_1}^{(1)}} & \text{if } 4 \nmid {d_1}^{(s)} \text{ or } q^k \equiv 1 \pmod 4, \\ \frac{{d_1}^{(s)}}{{d_1}^{(1)} \cdot 2^{\min\{\nu_2(\frac{n}{4}), \nu_2(\frac{q^k+1}{2})\}}} & \text{otherwise.} \end{cases}$
t_i	$= \min\{t \in \mathbb{N} : \frac{{d_2}^{(s)}}{{d_2}^{(t)}} \mid i\}$ *for every* $i \in \text{CR}_{q^k}({d_2}^{(s)})$
c_i	$= \text{lcm}(t_i, s_1)$ *for every* $i \in \text{CR}_{q^k}({d_2}^{(s)})$
r	$\begin{cases} r = 1 & \text{if } f = X - 1, \\ rn_2 \equiv 1 \pmod{\text{ord}(\alpha) \cdot {d_1}^{(s)}} & \text{otherwise.} \end{cases}$

Then there exists $\beta \in \mathbb{F}_{q^{ks}}^*$ such that $\beta^{{d_1}^{(s)}} = \alpha$ and the factorization of $f(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is

$$\prod_{j\in\{0,\dots,{d_1}^{(s)}-1\}/\sim_\beta} \prod_{v\mid\frac{n_2}{{d_2}^{(s)}}} \prod_{\substack{i\in \mathrm{CR}_{q^k}({d_2}^{(s)})\\ \gcd(i,v)=1}} \prod_{m=0}^{\gcd(t_i,s_1)-1} S_{(j,v,i,m)},$$

and for all applicable (j,v,i,m) the polynomial $S_{(j,v,i,m)}$ is defined as

$$\begin{aligned} S_{(j,v,i,m)} &= \mathrm{spin}_q\left[X^{\frac{n_1}{{d_1}^{(s)}}\cdot v} - \zeta_{{d_2}^{(s)}}^{i\cdot q^{km}}(\zeta_{{d_1}^{(s)}}^{j}\beta)^{rv}\right] \\ &= \prod_{u=0}^{kc_i-1}\left(X^{\frac{n_1}{{d_1}^{(s)}}\cdot v} - (\zeta_{{d_2}^{(s)}}^{i\cdot q^{km}}(\zeta_{{d_1}^{(s)}}^{j}\beta)^{rv})^{q^u}\right) \\ &= \sum_{l=0}^{kc_i} X^{\frac{n_1}{{d_1}^{(s)}}\cdot v\cdot(kc_i-l)}\cdot(-1)^l \sum_{\substack{\mathcal{U}\subseteq\{0,\dots,kc_i-1\}\\ |\mathcal{U}|=l}} \prod_{u\in\mathcal{U}} (\zeta_{{d_2}^{(s)}}^{i\cdot q^{km}}(\zeta_{{d_1}^{(s)}}^{j}\beta)^{rv})^{q^u}. \end{aligned}$$

$S_{(j,v,i,m)}$ has degree $k\cdot\frac{n_1}{{d_1}^{(s)}}\cdot v\cdot c_i$ and order $\mathrm{ord}(\alpha)\cdot n_1\cdot v\cdot\frac{{d_2}^{(s)}}{\gcd(i,{d_2}^{(s)})}$.

Proof. The factorization of $X^n-\alpha$ over $\mathbb{F}_{q^k}$ is given by Theorem 2.41 as

$$\prod_{j\in\{0,\dots,{d_1}^{(s)}-1\}/\sim} \prod_{v\mid\frac{n_2}{{d_2}^{(s)}}} \prod_{\substack{i\in \mathrm{CR}_{q^k}({d_2}^{(s)})\\ \gcd(i,v)=1}} \prod_{m=0}^{\gcd(t_i,s_1)-1} S_{(j,v,i,m)}$$

where $S_{(j,v,i,m)}$ is the q^k-spin of the binomial $R_{(j,v,i,m)} := X^{\frac{n_1}{{d_1}^{(s)}}\cdot v} - \zeta_{{d_2}^{(s)}}^{i\cdot q^{km}}(\zeta_{{d_1}^{(s)}}^{j}\beta)^{rv}$ with $\mathrm{coeffdeg}_{q^k}\left(R_{(j,v,i,m)}\right) = c_i$. The positive integer r satisfies $r=1$ if $\alpha=1$, which is equivalent to $f=X-1$, or $rn_2\equiv 1 \pmod{\mathrm{ord}(\alpha)\cdot {d_1}^{(s)}}$ otherwise. Then with Theorem 2.8 the factorization of $f(X^n)=f^Q$ for $Q=\frac{X^n}{1}\in\mathbb{F}_q(X)$ is given by

$$\prod_{j\in\{0,\dots,{d_1}^{(s)}-1\}/\sim} \prod_{v\mid\frac{n_2}{{d_2}^{(s)}}} \prod_{\substack{i\in \mathrm{CR}_{q^k}({d_2}^{(s)})\\ \gcd(i,v)=1}} \prod_{m=0}^{\gcd(t_i,s_1)-1} \mathrm{spin}_q\left[S_{(j,v,i,m)}\right],$$

and $\mathrm{coeffdeg}_q\left(S_{(j,v,i,m)}\right)=k$. With Fact 2.60 holds $\mathrm{spin}_q\left[S_{(j,v,i,m)}\right] = \mathrm{spin}_q\left[R_{(j,v,i,m)}\right]$ and $\mathrm{coeffdeg}_q\left(R_{(j,v,i,m)}\right) = kc_i$. Since $R_{(j,v,i,m)}$ is a binomial, its q-spin can easily be computed:

$$\sum_{l=0}^{kc_i} X^{\frac{n_1}{{d_1}^{(s)}}\cdot v\cdot l}\cdot(-1)^{kc_i-l} \sum_{\substack{\mathcal{U}\subseteq\{0,\dots,kc_i-1\}\\ |\mathcal{U}|=kc_i-l}} \prod_{u\in\mathcal{U}} (\zeta_{{d_2}^{(s)}}^{i\cdot q^{km}}(\zeta_{{d_1}^{(s)}}^{j}\beta)^{rv})^{q^u}$$

The degree of the q-spin of $R_{(j,v,i,m)}$ is $kc_i\cdot\frac{n_1}{{d_1}^{(s)}}\cdot v$ and its order is the same as the order of $R_{(j,v,i,m)}$, which is $\mathrm{ord}(\alpha)\cdot n_1\cdot v\cdot\frac{{d_2}^{(s)}}{\gcd(i,{d_2}^{(s)})}$. □

2.8 A new algorithm for the factorization of $f(X^n)$

With Theorem 2.61, Remark 2.59 and Remark 2.5 we define a new algorithm for the factorization of $f(X^n)$ for every irreducible polynomial $f\in\mathbb{F}_q[X]$, $f\neq X$, and for every positive integer n.

Notation. In the following algorithm we denote the characteristic $\mathrm{char}(\mathbb{F}_q)$ of the finite field $\mathbb{F}_q$ by p.

Algorithm 2 (Factorization of $f(X^n)$)**.**
Input: Field size q with $\mathrm{char}(\mathbb{F}_q) = p$ for a prime p, $n \in \mathbb{N}$,
$f \in \mathbb{F}_q[X]$ monic irreducible of degree k, $f \neq X$.
Output: Factorization of $f(X^n)$ into monic irreducible polynomials over $\mathbb{F}_q$.

2.1. Compute $\nu_p(n)$.
2.2. Set $\tilde{n} := \frac{n}{p^{\nu_p(n)}}$.
2.3. **If** $\mathbb{F}_{q^k}[X] = \mathbb{F}_q/\langle g \rangle$ for a primitive polynomial $g \in \mathbb{F}_q[X]$, then let γ be a root of g.
Else find a primitive element $\gamma \in \mathbb{F}_{q^k}$.
2.4. Find $\alpha \in \mathbb{F}_{q^k}$ such that $f(\alpha) = 0$.
2.5. Compute $\mathrm{ord}(\alpha)$.
2.6. Find $\tilde{\alpha} \in \{\gamma^{\frac{q^k-1}{\mathrm{ord}(\alpha)} \cdot l} : \gcd(l, \mathrm{ord}(\alpha)) = 1\}$ such that $\tilde{\alpha}^{p^{\nu_p(n)}} = \alpha$.
2.7. Apply Steps **1.1.** to **1.16.** of Algorithm 1 for input $q^k, \tilde{n}$ and $\tilde{\alpha}$.
2.8. **For** all $j \in \{0, \ldots, {d_1}^{(s)} - 1\}/\sim$,
for all $v \mid \frac{\tilde{n}}{{d_2}^{(s)}}$,
for all $i \in \mathrm{CR}_{q^k}({d_2}^{(s)})$ such that $\gcd(i, v) = 1$,
for all $m \in \{0, \ldots, \gcd(t_i, s_1) - 1\}$:
Compute

$$S_{(j,v,i,m)} = \prod_{u=0}^{kc_i-1} \left(X^{\frac{n_1}{{d_1}^{(s)}} \cdot v} - (\zeta_{{d_2}^{(s)}}^{i \cdot q^{km}} b_j^v)^{q^u} \right).$$

2.9. The factorization of $f(X^n)$ is

$$\prod_{j \in \{0,\ldots,{d_1}^{(s)}-1\}/\sim} \prod_{v \mid \frac{\tilde{n}}{{d_2}^{(s)}}} \prod_{\substack{i \in \mathrm{CR}_{q^k}({d_2}^{(s)}) \\ \gcd(i,v)=1}} \prod_{m=0}^{\gcd(t_i,s_1)-1} S_{(j,v,i,m)}^{p^{\nu_p(n)}}$$

Algorithm 2 is a generalization of Algorithm 1, because the polynomial $X - a \in \mathbb{F}_q[X]$ is irreducible over $\mathbb{F}_q$ for all $a \in \mathbb{F}_q^*$ and its composition with X^n is the binomial $X^n - a$. Consequently, it suffices to implement and use Algorithm 2.

Remark 2.62. For applicable (j, v, i, m) the polynomial $S_{(j,v,i,m)}$ is a composition of the form $g(X^{\frac{n_1}{{d_1}^{(s)}} \cdot v})$, where

$$g = \prod_{u=0}^{kci-1} \left(X - (\zeta_{{d_2}^{(s)}}^{i \cdot q^{km}} b_j^v)^{q^u} \right).$$

Therefore, instead of computing $S_{(j,v,i,m)}$ directly, we can compute g first and $g(X^{\frac{n_1}{{d_1}^{(s)}} \cdot v})$ afterwards. Computations with polynomials of a high degree are costly in computer algebra systems. Thus, for implementations this adaptation of **2.8.** has the advantage that the computation of the product over $0 \leq u \leq kc_i - 1$ happens with polynomials of much smaller degree.

In [MR18] an algorithm has been presented for the factorization of $f(X^n)$ for a monic irreducible polynomial $f \in \mathbb{F}_q[X]$ and every positive integer n such that $\nu_p(q-1) \geq \nu_p(n) + \nu_p(\mathrm{ord}(f))$ for every prime factor p of n. Note that this condition is equivalent to

$n \mid \frac{q-1}{\gcd(q-1,\mathrm{ord}(f))}$. Let f be of degree k and $\alpha \in \mathbb{F}_{q^k}$ be a root of f. The following lemma shows that the binomial $X^n - \alpha$ splits into linear factors over $\mathbb{F}_{q^k}$ if $n \mid \frac{q-1}{\gcd(q-1,\mathrm{ord}(f))} = \frac{q-1}{\gcd(q-1),\mathrm{ord}(\alpha)}$. In fact, the condition can be relaxed to $n \mid \frac{q^k-1}{\mathrm{ord}(\alpha)}$.

Lemma 2.63. *Let $a \in \mathbb{F}_q^*$ and n be a positive integer such that $n \mid \frac{q-1}{\mathrm{ord}(a)}$. Furthermore, write $n = n_1 \cdot n_2$ such that $\mathrm{rad}(n_1) \mid \mathrm{ord}(a)$ and $\gcd(n_2, \mathrm{ord}(a)) = 1$ and let r be a positive integer satisfying $r = 1$ if $a = 1$ and $rn_2 \equiv 1 \pmod{n_1 \cdot \mathrm{ord}(a)}$ otherwise. Then the factorization of the binomial $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is:*

$$X^n - a = \prod_{i=0}^{n-1} (X - \zeta_n^i b^r),$$

where $b \in \mathbb{F}_q$ satisfies $b^{n_1} = a$.

Proof. If $4 \mid n$, then $4 \mid q-1$ and $q \equiv 1 \pmod 4$. Let $n = n_1 \cdot n_2$ such that $\mathrm{rad}(n_1) \mid \mathrm{ord}(a)$ and $\gcd(n_2, \mathrm{ord}(a)) = 1$. Then the fact that $n \mid \frac{q-1}{\mathrm{ord}(a)}$ implies that $d_1 = \gcd(n_1, \frac{q-1}{\mathrm{ord}(a)}) = n_1$ and $d_2 = \gcd(n_2, q-1) = n_2$. Furthermore, we have $\mathrm{rad}(n) \mid q-1$ and the factorization of $X^n - a$ into monic irreducible factors over $\mathbb{F}_q$ is given by Theorem 2.37 as

$$\prod_{j=0}^{d_1-1} \prod_{v \mid \frac{n_2}{d_2}} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{d_2-1} (X^{\frac{n_1}{d_1} \cdot v} - \zeta_{d_2}^i \zeta_{d_1}^j b^{rv}) = \prod_{j=0}^{n_1-1} \prod_{i=0}^{n_2-1} (X - \zeta_{n_2}^i \zeta_{n_1}^j b^r),$$

where $b \in \mathbb{F}_q$ satisfies $b^{n_1} = a$ and r is a positive integer satisfying $r = 1$ if $a = 1$ and $r \cdot n_2 \equiv 1 \pmod{n_1 \cdot \mathrm{ord}(a)}$ otherwise. Since $\gcd(n_1, n_2) = 1$, there exists a primitive $n = n_1 n_2$-th root of unity ζ_n in $\mathbb{F}_q$ and we can write $X^n - a = \prod_{i=0}^{n-1} (X - \zeta_n^i b^r)$. □

From Lemma 2.63 and Theorem 2.8 follows that if $n \mid \frac{q^k-1}{\mathrm{ord}(f)}$ for a monic irreducible polynomial $f \in \mathbb{F}_q[X]$ of degree k, then the factorization of $f(X^n)$ over $\mathbb{F}_q$ equals

$$\prod_{i=0}^{n-1} \mathrm{spin}_q \left[X - \zeta_n^i \beta^r\right] = \prod_{i=0}^{n-1} \left[\prod_{j=0}^{k-1} (X - (\zeta_n^i \beta^r)^{q^j})\right], \tag{2.19}$$

where $\beta^{n_1} = \alpha$ for a root α of f and a positive integer r satisfying $r = 1$ if $\alpha = 1$ and $rn_2 \equiv 1 \pmod{\mathrm{ord}(\alpha) \cdot n_1}$ otherwise. The positive integers n_1, n_2 are defined by $n = n_1 \cdot n_2$, $\mathrm{rad}(n_1) \mid \mathrm{ord}(\alpha)$ and $\gcd(n_2, \mathrm{ord}(\alpha)) = 1$. The algorithm presented in [MR18] does not compute the formula given in (2.19) directly. It is defined only for prime powers $n = p^l$ for a positive integer l, which satisfy the condition $\nu_p(q-1) \geq l + \nu_p(\mathrm{ord}(f))$. If $n = p_1^{l_1} \cdots p_m^{l_m}$ for distinct prime factors $p_1, \ldots, p_m$, then the algorithm in [MR18] needs to be applied m times.

Note that the set of integers for which the algorithm from [MR18] can be applied is the set of divisors of $\frac{q-1}{\gcd(q-1,\mathrm{ord}(f))}$. If q is small or $\gcd(q-1, \mathrm{ord}(f))$ is large, then the number of divisors of $\frac{q-1}{\gcd(q-1,\mathrm{ord}(f))}$ is very small - might even be 1 - and also the divisors themselves are smaller than $q-1$. However, the authors state that their algorithm performs best if q is not very big, $q < e^{(kn)^{0.5626}}$. Thus, the selection of n is quite restrictive and the algorithm can only be applied for a limited number of polynomials f and integers n.

Since our algorithm allows us to factor the composition $f(X^n)$ for every positive integer n with one closed formula, it is much more powerful than the algorithm given in [MR18]. In particular, it allows us to compute the factorization of the composition $f(X^n)$ over the binary field $\mathbb{F}_2$, which is not possible for any positive integer n with the condition $n \mid \frac{q-1}{\gcd(q-1),\mathrm{ord}(f))}$.

In the following example, we factor the composition $f(X^n)$ for the monic irreducible polynomial $f = X^6 + X^5 + X^4 + X^2 + 1 \in \mathbb{F}_2[X]$ and the positive integer $n = 182\,952 = 2^3 \cdot 3^3 \cdot 7 \cdot 11^2$ over the binary field $\mathbb{F}_2$ with Algorithm 2. Then $\tilde{n} = 3^3 \cdot 7 \cdot 11^2$ satisfies $\gcd(\tilde{n}, 2) = 1$ and $\mathrm{ord}_{\mathrm{rad}(\tilde{n})}(2) = \mathrm{ord}_{3\cdot 7\cdot 11}(2) = 30$ is not prime. Furthermore, $\mathrm{ord}(f) = 21$ and $\gcd(\tilde{n}, \deg(f)) = 3 = \gcd(\tilde{n}, \mathrm{ord}(f))$. Thus, the factorization of $f(X^n)$ over $\mathbb{F}_2$ cannot be determined with either Theorem 2.51, Theorem 2.55, Theorem 2.57 or the algorithm presented in [MR18].

Example 2.64. Let $f = X^6 + X^5 + X^4 + X^2 + 1 \in \mathbb{F}_2[X]$ and $n = 182\,952 = 2^3 \cdot 3^3 \cdot 7 \cdot 11^2$.

2.1. We have $\mathrm{char}(\mathbb{F}_2) = 2$ and $\nu_2(n) = 3$.

2.2. $\tilde{n} := \frac{n}{2^{\nu_2(n)}} = 3^3 \cdot 7 \cdot 11^2 = 22\,869$.

2.3. Let $\mathbb{F}_{q^k} = \mathbb{F}_{2^6} = \mathbb{F}_{64} = \mathbb{F}_2(\gamma)$, where γ is a root of the monic irreducible and primitive polynomial $g = X^6 + X^4 + X^3 + X + 1 \in \mathbb{F}_2[X]$.

2.4. Then $\alpha = \gamma^3$ is a root of f.

2.5. The order of α is $21 = 7 \cdot 3$.

2.6. $\tilde{\alpha} = \gamma^5 + \gamma^3 + \gamma + 1 \in \mathbb{F}_{2^6}$ satisfies $\tilde{\alpha}^{2^3} = \alpha$.

2.7. **1.1.** The multiplicative order of 2 modulo $\mathrm{rad}(\tilde{n}) = 3 \cdot 7 \cdot 11 = 231$ equals $w = \mathrm{ord}_{231}(2) = 5$.

1.2. Since $4 \nmid \tilde{n}$, we set $s := w = 5$.

1.3. The order of $\tilde{\alpha}$ is the same as the order of α, because $\mathrm{ord}(\alpha) = \frac{\mathrm{ord}(\tilde{\alpha})}{\gcd(2^3, \mathrm{ord}(\tilde{\alpha}))}$ and $2 \nmid \mathrm{ord}(\tilde{\alpha})$. Thus, $\mathrm{ord}(\tilde{\alpha}) = 21$.

1.4. Thus, $n_1 = 3^3 \cdot 7$ and $n_2 = 11^2$.

1.5. We have $q^{ks} - 1 = 2^{6\cdot 5} - 1 = 3^2 \cdot 7 \cdot 11 \cdot 31 \cdot 151 \cdot 331$ and

$$d_1{}^{(5)} = \gcd(3^3 \cdot 7, \frac{3^2 \cdot 7 \cdot 11 \cdot 31 \cdot 151 \cdot 331}{3 \cdot 7}) = 3.$$

Furthermore, $q^k - 1 = 2^6 - 1 = 63 = 3^2 \cdot 7$ and $d_1{}^{(1)} = \gcd(3^3 \cdot 7, \frac{3^2 \cdot 7}{3 \cdot 7}) = 3$.

1.6. Since $4 \nmid d_1{}^{(s)}$, we set $s_1 := \frac{d_1{}^{(s)}}{d_1{}^{(1)}} = 1$.

1.7. $d_2{}^{(5)} = \gcd(11^2, 2^{6\cdot 5} - 1) = \gcd(11^2, 3^2 \cdot 7 \cdot 11 \cdot 31 \cdot 151 \cdot 331) = 11$.

1.8. $s_2 = \mathrm{ord}_{11}(2^6) = 5$.

1.9. The set of all divisors of $s_2 = 5$ is $\mathcal{T} = \{1, 5\}$. We determine $d_2{}^{(t)}$ for every $t \in \mathcal{T}$:

$$d_2{}^{(1)} = \gcd(n_2, q^k - 1) = \gcd(11^2, 63) = 1,$$
$$d_2{}^{(5)} = \gcd(11^2, 2^{6\cdot 5} - 1) = 11, \qquad \text{(see above)}$$

Thus, $t_i \in \mathcal{T} = \{1, 5\}$ for all $i \in \mathrm{CR}_{q^k}(d_2{}^{(s)}) = \mathrm{CR}_{2^6}(11)$.

1.10. The positive integer $r = 25$ satisfies $r \cdot n_2 = r \cdot 11^2 \equiv 1 \pmod{21 \cdot 3}$.

1.11. Let $\mathbb{F}_{2^{30}} = \mathbb{F}_2(\delta)$, where δ is a root of the monic irreducible and primitive polynomial $h = X^{30}+X^{17}+X^{16}+X^{13}+X^{11}+X^7+X^5+X^3+X^2+X+1 \in \mathbb{F}_2[X]$.

1.12. Then

$$\begin{aligned}
\zeta_{d_1{}^{(5)}} = \zeta_3 =& \delta^{29} + \delta^{27} + \delta^{26} + \delta^{25} + \delta^{24} + \delta^{23} + \delta^{20} + \delta^{19} + \delta^{17} + \delta^{16} \\
&+ \delta^{15} + \delta^{13} + \delta^{12} + \delta^8 + \delta^7 + \delta^6 + \delta^2 + \delta, \\
\zeta_{d_2{}^{(5)}} = \zeta_{11} =& \delta^{29} + \delta^{25} + \delta^{24} + \delta^{23} + \delta^{21} + \delta^{19} + \delta^{17} + \delta^{16} + \delta^{12} + \delta^{10} \\
&+ \delta^8 + \delta^5 + \delta^3, \\
\zeta_{d_1{}^{(5)}\cdot 21} = \zeta_{63} =& \delta^{28} + \delta^{27} + \delta^{24} + \delta^{22} + \delta^{21} + \delta^{20} + \delta^{17} + \delta^{15} + \delta^{14} + \delta^{13} \\
&+ \delta^{10} + \delta^8 + \delta^7 + \delta^4
\end{aligned}$$

1.13. An element $\beta \in \mathbb{F}_{2^{30}}$ such that $\beta^{{d_1}^{(5)}} = \beta^3 = \alpha$ is

$$\beta = \delta^{29} + \delta^{28} + \delta^{27} + \delta^{25} + \delta^{18} + \delta^{17} + \delta^{15} + \delta^{12} + \delta^{11} + \delta^9 + \delta^8 + \delta^4.$$

1.14.-1.15. Since $s_1 = 1$, we know that the equivalence class of every $0 \leq j \leq {d_1}^{(5)} - 1 = 2$ contains only j itself. Therefore, we can set ${}^{\{0,1,2\}}/_{\sim_\beta} = \{0, 1, 2\}$ without any further computation.

1.16. We determine the set of the 2^6-cyclotomic representatives modulo ${d_2}^{(5)} = 11$, $\mathrm{CR}_{2^6}({d_2}^{(5)}) = \mathrm{CR}_{64}(11)$, and for every $i \in \mathrm{CR}_{64}(11)$ we compute

$$t_i = \min\left\{t \in \mathcal{T} : \frac{{d_2}^{(5)}}{{d_2}^{(t)}} \mid i\right\} = \min\left\{t \in \{1, 5\} : \frac{11}{{d_2}^{(t)}} \mid i\right\}$$

and $c_i = \mathrm{lcm}(t_i, s_1) = \mathrm{lcm}(t_i, 1) = t_i$.

i	t_i	c_i	$\mathrm{C}_{64,11}(i)$
0	1	1	$\{0\}$
1	5	5	$\{1, 3, 4, 5, 9\}$
2	5	5	$\{2, 6, 7, 8, 10\}$

Consequently, $\mathrm{CR}_{64}(11) = \{0, 1, 2\}$. Note that for every $i \in \mathrm{CR}_{64}(11)$ we have $\gcd(t_i, s_1) = \gcd(t_i, 1) = 1$ and every cyclotomic coset induces exactly one monic irreducible factor of $f(X^n)$ for $m = 0$.

2.8. The divisors of $\frac{n_2}{{d_2}^{(5)}} = \frac{11^2}{11} = 11$ are $\{1, 11\}$ and $\frac{n_1}{{d_1}^{(5)}} = \frac{3^3 \cdot 7}{3} = 3^2 \cdot 7$. Thus, for all $j \in \{0, 1, 2\}, v \in \{1, 11\}, i \in \{0, 1, 2\}$ such that $\gcd(i, v) = 1$ and $m = 0$ we compute

$$\mathrm{spin}_q\left[X^{3^2 \cdot v} - \zeta_{d_2}^{i \cdot q^m} \zeta_{d_1}^j \beta^{rv}\right] = \prod_{l=0}^{k \cdot c_i} (X^{3^2 \cdot v} - (\zeta_{d_2}^i \zeta_{d_1}^j \beta^{rv})^{q^l}).$$

For $i = 0$, only $v = 1$ satisfies $\gcd(i, v) = 1$ and therefore $i = 0$ yields exactly three monic irreducible factors of degree $3^2 \cdot 7 \cdot 6 = 378$. For $i \in \{1, 2\}$ both $v = 1$ and $v = 11$ satisfy $\gcd(i, v) = 1$ and they each yield exactly three monic irreducible factors of degree $3^2 \cdot 7 \cdot 6 \cdot 5 = 1890$ and exactly three monic irreducible factors of degree $3^2 \cdot 7 \cdot 11 \cdot 6 \cdot 5 = 20\,790$.

(j, v, i, m)	$\mathrm{spin}_q\left[X^{3^2 \cdot v} - \zeta_{d_2}^{i \cdot q^m} \zeta_{d_1}^j \beta^{rv}\right]$
$(0, 1, 0, 0)$	$X^{378} + X^{252} + X^{189} + X^{63} + 1$
$(0, 1, 1, 0)$	$X^{1890} + X^{1827} + X^{1701} + X^{1638} + X^{1449} + X^{1386} + X^{1260}$ $+ X^{1197} + X^{1134} + X^{1008} + X^{945} + X^{882} + X^{756} + X^{630}$ $+ X^{567} + X^{504} + X^{378} + X^{126} + 1$
$(0, 1, 2, 0)$	$X^{1890} + X^{1827} + X^{1638} + X^{1575} + X^{1449} + X^{1386} + X^{1323}$ $+ X^{1134} + X^{1071} + X^{882} + X^{819} + X^{756} + X^{630} + X^{441}$ $+ X^{378} + X^{315} + X^{189} + X^{63} + 1$
$(0, 11, 1, 0)$	$X^{20790} + X^{20097} + X^{19404} + X^{18711} + X^{18018} + X^{16632}$ $+ X^{15246} + X^{13860} + X^{12474} + X^{11781} + X^{10395} + X^{9702}$ $+ X^{9009} + X^{8316} + X^{5544} + X^{4851} + X^{4158} + X^{3465}$ $+ X^{2772} + X^{1386} + 1$
$(0, 11, 2, 0)$	$X^{20790} + X^{18018} + X^{17325} + X^{15246} + X^{14553} + X^{13167}$

	$+ X^{9009} + X^{7623} + X^{6237} + X^{5544} + X^{3465} + X^{2772}$ $+ X^{1386} + X^{693} + 1$
$(1, 1, 0, 0)$	$X^{378} + X^{315} + 1$
$(1, 1, 1, 0)$	$X^{1890} + X^{1827} + X^{1764} + X^{1638} + X^{1512} + X^{1449} + X^{1260}$ $+ X^{1134} + X^{1008} + X^{945} + X^{756} + X^{693} + X^{315} + X^{252}$ $+ X^{126} + X^{63} + 1$
$(1, 1, 2, 0)$	$X^{1890} + X^{1701} + X^{1638} + X^{1575} + X^{1512} + X^{1449} + X^{1386}$ $+ X^{1323} + X^{1197} + X^{1071} + X^{1008} + X^{945} + X^{567} + X^{441}$ $+ X^{189} + X^{63} + 1$
$(1, 11, 1, 0)$	$X^{20790} + X^{20097} + X^{18018} + X^{17325} + X^{15939} + X^{15246}$ $+ X^{14553} + X^{12474} + X^{11781} + X^{9702} + X^{9009} + X^{8316}$ $+ X^{6930} + X^{4851} + X^{4158} + X^{3465} + X^{2079} + X^{693} + 1$
$(1, 11, 2, 0)$	$X^{20790} + X^{20097} + X^{18711} + X^{18018} + X^{15939} + X^{15246}$ $+ X^{13860} + X^{13167} + X^{12474} + X^{11088} + X^{10395} + X^{9702}$ $+ X^{8316} + X^{6930} + X^{6237} + X^{5544} + X^{4158} + X^{1386} + 1$
$(2, 1, 0, 0)$	$X^{378} + X^{315} + X^{126} + X^{63} + 1$
$(2, 1, 1, 0)$	$X^{1890} + X^{1638} + X^{1575} + X^{1386} + X^{1323} + X^{1197} + X^{819}$ $+ X^{693} + X^{567} + X^{504} + X^{315} + X^{252} + X^{126} + X^{63} + 1$
$(2, 1, 2, 0)$	$X^{1890} + X^{1827} + X^{1764} + X^{1701} + X^{1638} + X^{1512} + X^{1386}$ $+ X^{1260} + X^{1134} + X^{1071} + X^{945} + X^{882} + X^{819} + X^{756}$ $+ X^{504} + X^{441} + X^{378} + X^{315} + X^{252} + X^{126} + 1$
$(2, 11, 1, 0)$	$X^{20790} + X^{18711} + X^{18018} + X^{17325} + X^{16632} + X^{15939}$ $+ X^{15246} + X^{14553} + X^{13167} + X^{11781} + X^{11088} + X^{10395}$ $+ X^{6237} + X^{4851} + X^{2079} + X^{693} + 1$
$(2, 11, 2, 0)$	$X^{20790} + X^{20097} + X^{19404} + X^{18018} + X^{16632} + X^{15939}$ $+ X^{13860} + X^{12474} + X^{11088} + X^{10395} + X^{8316} + X^{7623}$ $+ X^{3465} + X^{2772} + X^{1386} + X^{693} + 1$

In the factorization of

$$f(X^n) = f(X^{\tilde{n} \cdot 8}) = f(X^{182\,952}) = X^{1\,097\,712} + X^{914\,760} + X^{731\,808} + X^{365\,904} + 1$$

every monic irreducible factor listed above appears with multiplicity 8.

The computation of Example 2.64 was done with our implementation of Algorithm 2, which we discuss in more detail in the following section.

2.9 Implementation of Algorithm 2

In Appendix B.4 we present an implementation of Algorithm 2, the SageMath file B.8 `fXnAlgorithm.sage`, which can be preparsed and loaded as a Python module (see Section 1.3). The algorithm is called with the function `factor_fXn(f,n,printing)`, where `f` is an irreducible polynomial given as a `RichPolynomial` instance, `n` is a positive integer and `printing` is either `True` or `False` depending on whether all steps of the algorithm should be printed or not. Thus, you can use it as follows:

Source Code 2.1: How to use fXnAlgorithm.sage

```
from fXnAlgorithm import *
from UsefulFunctions import *  # for Magma-style printing

q = 2
RFF = RichFiniteField(q,False,"X")
f = RichPolynomial([1,1,1,0,1,0,1], RFF)
n = 2^3*7*3^3*11^2

# with printing all algorithm steps:
print(factor_fXn(f,n,1))

# without printing all algorithm steps:
print(factor_fXn(f,n,0))

# in Magma-style:
print_Magma_style(factor_fXn(f,n,0))
```

The module `fXnAlgorithm` makes heavy use of the classes `RichPolynomial` and `RichFiniteField`. Even though they are practical, both classes are not very efficient. Our implementation could be improved by removing the dependency of the `Rich` classes, by implementing it in the programming language C and in many other ways. However, even our naive and slow implementation in Python performs much better than the existing SageMath function `factor` for univariate polynomials over finite fields. For large positive integers n our implementation also performs better than Magma's function `Factorization` for univariate polynomials.

The SageMath function `factor` calls the factorization of univariate polynomials from the computer algebra system PARI[1] which is written in C. PARI uses the Berlekamp-Zassenhaus algorithm with improvements by van Hoeij [Ber67; Ber70; CZ81; Van02]. As PARI, Magma's function `Factorization` for univariate polynomials in the computer algebra system Magma uses the Berlekamp-Zassenhaus algorithm with the improvements by Van Hoeij for the factorization of univariate polynomials over "very small finite fields". "For medium to large finite fields, the von zur Gathen/Kaltofen/Shoup algorithm is used by default."[2] (see [Van02; VS92; KS95; Sho95]) Magma's kernel is also written in C. The developers of the SageMath implementation wrote a *Performance Note* in the file `polynomial_element.pyx` of the SageMath source code (version 9.5). In this note they mention that Magma's performance is better than PARI's even though they use the same algorithm.

In the following tables we present computation time comparisons (CPU time) of our function `factor_fXn` with SageMath's `factor` and with Magma's `Factorization`. The measurements of our implementation and the SageMath function `factor` were done with the program B.9 `Ch2-TestNewAlgorithm.sage`. This program is designed for testing our implementation of Algorithm 2, measuring its computation time and comparing it with the SageMath function `factor`. The computations of Magma's `Factorization` function were measured with the script B.10 `Ch2-MagmaMeasurements.txt`.

[1] `https://pari.math.u-bordeaux.fr/`

[2] `https://Magma.maths.usyd.edu.au/Magma/handbook/text/237#2058`[February 14, 2024]

The interested reader might want to compare the computation times of our implementation and the SageMath function `factor` her- or himself. In this case, specify the examples you wish to compute as tuples (q, n, f) in the variable `qnfs` (look for `USER INPUT`). By default the program uses the module `multiprocessing` to measure the computations for all tuples in `qnfs` in parallel. However, the PARI implementation is not compatible with the module `signal` which is used for the computation timeout then. Set the variable `parallel_computations` to `False` if you run into problems and the program will execute the measurements consecutively.

The computations presented here have been done on a DL380 Gen10 server with 384GB of RAM. In the column of the SageMath CPU time appears the entry `stackovf` which means that the computation was terminated by a PARI Error due to a stack overflow. Note that the stack size is increased until the computer memory (RAM) is full. Thus, for computers with less RAM the SageMath function already exits due to a stack overflow for much smaller positive integers. For some (q, n, f) SageMath and/or Magma did not return a result after 600 seconds which we denote by a computation time of ∞ seconds.

First, we consider the monic irreducible polynomial $f = X^6 + X^5 + X^4 + X^2 + 1 \in \mathbb{F}_2[X]$ of order $21 = 3 \cdot 7$ from Example 2.64. The following table shows a comparison of the computation times for the factorization of the composition $f(X^n)$ for given positive integers n. The parameter s is either $\mathrm{ord}_{\mathrm{rad}(n)}(2^6)$ or $2 \cdot \mathrm{ord}_{\mathrm{rad}(n)}(2^6)$ as defined in Theorem 2.61. In the first row we present the computation times for Example 2.64.

Table 2.8: CPU times for $f = X^6 + X^5 + X^4 + X^2 + 1 \in \mathbb{F}_2[X]$

	n	s	SageMath	Magma	New Alg	SM:NA	MA:NA
1.1	$182\,952 = 2^3 \cdot 3^3 \cdot 7 \cdot 11^2$	5	77.727 s	1.240 s	0.727 s	107:1	2:1
1.2	$361 = 19^2$	3	0.020 s	0.000 s	0.232 s	1:11	0:1
1.3	$6\,859 = 19^3$	3	5.038 s	0.190 s	0.292 s	17:1	1:2
1.4	$130\,321 = 19^4$	3	∞ s	25.660 s	0.376 s	∞:1	68:1
1.5	$2\,476\,099 = 19^5$	3	`stackovf`	∞ s	0.530 s	∞:1	∞:1
1.6	$47\,045\,881 = 19^6$	3	∞ s	∞ s	1.764 s	∞:1	∞:1

Table 2.8 shows that for small positive integers (1.2 and 1.3) SageMath and/or Magma are faster than our implementation. We think that for small positive integers the computation of the preliminary parameters and representative systems is costly. However, our algorithm yields a result after less than 0.3 seconds and it is questionable whether in applications this difference in performance is relevant. For large positive integers (1.1, 1.4, 1.5 and 1.6) our algorithm becomes much faster than SageMath's and Magma's. For 1.4, 1.5 and 1.6, SageMath's and/or Magma's computation time explodes such that our algorithm yields a result after less than 2 seconds where both other computer algebra systems do not return a result after 600 seconds.

For SageMath's function the stack size needs to be adjusted to perform the computations. The stack sizes for the computations in Table 2.8 are shown in the following table. The stack size of 1.5 (401 353 109 504) is the maximum possible on a 384 GB RAM computer because PARI exits due to a stack overflow (see 1.5 in Table 2.8). We believe that the stack size is equally adjusted during the Magma computations because they use the same algorithm and the function also runs out of memory for the next example (2.6 in Table 2.11). Since Magma is closed-source we do not know for sure.

	n	SageMath stack size
1.1	$182\,952 = 2^3 \cdot 3^3 \cdot 7 \cdot 11^2$	8 192 000 000
1.2	$361 = 19^2$	2 000 000
1.3	$6\,859 = 19^3$	512 000 000
1.4	$130\,321 = 19^4$	262 144 000 000
1.5	$2\,476\,099 = 19^5$	401 353 109 504
1.6	$47\,045\,881 = 19^6$	

For $n = 19^6$ (1.6) SageMath does not give any information on the stack size. We believe that SageMath has problems computing the polynomial $f(X^{19^6})$ itself because of its very high degree:

$$f(X^{19^6}) = X^{282\,275\,286} + X^{235\,229\,405} + X^{188\,183\,524} + X^{94\,091\,762} + 1 \in \mathbb{F}_2[X].$$

In Algorithm 2 we do not need to compute the polynomial $f(X^n)$ itself. Thus, our implementation only needs to handle the degrees of the irreducible factors of $f(X^n)$. Let $s, k, n_1, {d_1}^{(s)}, \tilde{n}, {d_2}^{(s)}$ be defined as in Algorithm 2. Then the maximum degree appearing in the computation of Algorithm 2 is less than or equal to $s \cdot k \cdot \frac{n_1}{{d_1}^{(s)}} \cdot \frac{\tilde{n}}{{d_2}^{(s)}}$ because $c_i \leq s$ holds for all $i \in \mathrm{CR}_{q^k}({d_2}^{(s)})$. In general this number is much smaller than $k \cdot n$, the degree of $f(X^n)$. The following table shows the largest degree appearing in the factorization of $f(X^n)$ for the computations from Table 2.8.

	n	$6n$	max degree
1.1	$182\,952 = 2^3 \cdot 3^3 \cdot 7 \cdot 11^2$	1 097 712	20 790
1.2	$361 = 19^2$	2 052	342
1.3	$6\,859 = 19^3$	41 154	6 498
1.4	$130\,321 = 19^4$	781 926	123 462
1.5	$2\,476\,099 = 19^5$	14 856 594	2 345 778
1.6	$47\,045\,881 = 19^6$	282 275 286	44 569 782

Note that the Berlekamp-Zassenhaus algorithm first factors f over a suitable prime field $\mathbb{F}_p$ and then lifts the solution to $q = p^l$ for a positive integer l. Consequently, both SageMath and Magma perform best over prime fields. In the following computations we chose $g = X^6 + X^3 + \delta \in \mathbb{F}_4[X]$, where $F_4 = \mathbb{F}(\delta)$ and δ is a root of the monic irreducible polynomial $X^2 + X + 1 \in \mathbb{F}_2[X]$. SageMath returned a result only for $n = 19^2$ and Magma only for $n = 19^2$ and $n = 19^3$.

Table 2.11: CPU times for $g = X^6 + X^3 + \delta \in \mathbb{F}_4[X]$

	n	s	SageMath	Magma	New Alg	SM:NA	MA:NA
2.1	$182\,952 = 2^3 \cdot 3^3 \cdot 7 \cdot 11^2$	5	∞ s	∞ s	4.203s	∞:1	∞:1
2.2	$361 = 19^2$	3	468.408 s	0.030 s	0.516 s	908:1	1:17
2.3	$6\,859 = 19^3$	3	∞ s	62.160 s	0.787 s	∞:1	79:1

2.4	$130\,321 = 19^4$	3	∞ s	∞ s	2.332 s	∞:1	∞:1
2.5	$2\,476\,099 = 19^5$	3	∞ s	∞ s	28.634 s	∞:1	∞:1
2.6	$47\,045\,881 = 19^6$	3	∞ s	`outofmem`	533.414 s	∞:1	∞:1

For $n = 19^6$ (2.6), Magma's computation is terminated by the following error, which we denote by `outofmem`:

```
Current total memory usage: 75524.6MB, failed memory request:
    3727186.0MB
System error: Out of memory.
All virtual memory has been exhausted so Magma cannot perform this
    statement.
```

Thus, Magma was already using 75.5GB of RAM and would need at least 3 727GB of RAM to perform the computation for $n = 19^6$. Recall that the computations presented here were done on a computer with 384GB of RAM. Most computers do not even have 75.5GB of RAM. Home computers are sold with 8 to 32GB RAM in stores at the moment. Thus, on computers with up to 32GB of RAM both SageMath and Magma would not return factorizations even for small positive integers n.

The biggest advantage of our algorithm is that it uses almost no memory. The only values which need to be stored are the parameters computed in steps **1.1.** to **1.16.**, and the irreducible factors of $f(X^n)$ if we want to return a `Factorization` object. If we printed the irreducible factors directly, we could reduce the used memory even further.

The bottleneck of our algorithm is the conversion of the irreducible factors over $\mathbb{F}_{q^{ks}}$ to $\mathbb{F}_q$, where k is the degree of f and s either $\mathrm{ord}_{\mathrm{rad}(n)}(q^k)$ or $2 \cdot \mathrm{ord}_{\mathrm{rad}(n)}(q^k)$. The conversion function `to_basef` defined in our class `RichExtensionField` is a redirection to SageMath's `RelativeFiniteFieldExtension`. Eventually the conversion could be speeded up by a better implementation of `sage.coding.relative_finite_field_extension`. At the moment the module chooses a random homomorphism between the two finite fields instead of constructing the extension field with an irreducible polynomial over the base field. We believe that such an implementation could improve the performance of the module. If q is prime, then the conversion of the irreducible factors over $\mathbb{F}_{q^{ks}}$ to $\mathbb{F}_q$ could be omitted, because the coefficients are displayed in $\mathbb{F}_{q^{ks}}$ exactly as in $\mathbb{F}_q$. However, this simplification is currently not implemented in our function `factor_fXn` because we return a `Factorization` instance over the finite field $\mathbb{F}_q$. Otherwise the function would technically return a `Factorization` instance over $\mathbb{F}_{q^{ks}}$ even though this would not be visible in the printed output.

Thus, for factorizations with few monic irreducible factors or factors of low weight, our implementation performs better than when many coefficients need to be cast to the base field. If $s = 1$ and f is a binomial of the form $X - a$ for an element $a \in \mathbb{F}_q^*$, then no conversion is necessary and our algorithm performs best. In general, when s is small the performance is better than when s is large because computations in extension fields are costly (also for SageMath and Magma). Furthermore, if s is large, then the computation of the q-spin takes longer than when s is small. The following table illustrates this behavior of our implementation.

Let $q = 16$ and $\mathbb{F}_{16} = \mathbb{F}_2(\gamma)$, where γ is a root of the monic irreducible polynomial $X^4 + X + 1 \in \mathbb{F}_2[X]$. We compute the factorization of the binomial $X^n - \gamma^3 \in \mathbb{F}_{16}[X]$ for positive integers n, where $\mathrm{ord}(\gamma^3) = 5$.

Table 2.12: CPU times for $X - \gamma^3 \in \mathbb{F}_{16}[X]$

	n	s	SageMath	Magma	New Alg	SM:NA	MA:NA
3.1	$1\,331 = 11^3$	5	137.708 s	0.010 s	0.086 s	1607:1	1:8
3.2	$1\,321$	15	118.352 s	0.190 s	3.884 s	30:1	1:20
3.3	$4\,513$	47	∞ s	3.720 s	27.401 s	∞:1	1:7
3.4	$4\,177$	87	∞ s	2.090 s	59.457 s	∞:1	1:28
3.5	$3\,375 = 3^3 \cdot 5^3$	1	∞ s	0.050 s	0.044 s	∞ :1	1:1
3.6	$50\,625 = 3^4 \cdot 5^4$	1	∞ s	153.410 s	0.462 s	∞:1	332:1
3.7	$759\,375 = 3^5 \cdot 5^5$	1	∞ s	∞ s	6.590 s	∞:1	∞:1
3.8	$11\,390\,625 = 3^6 \cdot 5^6$	1	∞ s	∞ s	97.549 s	∞:1	∞:1
3.9	$170\,859\,375 = 3^7 \cdot 5^7$	1	∞ s	`outofmem`	1470.325 s	:1	∞:1

Rows 3.1 to 3.4 show that for equally large positive integers n the CPU time grows significantly if s grows. Rows 3.5 to 3.9 show that our algorithm performs very well for positive integers n such that $s = 1$, which is equivalent to $\mathrm{rad}(n) \mid (q-1)$ and ($4 \nmid n$ or $q \equiv 1 \pmod 4$). Even for relatively small positive integers such as $n = 3\,375 = 3^3 \cdot 5^3$, it is faster than Magma. For $n = 3^7 \cdot 5^7$ (3.9) we raised the maximum computation time to 1500 seconds. Even though the computation time grows, our algorithm returns a result eventually (after 1470 seconds) because it does not use much memory. Thus, in general our algorithm is much more stable than SageMath's or Magma's for large positive integers n.

We believe that an implementation of our algorithm in C would be a valuable addition to any computer algebra system for the factorization of polynomials of the form $f(X^n)$ for a polynomial $f \in \mathbb{F}_q[X]$, especially for large positive integers n. If f is not irreducible, Algorithm 2 can be applied for every monic irreducible factor of f separately.

2.10 Future Work

In Section 2.5.1 and Section 2.3.3 we determined the generating polynomials of all cyclic and all constacyclic codes of length n for every positive integer n. For applications it would be of interest to study their code parameters. In particular, determining the minimum distance of a constacyclic code is difficult. For some cyclic and constacyclic codes the minimum distance has been determined in [AP99; BR03; Din08; ÖÖ09; Lóp+13; LYL14; CLZ15; Liu+16; LC17; LYL15; Din+19; Din+21; DWS20; GYL21] but no general result exists.

Let $C \subseteq \mathbb{F}_q^n$ be a code of length n with minimum distance d, then its dual $C^\perp$ is defined as

$$C^\perp = \{v \in \mathbb{F}_q^n : v^T \cdot c = 0 \text{ for all } c \in C\}.$$

Among the cyclic and constacyclic codes defined by Corollary 2.48 and Corollary 2.45, we could identify special types of codes such as

- linear complementary-dual (LCD) codes (satisfying $C \cap C^\perp = \{0\}$),
- self-dual codes (satisfying $C^\perp = C$),
- dual containing codes (satisfying $C^\perp \subseteq C$),
- self-orthogonal codes (satisfying $C \subseteq C^\perp$),

- MDS (maximum distance separable) codes (satisfying $|C| = q^{n-d+1}$).

Identifying all MDS constacyclic codes is especially relevant for practical applications because they can be used to construct quantum MDS codes. Partial results for all of the types mentioned above exist but it would be of interest to completely describe all constacyclic codes satisfying these properties.

Chapter 3

Constructing irreducible polynomials recursively with a reverse composition method

The irreducible polynomials obtained with the composition method are usually of higher degree than the initial polynomial. However, if a given polynomial does not have good cryptographic or arithmetic properties, it is of interest to construct a large number of irreducible polynomials of the same degree from which better candidates can be selected.

In this chapter we present a recursive construction of monic irreducible polynomials which reverses the classical composition method for the composition $f(X^n)$. More precisely, we construct the minimal polynomial $m_{\beta^n,q}$ from the monic irreducible polynomial $f = m_{\beta,q} \in \mathbb{F}_q[X]$, where β is a root of f and n is a positive integer such that $\mathrm{rad}(n) \mid q-1$. Since our construction can be applied for every positive integer n satisfying $\mathrm{rad}(n) \mid q-1$, it yields a large number of monic irreducible polynomials of the same or a lower degree than $\deg(f)$.

Most of the content of this chapter has been published in the article *Constructing irreducible polynomials recursively with a reverse composition method* - [GK23] - which is a joint work with Gohar M. Kyureghyan.

We only consider minimal polynomials over a fixed finite field $\mathbb{F}_q$ in this chapter and therefore use the simplified notation m_β instead of $m_{\beta,q}$ and χ_β instead of $\chi_{\beta,q}$.

3.1 Known constructions of m_{β^n} from m_β

In this section we present two known constructions of the minimal polynomial m_{β^n} from the monic irreducible polynomial $f = m_\beta \in \mathbb{F}_q[X]$ of degree k, where n is a positive integer and $\beta \in \mathbb{F}_{q^k}$ is a root of f. In their first step both constructions compute the composition $\chi_{\beta^n}(X^n)$, where χ_{β^n} is the characteristic polynomial of β^n in $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$. Then in Step 2 the polynomial χ_{β^n} is extracted from the composition $\chi_{\beta^n}(X^n)$, which is a reversal of the classical composition method.

The first construction, Construction AD in this thesis, has been presented in [Alb56] for primitive polynomials and generalized in [Day65]. It is defined for every positive integer n. In its next steps, Step 3 and 4, either the order of β needs to be known or the polynomial χ_{β^n} needs to be factored in order to derive m_{β^n} from χ_{β^n}. Both operations are no simple computations.

The second construction from [KK22], Construction KK in this thesis, is applicable only for $n = 2$ in finite fields of odd characteristic. The authors develop a simple condition on

the coefficients of f which holds if and only if $\chi_{\beta^2} = m_{\beta^2}$. They apply the construction only if this condition is satisfied.

Recursive application of both constructions shows a periodic behavior and in [KK22] the authors explain how this behavior can be used to obtain information on the order of β if it was previously unknown.

We combine the ideas from [Alb56], [Day65] and [KK22] to develop a construction of m_{β^n} from $f = m_\beta$ for every positive integer n such that $\mathrm{rad}(n) \mid (q-1)$. We generalize the condition on the coefficients of f from Construction KK so that the construction does not rely on the order of β or the factorization of χ_{β^n}. Furthermore, our main result Theorem 3.21 allows us to compute the composition $m_{\beta^n}(X^n)$ instead of $\chi_{\beta^n}(X^n)$ in Step 1 even if the generalized condition does not hold for f. Therefore, m_{β^n} does not need to be derived from χ_{β^n}. Furthermore, we correct and generalize the method to obtain information on the order of β presented in [KK22]. Our construction can be used in finite fields of even characteristic which is important for applications in cryptography and coding theory.

We present the results from [Alb56; Day65; KK22] with a unified notation and terminology so that the similarities of the approaches become visible. Construction AD is based on the two following theorems, which were originally stated for primitive polynomials in [Alb56, pp.142-145] but have been generalized in [Day65].

Theorem 3.1 ([Alb56; Day65])**.** *Let $f \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k and $\beta \in \mathbb{F}_{q^k}$ a root of f. Let $n \in \mathbb{N}$ such that $n = \hat{n} \cdot \mathrm{char}(\mathbb{F}_q)^l$ for $l \geq 0$ and $\gcd(\hat{n}, q) = 1$. Then the characteristic polynomial $\chi_{\beta^n} \in \mathbb{F}_q[X]$ of $\beta^n \in \mathbb{F}_{q^k}$ over $\mathbb{F}_q$ satisfies*

$$\chi_{\beta^n}(X^n) = (-1)^{k(n+1)} \prod_{j=1}^{n} f(\zeta_{\hat{n}}^j X).$$

Given a monic irreducible polynomial $f = m_\beta$ over $\mathbb{F}_q$ of degree k, Theorem 3.1 allows us to build the composition $\chi_{\beta^n}(X^n)$ by computing the following product in $\mathbb{F}_q(\zeta_{\hat{n}})$:

$$(-1)^{k(n+1)} \prod_{j=1}^{n} f(\zeta_{\hat{n}}^j X).$$

From the composition $\chi_{\beta^n}(X^n) = \sum_{i=0}^{k} a_i X^{n \cdot i}$ we *extract* the polynomial $\chi_{\beta^n} = \sum_{i=0}^{k} a_i X^i$ by dividing all the exponents of X in $\chi_{\beta^n}(X^n)$ by the positive integer n. Since χ_{β^n} is a power of the minimal polynomial m_{β^n} of $\beta^n \in \mathbb{F}_{q^k}$ over $\mathbb{F}_q$, it remains to derive m_{β^n} from χ_{β^n}. The exponent of the power of m_{β^n} in χ_{β^n} depends on the order of the initial polynomial f as the following theorem shows.

Theorem 3.2 ([Alb56; Day65])**.** *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and order e and let $\beta \in \mathbb{F}_{q^k}^*$ be a root of f. Then for every positive integer n the characteristic polynomial $\chi_{\beta^n} \in \mathbb{F}_q[X]$ of $\beta^n \in \mathbb{F}_{q^n}$ over $\mathbb{F}_q$ satisfies $\chi_{\beta^n} = (m_{\beta^n})^{\frac{k}{m}}$, where the minimal polynomial m_{β^n} of β^n over $\mathbb{F}_q$ has order $\frac{e}{\gcd(e,n)}$ and degree m, which is the smallest positive integer for which $\frac{e}{\gcd(e,n)}$ divides $q^m - 1$.*

Based on Theorems 3.1 and 3.2 Albert defines a so-called cubic transformation, which maps m_β onto m_{β^3}. If there exists a positive integer m such that $m_{\beta^{3s}} = m_\beta$, he calls the transformation *periodic* and remarks that this happens if and only if there exist positive integers s and j such that $3^s \equiv q^j \pmod{\mathrm{ord}(\beta)}$. Then s is the *period* of the transformation. He further proves that the transformation is periodic if and only if 3 does not divide $\mathrm{ord}(\beta)$.

In [Day65] Daykin proves that there exists a sequence (n_i) of positive integers $1 \leq n_i \leq \mathrm{ord}(\beta)$ such that in the sequence $(m_{\beta^{n_i}})$ every irreducible polynomial whose order divides the order of β appears exactly once. The result is presented in the next theorem.

Theorem 3.3 ([Day65, Theorem 4]). *Let β be a proper element of $\mathbb{F}_{q^k}^*$ and $\operatorname{ord}(\beta) = e$. Further, let $(n_i)_{i\in\mathcal{I}}$ be a finite sequence of positive integers $n_i \leq e$ for $i \in \mathcal{I} \subsetneq \mathbb{N}$ defined as follows. Set $n_1 = 1$ and $n_i \leq e$ to be the least positive integer such that for all $1 \leq j < i$ and all $0 \leq l < \deg(m_{\beta^{n_i}})$ holds $n_i \not\equiv n_j \cdot q^l \pmod{e}$.*

Then each irreducible polynomial over $\mathbb{F}_q$ whose order divides e appears exactly once in the sequence $(m_{\beta^{n_i}})_{i\in\mathcal{I}}$.

We summarize the ideas from [Alb56; Day65] in the following construction:

Construction AD. *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and order e and let $\beta \in \mathbb{F}_{q^k}^*$ be a root of f. Given a positive integer $n \leq e$ such that $n = \hat{n} \cdot \operatorname{char}(\mathbb{F}_q)^l$ for $l \geq 0$. To construct the minimal polynomial m_{β^n} of $\beta^n \in \mathbb{F}_{q^n}$ over $\mathbb{F}_q$, do the following steps:*

Step 1. Compute the product

$$(-1)^{k(n+1)} \prod_{j=1}^{n} f\left(\zeta_{\hat{n}}^j X\right) = \chi_{\beta^n}\left(X^n\right).$$

Step 2. Extract χ_{β^n} from the composition $\chi_{\beta^n}\left(X^n\right)$.

Step 3. Determine m, the least positive integer for which $\frac{e}{\gcd(e,n)}$ divides $q^m - 1$.

Step 4. Derive the factor m_{β^n} from the product $\chi_{\beta^n} = (m_{\beta^n})^{\frac{k}{m}}$.

Remark 3.4. (a) In [Alb56] the author states that *"the main advantage of the cubing transformation is that it is a relatively simple matter to compute m_{β^3} when m_β is given"*. However, the $\hat{n}$-th root of unity $\zeta_{\hat{n}}$, which is needed for the computation in Step 1., is an element of $\mathbb{F}_q$ if and only if $\hat{n} \mid q-1$. Thus, if $\hat{n}$ does not divide $q-1$ the computation is carried out in a pure extension field of $\mathbb{F}_q$. Computations in extension fields are difficult to implement and take up more resources than computations in a fixed finite field $\mathbb{F}_q$.

(b) Construction AD can also be applied without the knowledge of the order e of the polynomial f. In that case we replace Step 3. and Step 4. with factoring χ_{β^n}, which will be an unknown power of the minimal polynomial m_{β^n} of β^n over $\mathbb{F}_q$.

(c) On the other hand, if the order e of f is known, it is possible to avoid the computation intensive Step 4. by selecting n such that $k = m$. Then the characteristic and the minimal polynomial of β^n over $\mathbb{F}_q$ are equal.

The following result gives all the information needed to formulate Construction KK.

Theorem 3.5 ([KK22, Corollary 3]). *Let q be odd and $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and order e. Let $\beta \in \mathbb{F}_{q^k}$ be a root of f. Then the following statements hold:*

(i) There exists a polynomial $C \in \mathbb{F}_q[X]$ such that $C\left(X^2\right) = f(X) \cdot (-1)^k f(-X)$. More precisely, $C(X) = (-1)^k \sum_{j=0}^{k} \sum_{u=0}^{2j} (-1)^u c_u c_{2j-u} X^j$, where $c_0, \ldots, c_k$ are the coefficients of f and $c_u = 0$ for $u > k$.

(ii) If C is irreducible, it is the minimal polynomial of β^2 over $\mathbb{F}_q$ and $\operatorname{ord}(C) = \frac{e}{\gcd(e,2)}$.

(iii) The polynomial C is irreducible if and only if there does not exist a polynomial $D \in \mathbb{F}_q[X]$ such that $f(X) = D\left(X^2\right)$.

Theorem 3.5 (i) and (ii) follow directly from Theorems 3.1 and 3.2, but the theorem can also be proved by elementary means. The polynomial C is in fact the characteristic polynomial of $\beta^2 \in \mathbb{F}_{q^n}$ over $\mathbb{F}_q$, which means that C is irreducible if and only if it is the minimal polynomial of β^2 over $\mathbb{F}_q$. Theorem 3.5 (iii) allows us to determine whether C equals m_{β^2} or not by a simple examination of the coefficients of the polynomial f. This leads to the following construction, which is the key step of constructions [KK22, Construction 1] and [KK22, Construction 2].

Construction KK ([KK22])**.** *Let q be odd and $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k such that there does not exist a polynomial $D \in \mathbb{F}_q[X]$ with $f(X) = D\left(X^2\right)$. To construct the monic irreducible polynomial $C \in \mathbb{F}_q[X]$ of degree k over $\mathbb{F}_q$, do the following steps:*

Step 1. Compute the product $(-1)^k \cdot f(X) \cdot f(-X) = C\left(X^2\right)$.

Step 2. Extract C from the composition $C\left(X^2\right)$.

Numerical results obtained from a recursive application of Construction KK are presented in [KK22, Example 2]. These computations were done with the SageMath program B.13 `Ch3-ConstructionKK.sage` which we present in Appendix B.5.

A recursive application of Construction KK shows the same periodic behavior that Albert describes for his cubing transformation. In [KK22, Construction 2, Remark 1] the authors explain how this behavior can be used to gain information on the order of the initial polynomial if it is not known. Suppose that the initial polynomial $f = m_\beta$ for $\beta \in \mathbb{F}_{q^k}$ satisfies $\operatorname{ord}(f) = 2^t \cdot r$, with $\gcd(2, r) = 1$. Then the construction first yields $t - 1$ polynomials of strictly decreasing even order. The following sequence of constructed polynomials begins with $m_{\beta^{2^t}}$. After a finite number of applications this polynomial is the first to appear twice. Then the length of the sequence of polynomials constructed, starting from $m_{\beta^{2^t}}$ up to the second appearance of $m_{\beta^{2^t}}$, is a divisor of $\operatorname{ord}_2(r)$.

Remark 3.6. In the first version of [KK22] the authors stated that this length equals $\operatorname{ord}_2(r)$ by overseeing that $\beta^{2^{t+s}}$ might be an $\mathbb{F}_q$-conjugate of β^{2^t} for a positive integer $s < \operatorname{ord}_2(r)$. Then $m_{\beta^{t+s}}$ equals m_{β^t} and the length of the sequence of polynomials constructed is s and not $\operatorname{ord}_2(r)$.

In Section 3.3 we discuss the generalization of these ideas in more detail.

3.2 New results on χ_{β^n} and m_{β^n}

In this section we show how for a proper element β of $\mathbb{F}_{q^k}$ and any divisor n of $q-1$ the composition $m_{\beta^n}(X^n)$ can be computed directly from m_β. For this we combine and extend the ideas from [Alb56; Day65] and [KK22]. This section is basically the proof of our main result in this chapter, Theorem 3.21, which states that for a divisor n of $q-1$ holds

$$m_{\beta^n}(X^n) = \prod_{j=1}^{\frac{n}{t}} \zeta_n^{-jk} f(\zeta_n^j X),$$

where $t = \max\{m \mid \gcd(n,k) : f(X) = g(X^m) \text{ for a polynomial } g \in \mathbb{F}_q[X]\}$. However, most results in this section hold not only for divisors n of $q-1$ but for arbitrary positive integers n.

Remark 3.7. It suffices to consider only the positive integers n such that $\gcd(q, n) = 1$. Indeed, if $\gcd(q, n) > 1$ we can write $n = p^l \cdot \hat{n}$, where $\gcd(q, \hat{n}) = 1$ and $p = \operatorname{char}(\mathbb{F}_q)$, and the coefficients of m_{β^n} can easily be derived from the coefficients of $m_{\beta^{\hat{n}}}$. Indeed, let $e = \operatorname{ord}(\beta)$, then Theorem 3.2 implies that

$$\operatorname{ord}(m_{\beta^n}) = \frac{e}{\gcd(e, n)} = \frac{e}{\gcd(e, \hat{n})} = \operatorname{ord}(m_{\beta^{\hat{n}}})$$

and therefore $\deg(m_{\beta^{\hat{n}}}) = \deg(m_{\beta^n}) = m$. Suppose that $m_{\beta^{\hat{n}}} = \sum_{i=0}^{m} a_i X^i$ and set $g = \sum_{i=0}^{m} a_i^{p^l} X^i \in \mathbb{F}_q[X]$. Then

$$g\left(\beta^n\right) = \sum_{i=0}^{m} a_i^{p^l} \left(\beta^n\right)^i = \sum_{i=0}^{m} a_i^{p^l} \left(\beta^{\hat{n}}\right)^{i \cdot p^l} = \left(m_{\beta^{\hat{n}}}\left(\beta^{\hat{n}}\right)\right)^{p^l} = 0.$$

Thus, β^n is a root of g and since $\deg(g) = m = \deg(m_{\beta^n})$ the polynomial g is the minimal polynomial of β^n over $\mathbb{F}_q$. That is, $m_{\beta^n} = \sum_{i=0}^{m} a_i^{p^l} X^i$.

Example 3.8. Let $\mathbb{F}_4 = \mathbb{F}_2(a)$, where a is a root of the polynomial $X^2 + X + 1 \in \mathbb{F}_2[X]$. In the next three examples we consider the polynomial $f = X^2 + aX + a$ over the finite field $\mathbb{F}_4$. f is irreducible over $\mathbb{F}_4$ and the minimal polynomial of $\beta = b^3 + b + 1$ over $\mathbb{F}_4$. β is a proper element of the finite field $\mathbb{F}_{16} = \mathbb{F}_2(b)$, where b is a root of the polynomial $X^4 + X + 1 \in \mathbb{F}_2[X]$.

For $n = 10 = 2 \cdot 5$ the element β^{10} is $b^2 + b + 1$ and its minimal polynomial over $\mathbb{F}_4$ is $m_{\beta^{10}} = X + a + 1$. Then $\hat{n} = 5$ and the minimal polynomial of $\beta^5 = b^2 + b$ over $\mathbb{F}_4$ is $m_{\beta^5} = X + a$. With the fact that $a^2 = a + 1$, we easily see that the coefficients of $m_{\beta^{10}}$ can indeed be derived from the coefficients of m_{β^5} as described in Remark 3.7.

The computations for all examples in Chapter 3 (3.8, 3.10, 3.12, 3.14, 3.17, 3.20, 3.22, 3.24, 3.28, 3.29) were done with the SageMath script B.11 `Ch3-Example.sage`. The user can change the variable `examples` and include only the numbers of the examples she or he wishes to see (look for `USER INPUT` in the code).

We state all of our results only for positive integers n satisfying $\gcd(q, n) = 1$. Nonetheless, note that all results also hold if $\gcd(q, n) > 1$. A big advantage of considering only positive integers n such that $\gcd(q, n) = 1$ is that there always exist exactly n distinct n-th roots of unity in an extension field $\mathbb{E} \geq \mathbb{F}_q$ of $\mathbb{F}_q$.

The next theorem shows that there exists a direct connection between the exponent t of the minimal polynomial m_{β^n} in $\chi_{\beta^n} = (m_{\beta^n})^t$ and the multiplicities of the roots of the polynomial $\chi_{\beta^n}(X^n)$. Note that the main statement of the theorem is that the multiplicities of the roots of $\chi_{\beta^n}(X^n)$ equal those of the roots of χ_{β^n}.

Theorem 3.9 ([GK23, Theorem 4])**.** *Let $n \in \mathbb{N}$ with $\gcd(q, n) = 1$. Further, let $\beta \in \mathbb{F}_{q^k}$ be a proper element of $\mathbb{F}_{q^k}$ and χ_{β^n} be the characteristic polynomial of $\beta^n \in \mathbb{F}_{q^n}$ over $\mathbb{F}_q$. Then $\chi_{\beta^n} = (m_{\beta^n})^t$ for a positive integer $t \in \mathbb{N}$ if and only if every root of the polynomial $\chi_{\beta^n}(X^n)$ has multiplicity t. That is, the roots of $\chi_{\beta^n}(X)$ and the roots of $\chi_{\beta^n}(X^n)$ have the same multiplicity t.*

Proof. Since χ_{β^n} is the characteristic polynomial of β^n over $\mathbb{F}_q$, there exists a positive integer $t \geq 1$ such that $\chi_{\beta^n}(X^n) = (m_{\beta^n}(X^n))^t$. Furthermore,

$$m_{\beta^n}(X^n) = \prod_{i=0}^{\frac{k}{t}-1} \left(X^n - \beta^{n \cdot q^i}\right)$$

and for every i the polynomial $X^n - \left(\beta^{q^i}\right)^n$ has n distinct roots of the form $\zeta_n^j \beta^{q^i}$ in an extension field of $\mathbb{F}_q$, where $1 \leq j \leq n$. Thus, $\chi_{\beta^n}(X^n) = \prod_{i=0}^{\frac{k}{t}-1} \prod_{j=1}^{n} \left(X - \zeta_n^j \beta^{q^i}\right)^t$. Note that the roots $\zeta_n^j \beta^{q^i}$ of $\chi_{\beta^n}(X^n)$ for $1 \leq j \leq n$ and $0 \leq i \leq \frac{k}{k}t - 1$ are distinct. Indeed, if for $1 \leq j_1, j_2 \leq n$ and $0 \leq i_1, i_2 \leq \frac{k}{t} - 1$ the two roots $\zeta_n^{j_1} \beta^{q^{i_1}}$ and $\zeta_n^{j_2} \beta^{q^{i_2}}$ were equal, we would have $(\beta^n)^{q^{i_1}} = \left(\zeta_n^{j_1} \beta^{q^{i_1}}\right)^n = \left(\zeta_n^{j_2} \beta^{q^{i_2}}\right)^n = (\beta^n)^{q^{i_2}}$ and since the elements $(\beta^n)^{q^i}$ of $\mathbb{F}_{q^{\frac{k}{t}}}$ are distinct, we have $i_1 = i_2$ and consequently also $j_1 = j_2$. To complete the proof, recall that the roots of irreducible polynomials over finite fields are simple. □

Example 3.10. Let $n = 5$ and we consider again the polynomial $f = X^2 + aX + a$ over $\mathbb{F}_4 = \mathbb{F}_2(a)$. Then $\beta^5 = b^2 + b = a$ is an element of $\mathbb{F}_4$ and $m_{\beta^5} = X + a$.

We need to factor the polynomials χ_{β^5} and $\chi_{\beta^5}(X^5)$ in their respective splitting fields to determine the multiplicity of their roots. The splitting field of χ_{β^5} is always a subfield of $\mathbb{F}_{q^k}$. Here, the characteristic polynomial of β^5 in $\mathbb{F}_{16}$ over $\mathbb{F}_4$ satisfies $\chi_{\beta^5} = (m_{\beta^5})^2 = (X + a)^2$ and the multiplicity of the root a is obviously equal to the exponent of m_{β^5} in χ_{β^5}.

The splitting field of $\chi_{\beta^5}(X^5)$ is $\mathbb{F}_{16}$. This is convenient for us since we do not need to introduce another finite field. In general this is not true and the splitting field of $\chi_{\beta^n}(X^n)$ can be a proper extension of $\mathbb{F}_{q^k}$. We denote the polynomial ring over $\mathbb{F}_{16}$ by $\mathbb{F}_{16}[Y]$. Then the polynomial $\chi_{\beta^5}(X^5)$ splits into linear factors as follows:

$$\begin{aligned}\chi_{\beta^5}(X^5) =&(Y + b)^2 \cdot (Y + b + 1)^2 \cdot (Y + b^2 + b + 1)^2 \cdot (Y + b^3 + b + 1)^2 \\ &\cdot (Y + b^3 + b^2 + 1)^2\end{aligned}$$

As Theorem 3.9 states, all roots of the polynomial $\chi_{\beta^5}(X^5)$ have multiplicity 2 as well. Except for one root, $b^2 + b + 1 = a + 1$, all roots are proper elements of $\mathbb{F}_{16} = \mathbb{F}_2(b)$.

In general the roots of $\chi_{\beta^n}(X^n)$ are not elements of $\mathbb{F}_q$ and the check for their multiplicity would require a factorization in an extension field of $\mathbb{F}_{q^k}$. Since we want to remain in $\mathbb{F}_q$, we translate the property of Theorem 3.9 to the irreducible factors of $\chi_{\beta^n}(X^n)$ in $\mathbb{F}_q[X]$.

Corollary 3.11 ([GK23, Corollary 5])**.** *Let $n \in \mathbb{N}$ such that $\gcd(q, n) = 1$. Further, let $\beta \in \mathbb{F}_{q^k}$ be a proper element of $\mathbb{F}_{q^k}$ and χ_{β^n} be the characteristic polynomial of $\beta^n \in \mathbb{F}_{q^k}$ over $\mathbb{F}_q$.*
Then $\chi_{\beta^n} = (m_{\beta^n})^t$ for a positive integer t if and only if every irreducible factor of $\chi_{\beta^n}(X^n)$ over $\mathbb{F}_q$ appears with multiplicity t.

Example 3.12. Let $n = 5$ and $f = X^2 + aX + a \in \mathbb{F}_4[X]$ and $\beta = b^3 + b + 1 \in \mathbb{F}_{16}$ as in Example 3.10. In this example we use Theorem 3.1 to obtain $\chi_{\beta^5}(X^5)$ and the multiplicity of its irreducible factors over $\mathbb{F}_4$. The formula given in Theorem 3.1 yields:

$$\chi_{\beta^5}(X^5) = (-1)^{2(5+1)} \prod_{j=1}^{5} f(\zeta_5 X). \tag{3.1}$$

For the computation we need to determine a 5-th root of unity over $\mathbb{F}_4$. Since 5 does not divide $4 - 1 = 3$, ζ_5 is not an element of $\mathbb{F}_4$. However, 5 is a divisor of $16 - 1 = 15 = 3 \cdot 5$ and $\zeta_5 = b^3$. This is convenient for us since we do not need to introduce another finite field. In general $\mathbb{F}_q(\zeta_n)$ is not equal to $\mathbb{F}_{q^k}$.

Thus, (3.1) needs to be computed in $\mathbb{F}_{16}$ even if we are only interested in the irreducible factors over $\mathbb{F}_4$. The polynomial f cast into $\mathbb{F}_{16}[Y]$ is:

$$f = X^2 + aX + a = Y^2 + (b^2 + b)Y + b^2 + b$$

For (3.1) we need to compute $f(\zeta_5^j X)$ for $1 \leq j \leq 5$:

$$\begin{aligned}f(\zeta_5 Y) &= (b^3 + b^2)Y^2 + (b^2 + 1)Y + b^2 + b \\ &= (b^3 + b^2) \cdot (Y + b + 1) \cdot (Y + b^2 + b + 1) \\ f(\zeta_5^2 Y) &= (b^3 + b^2 + b + 1)Y^2 + (b^3 + b^2 + b)Y + b^2 + b \\ &= (b^3 + b^2 + b + 1) \cdot (Y + b) \cdot (Y + b^3 + b + 1) \\ f(\zeta_5^3 Y) &= b^3 Y^2 + (b^3 + 1)Y + b^2 + b \\ &= (b^3) \cdot (Y + b + 1) \cdot (Y + b^3 + b^2 + 1) \\ f(\zeta_5^4 Y) &= (b^3 + b)Y^2 + b^2 Y + b^2 + b\end{aligned}$$

$$= (b^3+b)\cdot(Y+b)\cdot(Y+b^2+b+1)$$
$$f(\zeta_5^5 Y) = f(Y).$$

Except for $f(Y)$, the polynomials $f(\zeta_5^j Y)$, $1 \le j \le 4$, are not polynomials over $\mathbb{F}_4$. Thus Theorem 3.1 does not give the factorization of $\chi_{\beta^n}(X^n)$ into irreducible factors over $\mathbb{F}_q$.

Computing the product of the polynomials above yields: $\chi_{\beta^5}(Y^5) = Y^{10}+b^2+b+1$. We know that $\chi_{\beta^5}(Y^5)$ is a polynomial over $\mathbb{F}_4$ and it remains to cast the coefficients to the field $\mathbb{F}_4$: $\chi_{\beta^5}(X^5) = X^{10}+a+1$. The factorization of $\chi_{\beta^5}(X^5)$ into irreducible factors over $\mathbb{F}_{16}$ is

$$X^{10}+a+1 = (X+a+1)^2\cdot(X^2+X+a)^2\cdot(X^2+aX+a)^2$$

As Theorem 3.9 states, the multiplicity of the irreducible factors of $\chi_{\beta^5}(X^5)$ over $\mathbb{F}_4$ is the same as the multiplicity of any root of $\chi_{\beta^5}(X^5)$.

As we have seen in the last example, in general Theorem 3.1 does not yield the factorization of $\chi_{\beta^n}(X^n)$ into irreducible factors over $\mathbb{F}_q$. The next three theorems examine under which conditions this does happen. These three results, Theorem 3.13, Theorem 3.15 and Corollary 3.16 are new and do not appear in [GK23].

From the proof of Theorem 3.1 follows directly that $(-1)^{k(n+1)} = \prod_{j=1}^{n} \zeta_n^{-j}$. Thus, we also can compute the composition $\chi_{\beta^n}(X^n)$ in the following way:

$$\chi_{\beta^n}(X^n) = \prod_{j=1}^{n} \zeta_n^{-jk} f(\zeta_n^j X) \tag{3.2}$$

We consider this variation of Theorem 3.1, because the polynomials $\zeta_n^{-jk} f(\zeta_n^j X)$ are monic. The next theorem shows that the coefficients of these polynomials are elements of $\mathbb{F}_q$ if and only if the polynomial f is a composition with X^m for a positive integer m such that $\frac{n}{m} \mid q-1$. Note that m is not necessarily the largest positive integer d satisfying $f = g(X^d)$ for a polynomial $g \in \mathbb{F}_q[X]$.

Theorem 3.13. *Let $f = \sum_{i=0}^{k} a_i X^i \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k and $\beta \in \mathbb{F}_{q^k}$ a root of f. Let $n \in \mathbb{N}$ such that $\gcd(q,n) = 1$. Then the factorization of the polynomial $\chi_{\beta^n}(X^n)$ into monic (not necessarily irreducible) factors over $\mathbb{F}_q$ is given by $\prod_{j=1}^{n} \zeta_n^{-jk}\ f(\zeta_n^j X)$ if and only if there exists a positive integer m such that $\frac{n}{m} \mid (q-1)$ and a polynomial $g \in \mathbb{F}_q[X]$ such that $f = g(X^m)$.*

Proof. "⇒": Suppose that $\zeta_n^{-jk} f(\zeta_n^j X)$ is a polynomial over $\mathbb{F}_q$ for all $1 \le j \le n$. We define $\mathcal{I} := \{0 \le i \le k : a_i \neq 0\}$. Then $0, k \in \mathcal{I}$ because f is an irreducible polynomial of degree k. We can write

$$\zeta_n^{-jk}\ f(\zeta_n^j X) = \sum_{i\in\mathcal{I}} a_i\ \zeta_n^{-kj}\ \zeta_n^{ij}\ X^i.$$

We select $j = 1$ and $i = 0 \in \mathcal{I}$. Then $a_i\ \zeta_n^{-kj}\ \zeta_n^{ij} = a_0 \cdot \zeta_n^{-k} \in \mathbb{F}_q$ and since a_0 is an element of $\mathbb{F}_q$, also ζ_n^{-k} has to be an element of $\mathbb{F}_q$. This is equivalent to $\zeta_n^k \in \mathbb{F}_q$. Thus, $\zeta_n^{-kj} \in \mathbb{F}_q$ for all $1 \le j \le n$ and $a_i\ \zeta_n^{-jk}\ \zeta_n^{ij}$ is an element of $\mathbb{F}_q$ if and only if $\zeta_n^{ij} \in \mathbb{F}_q$. With $j = 1$ follows that $\zeta_n^i \in \mathbb{F}_q$ for all $i \in \mathcal{I}$. Let $d = \gcd(\mathcal{I})$ and $i_1 \neq i_2 \in \mathcal{I}$. Then $d = l\cdot\gcd(i_1,i_2)$ for a positive integer l and there exist $q_1, q_2 \in \mathbb{Z}$ such that $d = q_1 i_1 + q_2 i_2$. Thus, $\zeta_n^d = \zeta_n^{i_1 q_1 + i_2 q_2} = (\zeta_n^{i_1})^{q_1}\cdot(\zeta_n^{i_2})^{q_2} \in \mathbb{F}_q$ and $\frac{n}{\gcd(n,d)} \mid q-1$. Set $m := \gcd(n,d)$, then $m \mid n$ and $\frac{n}{m} \mid q-1$. Furthermore, since $d = \gcd(\mathcal{I})$, there exists a polynomial $h \in \mathbb{F}_q[X]$ such that $f = h(X^d) = h((X^m)^{\frac{d}{m}})$. Set $g = h(X^{\frac{d}{m}})$, then $f = g(X^m)$.

"⇐": If $f(X) = g(X^m)$ for a positive integer such that $\frac{n}{m} \mid q-1$, then $\zeta_n^{-jk} f(\zeta_n^j X) = \zeta_n^{-m\frac{k}{m}j} g(\zeta_n^{mj} X^m)$, where $\zeta_n^m = \zeta_{\frac{n}{m}}$ is an $\frac{n}{m}$-th primitive root of unity. Thus, $\zeta_n^{-jk} f(\zeta_n^j X) = \zeta_{\frac{n}{m}}^{-\frac{k}{m}j} g(\zeta_{\frac{n}{m}}^j X^m)$, which is a monic polynomial over $\mathbb{F}_q$. □

In the following results the irreducible polynomials of the form $f = g(X^m)$ for $g \in \mathbb{F}_q[X]$ and $m \in \mathbb{N}$ play an important role. The interested reader might check the statements of the results herself or himself and needs such a polynomial to begin with. For this, the SageMath-script B.6 `CompositionGenerator.sage` is useful. It computes all irreducible compositions of the form $f = g(X^m) \in \mathbb{F}_q[X]$, where $g \in \mathbb{F}_q[X]$, of degree `mindeg` $\leq \deg(f) \leq$ `maxdeg` for a fixed positive integer m over a fixed finite field of size q. Look for `USER INPUT` to change the parameters `q`, `m`, `mindeg` and `maxdeg`.

Example 3.14. Again let $\mathbb{F}_4 = \mathbb{F}_2(a)$, where a is a root of the polynomial $X^2+X+1 \in \mathbb{F}_2[X]$. In this example we consider the monic irreducible polynomial

$$f = X^6 + X^3 + a + 1 \in \mathbb{F}_4[X]$$

It satisfies $f = g(X^3)$ for the monic irreducible polynomial $g = X^2 + X + a + 1 \in \mathbb{F}_4[X]$. Its root β is an element of $\mathbb{F}_{4096}$.

Let $n = 9$ and $m = 3$, then $\frac{n}{m} = 3$ divides $q - 1$. Since $\mathbb{F}_4(\zeta_9) = \mathbb{F}_{64}$, we usually would compute formula (3.2) in the extension field $\mathbb{F}_{64}$. However, from the proof of Theorem 3.13 follows that since $\zeta_9^3 = \zeta_3$, we can compute $\zeta_3^{-2j} g(\zeta_3^j X^3)$ instead of $\zeta_9^{-6j} f(\zeta_9^j X)$. This computation happens in $\mathbb{F}_4$, so that we are able to remain in the original field $\mathbb{F}_q$.

As j runs through the integers $1, \ldots, 9$, the polynomials $\zeta_3^{-2j} g(\zeta_3^j X^3)$ and $\zeta_3^{-2(j+3)} g(\zeta_3^{j+3} X^3)$ are equal. Therefore, there are only three distinct polynomials appearing three times each:

$$\begin{aligned}
\zeta_3^{-2} g(\zeta_3 X^3) &= \zeta_3^{-8} g(\zeta_3^4 X^3) = \zeta_3^{-14} g(\zeta_3^7 X^3) \\
&= X^6 + (a+1)X^3 + 1 \\
\zeta_3^{-4} g(\zeta_3^2 X^3) &= \zeta_3^{-10} g(\zeta_3^5 X^3) = \zeta_3^{-16} g(\zeta_3^8 X^3) \\
&= X^6 + aX^3 + a \\
f(X) &= \zeta_3^{-6} g(\zeta_3^3 X^3) = \zeta_3^{-12} g(\zeta_3^6 X^3) = \zeta_3^{-18} g(\zeta_3^9 X^3) \\
&= X^6 + X^3 + a + 1
\end{aligned}$$

All three polynomials are polynomials over $\mathbb{F}_4$, but only the second and third polynomial are irreducible over $\mathbb{F}_4$. The first polynomial factors as

$$X^6 + (a+1)X^3 + 1 = (X^2 + X + a) \cdot (X^2 + aX + 1) \cdot (X^2 + (a+1)X + a + 1)$$

over $\mathbb{F}_4$. The factorization of $\chi_{\beta^9}(X^9)$ into monic irreducible factors over $\mathbb{F}_4$ is:

$$\begin{aligned}
\chi_{\beta^9}(X^9) =& (X^2 + X + a)^3 \cdot (X^2 + aX + 1)^3 \cdot (X^2 + (a+1)X + a + 1)^3 \\
&\cdot (X^6 + X^3 + a + 1)^3 \cdot (X^6 + aX^3 + a)^3
\end{aligned}$$

Thus, (3.2) does yield monic factors over $\mathbb{F}_q$ if the condition of Theorem 3.13 is satisfied. For all of them to be irreducible, we need more conditions on the polynomial f as the following theorem shows.

Note that the proof of Theorem 3.15 relies on Theorem 2.37, our closed formula for the factorization of $X^n - a$ over $\mathbb{F}_q$ for every positive integer n satisfying $\mathrm{rad}(n) \mid (q-1)$ and $(4 \nmid n$ or $q \equiv 1 \pmod 4)$. Here we do not use Theorem 2.37 to determine the factorization of a binomial of the form $X^n - a$ but to determine the order of elements of the form $\zeta_n^j \beta$ which shows that the theorem can be useful in many applications besides factorization.

Theorem 3.15. *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and $\beta \in \mathbb{F}_{q^k}^*$ a root of f. Let $n, m \in \mathbb{N}$ such that $\gcd(q, n) = 1$, $\frac{n}{m} \mid (q-1)$ and $f(X) = g(X^m)$ for a polynomial $g \in \mathbb{F}_q[X]$. Then all polynomials in $\{\zeta_n^{-jk} f(\zeta_n^j X) : 1 \leq j \leq n\}$ are irreducible if and only if there does not exist a prime p such that $p \mid m$ and $\nu_p(\mathrm{ord}(\beta)) - \nu_p(m) \leq \nu_p(\frac{n}{m})$.*

Proof. Since $f = g(X^m)$ is irreducible, g must be irreducible itself. We know that β^m is a root of g and a proper element of $\mathbb{F}_{q^{k/m}}$. Thus, from Theorem 2.2 follows that the following three conditions hold:

$$(i) \quad \mathrm{rad}(m) \mid \mathrm{ord}(\beta^m) \qquad (ii) \quad \gcd\left(m, \frac{q^{\frac{k}{m}}-1}{\mathrm{ord}(\beta^m)}\right) = 1 \qquad (3.3)$$

$$(iii) \quad 4 \nmid m \text{ or } q^{\frac{k}{m}} \equiv 1 \pmod 4.$$

The polynomial $\zeta_n^{-jk} f(\zeta_n^j X) = \zeta_{n/m}^{-j\frac{k}{m}} g(\zeta_{n/m}^j X^m)$ is irreducible if and only if $g(\zeta_{n/m}^j X^m)$ is irreducible. Suppose that γ is the root of $g(\zeta_{n/m}^j X^m)$ satisfying $\zeta_{n/m}^j \gamma^m = \beta^m$. Then γ is a root of the polynomial $X^m - \zeta_{n/m}^{-j}\beta^m$. Since $\zeta_{n/m} \in \mathbb{F}_q$ and β^m is a proper element of $\mathbb{F}_{q^{k/m}}$, this is a polynomial with coefficient degree $\frac{k}{m}$ over $\mathbb{F}_q$. The polynomial $g(\zeta_{n/m}^j X^m)$ is irreducible if and only if γ is a proper element of $\mathbb{F}_{q^k}$. This is equivalent to the polynomial $X^m - \zeta_{n/m}^{-j}\beta^m$ being irreducible. With Theorem 2.22 we need to check if the following three conditions hold:

$$(i) \quad \mathrm{rad}(m) \mid \mathrm{ord}(\zeta_{n/m}^{-j}\beta^m) \qquad (ii) \quad \gcd\left(m, \frac{q^{\frac{k}{m}}-1}{\mathrm{ord}(\zeta_{n/m}^{-j}\beta^m)}\right) = 1 \qquad (3.4)$$

$$(iii) \quad 4 \nmid m \text{ or } q^{\frac{k}{m}} \equiv 1 \pmod 4.$$

From (3.3) (iii) follows directly that condition (3.4) (iii) above is always satisfied. If $\mathrm{ord}(\beta^m) \mid \mathrm{ord}(\zeta_{n/m}^{-j}\beta^m)$, then (3.3) (i) and (ii) also are satisfied. If $\gcd(\frac{n}{m}, \mathrm{ord}(\beta^m)) = 1$, then $\mathrm{ord}(\zeta_{n/m}^{-j}\beta^m) = \mathrm{ord}(\zeta_{n/m}^{-j}) \cdot \mathrm{ord}(\beta^m)$ and conditions (3.4) (i) and (ii) hold. Consequently, for (3.4) (i) or (ii) to be violated, we have $\gcd(\frac{n}{m}, \mathrm{ord}(\beta^m)) > 1$. More precisely, condition (i) is violated if and only if there exists a prime p dividing m such that p does not divide $\mathrm{ord}(\zeta_{n/m}^{-j}\beta^m)$. Condition (3.3) (i) implies that $p \mid \mathrm{ord}(\beta^m)$ and we reformulate as follows: Condition (3.4) (i) is violated if and only if

$$\text{there exists a prime } p \text{ such that } p \mid m \text{ and } 0 = \nu_p(\mathrm{ord}(\zeta_{n/m}^{-j}\beta^m)) < \nu_p(\mathrm{ord}(\beta^m)). \quad (3.5)$$

Furthermore, condition (3.4) (ii) is violated if and only if there exists a prime p dividing m such that $\nu_p(\mathrm{ord}(\zeta_{n/m}^{-j}\beta^m)) < \nu_p(q^{\frac{k}{m}} - 1)$. Since (3.3) (ii) holds for $\mathrm{ord}(\beta^m)$, we know that $\nu_p(\mathrm{ord}(\beta^m)) = \nu_p(q^{\frac{k}{m}} - 1)$. Thus, (ii) is violated if and only if

$$\text{there exists a prime } p \text{ such that } p \mid m \text{ and } \nu_p(\mathrm{ord}(\zeta_{n/m}^{-j}\beta^m)) < \nu_p(\mathrm{ord}(\beta^m)). \quad (3.6)$$

Since (3.5) implies (3.6), we only need to check whether (3.6) holds for the order of the elements $\zeta_{n/m}^{-j}\beta^m$.

Note that $\{1 \le j \le n\} = \dot{\bigcup}_{0 \le c \le m-1}\{c \cdot \frac{n}{m}, \ldots, (c+1) \cdot \frac{n}{m}\}$ and for any $0 \le c \le (m-1)$ the elements $\zeta_{n/m}^{-j}\beta^m$ for $c \cdot \frac{n}{m} + 1 \le j \le (c+1) \cdot \frac{n}{m}$ are the roots of the polynomial $X^{\frac{n}{m}} - (\beta^m)^{\frac{n}{m}}$. Since $\frac{n}{m} \mid q-1$, the positive integer $\frac{n}{m}$ also divides $q^{\frac{k}{m}} - 1$ and $q^{\frac{2k}{m}} - 1$. We set $s = \frac{k}{m}$ if $4 \nmid \frac{n}{m}$ or $q^{\frac{k}{m}} \equiv 1 \pmod 4$ and we set $s = \frac{2k}{m}$ if $4 \mid \frac{n}{m}$ and $q^{\frac{k}{m}} \equiv 3 \pmod 4$. Then in $\mathbb{F}_{q^s}$ we can apply Theorem 2.37, which yields:

$$X^{\frac{n}{m}} - (\beta^m)^{\frac{n}{m}} = \prod_{j=1}^{\frac{n}{m}} (X - \zeta_{\frac{n}{m}}^{-j}\beta^m) = \prod_{l=0}^{d_1-1} \prod_{v \mid \frac{n_2}{d_2}} \prod_{\substack{i=0 \\ \gcd(i,v)=1}}^{d_2-1} (X^{\frac{n_1}{d_1} \cdot v} - \zeta_{d_2}^i (\zeta_{d_1}^l b)^{rv}),$$

where $\frac{n}{m} = n_1 \cdot n_2$ such that $\mathrm{rad}(n_1) \mid \mathrm{ord}\left((\beta^m)^{\frac{n}{m}}\right)$ and $\gcd(n_2, \mathrm{ord}((\beta^m)^{\frac{n}{m}})) = 1$.

If $d = \gcd(\frac{n}{m}, \mathrm{ord}(\beta^m)) > 1$, then

$$\mathrm{ord}((\beta^m)^{\frac{n}{m}}) = \frac{\mathrm{ord}(\beta^m)}{\gcd(\frac{n}{m}, \mathrm{ord}(\beta^m))} = \frac{\mathrm{ord}(\beta)}{d} < \mathrm{ord}(\beta^m).$$

Furthermore, $d_1 = \gcd(n_1, \frac{q^s-1}{\mathrm{ord}(\beta^m)/d})$ and $d_2 = \gcd(n_2, q^s - 1)$ and the element $b^r \in \mathbb{F}_{q^s}$ satisfies $(b^r)^{d_1 \cdot d_2} = (\beta^m)^{\frac{n}{m}}$. Since the maximum degree of the irreducible factors is 1, we can deduce that $n_1 = d_1$ and $n_2 = d_2$ and rewrite the factorization in the following way:

$$\prod_{j=1}^{\frac{n}{m}} (X - \zeta_{\frac{n}{m}}^{-j} \beta^m) = \prod_{l=0}^{n_1-1} \prod_{i=0}^{n_2-1} (X - \zeta_{n_2}^i (\zeta_{n_1}^l b)^r).$$

Theorem 2.37 (ii) further states that

$$\mathrm{ord}(\zeta_{\frac{n}{m}}^{-j} \beta^m) = \mathrm{ord}(\zeta_{n_2}^i (\zeta_{n_1}^l b)^r) = \frac{\mathrm{ord}(\beta^m)}{d} \cdot n_1 \cdot \frac{n_2}{\gcd(n_2, i)}$$

for integers $0 \leq i \leq n_2 - 1$ and $0 \leq l \leq n_1 - 1$. Consequently, (3.6) holds if and only if there exists a prime p such that

$$p \mid m,\ p \mid d,\ p \mid n_2 \ \text{ and } \ \nu_p\left(\frac{n_2}{\gcd(n_2, i)}\right) < \nu_p(d) \text{ for a positive integer } 0 \leq i \leq n_2 - 1.$$

The prime p divides d if and only if $p \mid \frac{n}{m}$ and $p \mid \mathrm{ord}(\mathrm{ord}(\beta^m))$. Furthermore, $p \mid n_2$ if and only if $\nu_p(\frac{n}{m}) \geq \nu_p(\mathrm{ord}(\beta^m))$. Thus, the properties $p \mid d$ and $p \mid n_2$ are equivalent to $\nu_p(\frac{n}{m}) \geq \nu_p(\mathrm{ord}(\beta^m)) \geq 1$. From the fact that $\nu_p(\mathrm{ord}(\beta^m)) \leq \nu_p(\frac{n}{m})$ we can derive that $\nu_p(d) = \nu_p(\mathrm{ord}(\beta^m))$. Then since $\nu_p(n_2) = \nu_p(\frac{n}{m}) = \nu_p(d) + (\nu_p(\frac{n}{m}) - \nu_p(d))$ holds, the property $\nu_p\left(\frac{n_2}{\gcd(n_2,i)}\right) < \nu_p(d)$ is satisfied for $0 \leq i \leq n_2 - 1$ if and only if $\nu_p(i) > \nu_p(\frac{n}{m}) - \nu_p(d) = \nu_p(\frac{n}{m}) - \nu_p(\mathrm{ord}(\beta^m))$. This is equivalent to $\tilde{p} := p^{\nu_p(\frac{n}{m}) - \nu_p(\mathrm{ord}(\beta^m)) + 1}$ dividing i.

Obviously, $i = 0$ is divided by $\tilde{p}$ and condition (3.4) (ii) is violated if there exists a prime p such that $p \mid m$ and $1 \leq \nu_p(\mathrm{ord}(\beta^m)) \leq \nu_p(\frac{n}{m})$. Note that (3.3) implies $\nu_p(\mathrm{ord}(\beta^m)) \geq 1$ and we can omit this condition. Furthermore, $\mathrm{ord}(\beta^m) = \frac{\mathrm{ord}(\beta)}{\gcd(m, \mathrm{ord}(\beta))}$ and from $\nu_p(m) \geq 1$ and $\nu_p(\mathrm{ord}(\beta^m)) \geq 1$ follows that $\nu_p(\mathrm{ord}(\beta^m)) = \nu_p(\mathrm{ord}(\beta)) - \nu_p(m)$. This proves the statement of Theorem 3.15.

If there exists a prime p dividing m such that $\nu_p(\mathrm{ord}(\beta)) - \nu_p(m) \leq \nu_p(\frac{n}{m})$, in total exactly $\frac{n_2}{\tilde{p}}$ integers $0 \leq i \leq n_2 - 1$ satisfy that $\tilde{p} \mid i$. Since l runs through $\{0, \ldots, n_1 - 1\}$, the polynomial $(X^{\frac{n}{m}} - (\beta^m)^{\frac{n}{m}})$ has at least $n_1 \cdot \frac{n_2}{\tilde{p}} = \frac{n/m}{\tilde{p}}$ roots of order smaller than $\mathrm{ord}(\beta^m)$. Consequently, the polynomial $\chi_{\beta^n}(X^n)$ has at least $m \cdot \frac{n/m}{\tilde{p}} = \frac{n}{\tilde{p}}$ reducible factors of the form $\zeta_n^{-jk}\ f(\zeta_n^j X)$, maybe more for other primes p. □

The following theorem is a direct corollary of Theorems 3.13 and 3.15.

Corollary 3.16. *Let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k and $\beta \in \mathbb{F}_{q^k}^*$ a root of f. Let $n \in \mathbb{N}$ such that $\gcd(q, n) = 1$. Then the factorization of the polynomial $\chi_{\beta^n}(X^n)$ into monic irreducible factors over $\mathbb{F}_q$ is given by $\prod_{j=1}^{n} \zeta_n^{-jk}\ f(\zeta_n^j X)$ if and only if the following two conditions hold:*

(i) There exists a positive integer m such that $\frac{n}{m} \mid q - 1$ and a polynomial $g \in \mathbb{F}_q[X]$ such that $f = g(X^m)$.

(ii) There does not exist a prime p such that $p \mid m$ and $\nu_p(\mathrm{ord}(\beta)) - \nu_p(m) \leq \nu_p(\frac{n}{m})$.

Example 3.17. Again let $\mathbb{F}_4 = \mathbb{F}_2(a)$, where a is a root of the polynomial $X^2+X+1 \in \mathbb{F}_2[X]$. In this example we consider the monic irreducible polynomial

$$f = X^{30} + (a+1)X^{25} + (a+1)X^{20} + aX^{15} + aX^{10} + (a+1)X^5 + a + 1 \in \mathbb{F}_4[X]$$

It satisfies $f = g(X^5)$ for the monic irreducible polynomial

$$g = X^6 + (a+1)X^5 + (a+1)X^4 + aX^3 + aX^2 + (a+1)X^1 + a + 1 \in \mathbb{F}_4[X].$$

Let $n = 15$ and $m = 5$, then $\frac{n}{m} = 3$ divides $q-1$. Furthermore, its root β is an element of $\mathbb{F}_{2^{60}}$ satisfying $\operatorname{ord}(\beta) = 3^2 \cdot 5^2 \cdot 7 \cdot 13$. Thus, $\nu_5(\operatorname{ord}(\beta)) - \nu_5(m) = 1 > 0 = \nu_5(3)$ and with Corollary 3.16 formula (3.2) yields the factorization of $\chi_{\beta^{15}}(X^{15})$ into monic irreducible factors over $\mathbb{F}_4$.

As in Example 3.14, we can compute $\zeta_3^{-6j} g(\zeta_3^j X^5) = g(\zeta_3^j X^5)$ in $\mathbb{F}_4$ instead of $\zeta_{15}^{-30j} f(\zeta_{15}^j X)$. As j runs through the integers $1, \ldots, 15$, the polynomials $g(\zeta_3^j X^5)$ and $g(\zeta_3^{j+3} X^5)$ are equal. Therefore, there are only three distinct polynomials appearing five times each:

$$\begin{aligned}
g(\zeta_3 X^5) &= g(\zeta_3^4 X^5) = g(\zeta_3^7 X^5) = g(\zeta_3^{10} X^5) = g(\zeta_3^{13} X^5) \\
&= X^{30} + aX^{25} + X^{20} + aX^{15} + X^{10} + X^5 + a + 1 \\
g(\zeta_3^2 X^5) &= g(\zeta_3^5 X^5) = g(\zeta_3^8 X^5) = g(\zeta_3^{11} X^5) = g(\zeta_3^{14} X^5) \\
&= X^{30} + X^{25} + aX^{20} + aX^{15} + (a+1)X^{10} + aX^5 + a + 1 \\
f(X) &= g(\zeta_3^3 X^5) = g(\zeta_3^6 X^5) = g(\zeta_3^9 X^5) = g(\zeta_3^{12} X^5) = g(\zeta_3^{15} X^5) \\
&= X^{30} + (a+1)X^{25} + (a+1)X^{20} + aX^{15} + aX^{10} + (a+1)X^5 + a + 1
\end{aligned}$$

All three polynomials are monic irreducible polynomials over $\mathbb{F}_4$ and the factorization of $\chi_{\beta^5}(X^5)$ into monic irreducible polynomials over $\mathbb{F}_4$ is:

$$\begin{aligned}
\chi_{\beta^5}(X^5) = &(X^{30} + aX^{25} + X^{20} + aX^{15} + X^{10} + X^5 + a + 1)^5 \\
&\cdot (X^{30} + X^{25} + aX^{20} + aX^{15} + (a+1)X^{10} + aX^5 + a + 1)^5 \\
&\cdot (X^{30} + (a+1)X^{25} + (a+1)X^{20} + aX^{15} + aX^{10} + (a+1)X^5 + a + 1)^5.
\end{aligned}$$

Note that in both Example 3.14 and Example 3.17 the multiplicity of the irreducible factors of $\chi_{\beta^n}(X^n)$ was exactly the number of polynomials of the form $\zeta_n^{-jk} f(\zeta_n^j X)$ that were equal. Since with Corollary 3.11 this is the exponent of the minimal polynomial m_{β^n} in the characteristic polynomial χ_{β^n}, we are interested in the conditions under which the polynomials of the form $\zeta_n^{-jk} f(\zeta_n^j X)$ are equal. Theorem 3.19 specifies these conditions and we use the following proposition for its proof. Note that here the primitive n-th root of unity ζ_n is not necessarily an element of $\mathbb{F}_q$.

Proposition 3.18 ([GK23, Proposition 6(a)])**.** *Let $n \in \mathbb{N}$ such that $\gcd(q,n) = 1$ and $f \in \mathbb{F}_q[X]$. Then there exists $g \in \mathbb{F}_q[X]$ such that $f(X) = g(X^n)$ if and only if $f(X) = f(\zeta_n X)$.*

Proof. If $f(X) = g(X^n)$, then $f(\zeta_n X) = g(\zeta_n^n X^n) = g(X^n) = f(X)$. Vice versa, suppose that $f(X) = f(\zeta_n X)$. Then if $f(X) = \sum_{i=0}^k a_i X^i$, we have $f(\zeta_n X) = \sum_{i=0}^k a_i \zeta_n^i X^i$. Thus, $\zeta_n^i = 1$ for all $0 \le i \le k$ such that $a_i \neq 0$. Consequently, $n = \operatorname{ord}(\zeta_n) \mid i$ for all $0 \le i \le k$ such that $a_i \neq 0$. □

The next theorem shows that for any n with $\gcd(q,n) = 1$, polynomials of the form $\zeta_n^{-jk} f(\zeta_n^j X)$ for distinct $1 \le j \le n$ can only be equal if there exists a divisor m of $\gcd(n,k)$ greater than 1 such that $f(X) = g(X^m)$ for a polynomial $g \in \mathbb{F}_q[X]$.

Theorem 3.19 ([GK23, Theorem 7]). *Let $n \in \mathbb{N}$ such that $\gcd(n,q)=1$ and let $f \in \mathbb{F}_q[X]$ be a polynomial of degree k such that $f(0) \neq 0$. Set*

$$t = \max\{m \mid \gcd(n,k) : f(X) = g(X^m) \text{ for a polynomial } g \in \mathbb{F}_q[X]\}.$$

Then for $0 \leq j, j' \leq n-1$ the two polynomials $\zeta_n^{-jk} f\left(\zeta_n^j X\right)$ and $\zeta_n^{-j'k} f\left(\zeta_n^{j'} X\right)$ are equal if and only if $j \equiv j' \mod \frac{n}{t}$.

Proof. "⇐": Note that since $t \mid n$ the element $\zeta_n^{\frac{n}{t}} = \zeta_t$ generates the subgroup U_t of the t-th roots of unity of $\mathbb{F}_q^*$. If $j \equiv j' \mod \frac{n}{t}$, then $j - j' = v \cdot \frac{n}{t}$ for an integer v and we have

$$\begin{aligned}\zeta_n^{-jk} f\left(\zeta_n^j X\right) = \zeta_n^{-(j-j')k}\zeta_n^{-j'k} f\left(\zeta_n^{(j-j')}\zeta_n^{j'} X\right) &= \zeta_n^{-\frac{n}{t}\cdot v\cdot k}\zeta_n^{-j'k} f\left(\zeta_n^{\frac{n}{t}\cdot v}\zeta_n^{j'} X\right)\\ = \zeta_n^{-n\cdot v\cdot \frac{k}{t}}\zeta_n^{-j'k} f\left(\zeta_t^v \zeta_n^{j'} X\right) &= \zeta_n^{-j'k} f\left(\zeta_t^v \zeta_n^{j'} X\right).\end{aligned}$$

From the definition of t and Proposition 3.18 follows that $f(X) = f(\zeta_t X)$ and therefore also $f(X) = f(\zeta_t^v X)$. Thus, $\zeta_n^{-j'k} f\left(\zeta_t^v \zeta_n^{j'} X\right) = \zeta_n^{-j'k} f\left(\zeta_n^{j'} X\right)$.

"⇒": Suppose that $\zeta_n^{-jk} f\left(\zeta_n^j X\right) = \zeta_n^{-j'k} f\left(\zeta_n^{j'} X\right)$. Then also

$$\begin{aligned}{\zeta_n}^{-(j-j')k} f\left(\zeta_n^{j-j'} X\right) &= \zeta_n^{j'k} \cdot \zeta_n^{-jk} f\left(\zeta_n^j \left(\zeta_n^{-j'} X\right)\right)\\ &= \zeta_n^{j'k} \cdot \zeta_n^{-j'k} f\left(\zeta_n^{j'} \left(\zeta_n^{-j'} X\right)\right) = f(X)\end{aligned} \tag{3.7}$$

Let $f = \sum_{i=0}^k a_i X^i \in \mathbb{F}_q[X]$. Then we have

$$\zeta_n^{-(j-j')k} f\left(\zeta_n^{j-j'} X\right) = \sum_{i=0}^{k} a_i \zeta_n^{-(j-j')(k-i)} X^i.$$

For this polynomial to be equal to $f(X)$, we need $n \mid (j-j')(k-i)$ for all $a_i \neq 0$. Note that $a_0 = f(0) \neq 0$. Consequently, $n \mid (j-j')\cdot k$. Let $d := \gcd(n,k)$, then $\frac{n}{d} \mid (j-j')$ and there exists $v \in \mathbb{N}$ such that $j - j' = v \cdot \frac{n}{d}$. Furthermore, the element $\zeta_n^{\frac{n}{d}} = \zeta_d$ generates the subgroup U_d of the d-th roots of unity of $\mathbb{F}_q$ and we obtain

$$\zeta_n^{-(j-j')k} f\left(\zeta_n^{(j-j')} X\right) = \zeta_n^{-v\cdot\frac{n}{d}\cdot d\cdot\frac{k}{d}} f\left(\zeta_n^{v\cdot\frac{n}{d}} X\right) = f\left(\zeta_d^v X\right). \tag{3.8}$$

If $l = \frac{d}{\gcd(d,v)}$, the element $\zeta_d^v = \zeta_l$ generates the set U_l of the l-th roots of unity over $\mathbb{F}_q$. Equations (3.7) and (3.8) yield that $f(X) = f(\zeta_l X)$. Note that $\gcd(d,q) = 1$ and with Proposition 3.18 we obtain that $M := \{m \mid d : f(X) = g(X^m), g \in \mathbb{F}_q[X]\}$ is equal to the set $\{m \mid d : f(X) = f(\zeta_m X)\}$ and consequently, $l \in M$.

Let $t := \max M$. We will prove that M is in fact the set of all divisors of t. Note that if $f(X) = f(\zeta_t X)$, also $f(X) = f\left(\zeta_t^i X\right)$ for all $1 \leq i \leq t$ and any divisor m of t satisfies that $\zeta_m = \zeta_t^{\frac{t}{m}}$. Thus, all divisors of t are elements of M.

Suppose that there exists an element $m \in M$ such that m does not divide t. Then for all $0 \leq i \leq k$ such that $a_i \neq 0$, we have $m \mid i$ and $t \mid i$. Consequently, the least common multiple of m and t, $\operatorname{lcm}(m,t) = t \cdot \frac{m}{\gcd(m,t)}$, divides i. Since both m and t divide d, we obtain that $\operatorname{lcm}(t,m) \in M$. But $\operatorname{lcm}(t,m) > t$, because $m \nmid t$. This is a contradiction to the choice of t and M is in fact the set of all divisors of t.

Thus, the fact $l \in M$ is equivalent to $l \mid t$. Recall that $l = \frac{d}{\gcd(d,v)}$ and therefore $\frac{d}{\gcd(d,v)} \mid t$ which is equivalent to $\frac{d}{t} \mid \gcd(d,v)$ and this again is equivalent to $\frac{d}{t} \mid v$. Thus, there exists an integer w such that $v = \frac{d}{t} \cdot w$. Since $v = \frac{j-j'}{\frac{n}{d}}$, we have $j - j' = \frac{n}{t} \cdot w$, which is equivalent to $j \equiv j' \mod \frac{n}{t}$. □

Example 3.20. Again let $\mathbb{F}_4 = \mathbb{F}_2(a)$, where a is a root of the polynomial $X^2+X+1 \in \mathbb{F}_2[X]$. First, we check the statement of Theorem 3.19 for the polynomial $f = X^6 + X^3 + a + 1 \in \mathbb{F}_4[X]$ from Example 3.14. For $n = 9$ we have $\gcd(n, k) = \gcd(6, 9) = 3$ and since $f = g(X^3)$ for $g = X^2 + X + a + 1$ the positive integer t equals 3. Then Theorem 3.19 yields that two polynomials $\zeta_9^{-6j} f(\zeta_9^j X)$ and $\zeta_9^{-6j'} f(\zeta_9^{j'} X)$ are equal if and only if $j \equiv j' \mod \frac{9}{3} = 3$. Recall that the polynomials $\zeta_9^{-j6} f(\zeta_9^j X)$ and $\zeta_9^{-j'6} f(\zeta_9^{j'} X)$ were equal if and only if $j \equiv j' \pmod 3$ and the statement is correct for this polynomial f.

Next, we consider $f = X^{30}+(a+1)X^{25}+(a+1)X^{20}+aX^{15}+aX^{10}+(a+1)X^5+a+1 \in \mathbb{F}_4[X]$ from Example 3.17 and the positive integer $n = 15$. Then $\gcd(n, k) = \gcd(15, 30) = 15$. The largest divisor t of 15 such that $f = g(X^t)$ is $t = 5$. Theorem 3.19 yields that the polynomials $\zeta_{15}^{-30j} f(\zeta_{15}^j X)$ and $\zeta_{15}^{-30j'} f(\zeta_{15}^{j'} X)$ are equal if and only if $j \equiv j' \mod \frac{15}{5} = 3$, which matches the observations in Example 3.17.

Theorem 3.13 shows that if we want to compute the composition $\chi_{\beta^n}(X^n)$ in $\mathbb{F}_q$, we need to choose a polynomial $f \in \mathbb{F}_q[X]$ such that $f = g(X^m)$ and $\frac{n}{m} \mid q-1$. If we choose a positive integer n such that $n \mid (q-1)$ this condition is satisfied for every irreducible polynomial f over $\mathbb{F}_q$. This is good for the application of our construction because the user does not need to check any additional condition on the polynomial f besides irreducibility.

If $n \mid (q-1)$, then ζ_n is an element of $\mathbb{F}_q$ and for every $1 \le j \le n$ the element $\zeta_n^{-j}\beta$ is a proper element of $\mathbb{F}_{q^k}$. Since $\zeta_n^{-j}\beta$ is a root of $\zeta_n^{-jk} f(\zeta_n^j X) \in \mathbb{F}_q[X]$, this polynomial is the minimal polynomial of $\zeta_n^{-j}\beta$ over $\mathbb{F}_q$ and (3.2) yields the factorization of $\chi_{\beta^n}(X^n)$ into monic irreducible factors over $\mathbb{F}_q$.

Theorem 3.19 and Corollary 3.11 combined yield the following direct formula for the computation of $m_{\beta^n}(X^n)$ for all positive integers n satisfying $n \mid q-1$:

Theorem 3.21 ([GK23, Corollary 8])**.** *Let $n \in \mathbb{N}$ such that $n \mid q-1$ and let $f \in \mathbb{F}_q[X]$, $f \neq X$, be a monic irreducible polynomial of degree k. Further, let $\beta \in \mathbb{F}_{q^k}$ be a root of f and $m_{\beta^n} \in \mathbb{F}_q[X]$ be the minimal polynomial of β^n over $\mathbb{F}_q$. Set $t = \max\{m \mid \gcd(n,k) : f(X) = g(X^m)$ for a polynomial $g \in \mathbb{F}_q[X]\}$. Then*

$$m_{\beta^n}(X^n) = \prod_{j=1}^{\frac{n}{t}} \zeta_n^{-jk} f\left(\zeta_n^j X\right).$$

Proof. Application of Theorem 3.19 to equation (3.2) yields:

$$\chi_{\beta^n}(X^n) = \prod_{j=1}^{n} \zeta_n^{-jk} f\left(\zeta_n^j X\right) = \left(\prod_{j=1}^{\frac{n}{t}} \zeta_n^{-jk} f\left(\zeta_{n^j} X\right)\right)^t.$$

Furthermore, from Theorem 3.19 follows that the polynomials $\zeta_n^{-jk} f\left(\zeta_n^j X\right)$ for $1 \le j \le \frac{n}{t}$ are distinct. Then Corollary 3.11 completes the proof. □

Recall that Construction KK only considers the case where $\chi_{\beta^2} = m_{\beta^2}$ and Construction AD relies on the derivation of m_{β^n} from χ_{β^n}, if the order of f is known, or on a complete factorization of χ_{β^n} in $\mathbb{F}_q[X]$ in order to obtain the monic irreducible polynomial m_{β^n}. Theorem 3.21 allows us to compute the polynomial $m_{\beta^n}(X^n)$ directly, from which m_{β^n} can easily be extracted as the following example shows:

Example 3.22. We consider the monic irreducible polynomial $f = X^6 + X^3 + a \in \mathbb{F}_4[X]$ and the positive integer $n = 3$ which divides $4-1$. Then $t = 3$, because $3 \mid \gcd(n, 6) = 3$ and $f = g(X^3)$ for the polynomial $g = X^2 + X + a$. Theorem 3.21 yields that

$$m_{\beta^3}(X^3) = \prod_{j=1}^{1} \zeta_3^{-j6} f(\zeta_3^j X) = g((\zeta_3 X)^3) = g(X^3).$$

Consequently, $m_{\beta^3} = g = X^2 + X + a$.

For the monic irreducible polynomial $f = X^6 + X^5 + X + a \in \mathbb{F}_4[X]$ we have $t = 1$ and Theorem 3.21 implies that

$$\begin{aligned} m_{\beta^3}(X^3) &= \prod_{j=1}^{3} \zeta_3^{-j6} f(\zeta_3^j X) \\ &= (X^6 + (a+1)X^5 + aX + a) \cdot (X^6 + aX^5 + (a+1)X + a) \cdot (X^6 + X^5 + X + a) \\ &= X^{18} + X^{15} + (a+1)X^{12} + X^6 + X^3 + 1. \end{aligned}$$

We can easily extract $m_{\beta^3} = X^6 + X^5 + (a+1)X^4 + X^2 + X + 1$ from this composition. Note that since $n = 3$ is prime, only the two options $t = 1$ and $t = 3$ exist. This observation is discussed in general in the following remark.

Remark 3.23. If $n \mid (q-1)$ and n is prime, then the parameter t from Theorem 3.21 is greater than 1 if and only if $t = n$. Thus, only two cases can arise. Either $t = 1$ and we obtain the minimal polynomial of β^n over $\mathbb{F}_q$ by extracting it from the composition $m_{\beta^n}(X^n) = \prod_{j=1}^{n} \zeta_n^{-jk} f\left(\zeta_n^j X\right) = (-1)^{k(n+1)} \prod_{j=1}^{n} f\left(\zeta_n^j X\right)$. Or $t = n$, which implies that $f(X) = g(X^n)$ for a polynomial $g \in \mathbb{F}_q[X]$. Then $m_{\beta^n}(X^n) = \zeta_n^{-k} f(\zeta_n X) = 1 \cdot g((\zeta_n X)^n) = g(X^n)$ and consequently $m_{\beta^n} = g$, which we easily extract from $f = g(X^n)$.

3.3 The new recursive construction of m_{β^n} from m_β

Observe that Theorem 3.21 can be applied recursively in the following way:

Let n be a postive integer such that $n = n_1 \cdots n_m$ and $n_i \mid q-1$ for all $1 \leq i \leq m$. Let $k \in \mathbb{N}$ and β be a proper element of $\mathbb{F}_{q^k}$. Then the polynomial $m_{\beta^{n_1}}(X^{n_1})$ can be computed directly from m_β with Theorem 3.21. From this polynomial we easily extract $m_{\beta^{n_1}}$. In the next step we compute $m_{\beta^{n_1 \cdot n_2}}(X^{n_2})$ from $m_{\beta^{n_1}}$ with Theorem 3.21 and extract $m_{\beta^{n_1 \cdot n_2}}$. This procedure can be repeated m times and in the end we obtain $m_{\beta^{n_1 \cdots n_m}}(X^{n_m})$ from $m_{\beta^{n_1 \cdots n_{m-1}}}$ with Theorem 3.21 and extract the monic irreducible polynomial $m_{\beta^{n_1 \cdots n_m}} = m_{\beta^n}$.

Combining these ideas, the unique prime factorization of a positive integer n and Remark 3.23, we obtain the following new construction of m_{β^n} from m_β for any positive integer such that $\mathrm{rad}(n) \mid (q-1)$:

Construction 1. *Let $n \in \mathbb{N}$ such that $n = n_1 \cdots n_m$ where $n_1, \ldots, n_m$ are prime factors of $q-1$ (which are not necessarily distinct). Further, let $f \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k. Set $f_0 := f$. For $1 \leq i \leq m$ compute the monic irreducible polynomial f_i in the following way:*
If there exists a polynomial $g \in \mathbb{F}_q[X]$ such that $f_{i-1}(X) = g(X^{n_i})$, then $f_i = g$. Otherwise, compute

$$(-1)^{\deg(f_{i-1}) \cdot (n_i+1)} \prod_{j=1}^{n_i} f_{i-1}\left(\zeta_{n_i}^j X\right) = f_i(X^{n_i}) \tag{3.9}$$

and extract f_i from the composition. Then f_m is the minimal polynomial of $\beta^n \in \mathbb{F}_{q^k}$ over $\mathbb{F}_q$, where $\beta \in \mathbb{F}_{q^k}$ is a root of f.

An implementation of one single step of Construction 1 for $n = p$ prime is included in the class `RichPolynomial` as the function `construction_mbetap` and we present the code in detail below. Note that for the computation of (3.9) the polynomial needs to be used as a polynomial, which is the data structure `Polynomial` from the module `sage.rings.polyno-`

`mial.polynomial_element` in SageMath, and for the extraction of m_{β^p} from the composition $m_{\beta^p}(X^p)$ the polynomial needs to be considered as a list of coefficients for which we need the Python data structure `list`. The switching between these two data structures of a polynomial is what the class `RichPolynomial` has been designed for. Additionally, the function `construction_mbetap` makes use of the two functions `is_composition` and `extract` of the class `RichPolynomial`, which check if a polynomial is a composition with X^p and can extract a polynomial g from a composition $g(X^p)$.

Source Code 3.1: Construction of m_{β^p}

```
def construction_mbetap(self, p):
    # checks if p divides q-1:
    if p not in (self._RFF).get_primefactors():
        raise ValueError(str(p)+" is not a prime factor of "+str((self._RFF
    ).q)+"-1.")

    if self.is_composition(p):
        # if f = g(X^p), then m_{beta^p}(X^p)=f
        m_betap_Xp = self
    else:
        # if f is no composition with X^p, then m_{beta^p}(X^p) is computed
     with formula:
        m_beta = self.get_polynomial()
        zeta_p = (self._RFF).get_unityroot(p)

        m_betap_Xp = (-1)^(self.deg*(p+1)) * prod([m_beta(zeta_p^j*(self._
    RFF).x) for j in range(p)])
        m_betap_Xp = RichPolynomial(m_betap_Xp, self._RFF)

    return m_betap_Xp.extract(p)
```

Example 3.24. Let $\mathbb{F}_q = \mathbb{F}_4$ as before and we consider again the monic irreducible polynomial $f = X^6 + X^3 + a \in \mathbb{F}_4[X]$. The positive integer $n = 27$ satisfies $\mathrm{rad}(n) = 3 \mid (4-1)$ and $n = n_1 \cdot n_2 \cdot n_3$ for $n_1 = n_2 = n_3 = 3$.

Then $f_0 := f$ and since $f_0 = g(X^3)$ for the monic irreducible polynomial $g = X^2 + X + a \in \mathbb{F}_4[X]$, we set $f_1 = g$. Since f_1 is not a composition with X^3, we compute $f_2(X^3)$ by (3.9) and extract f_2. For this we use `construction_mbetap`, which yields that

$$f_2 = X^2 + (a+1)X + 1.$$

At last, with `construction_mbetap` we obtain that

$$f_3 = X^2 + aX + 1 = m_{\beta^{27}}.$$

As we can see from Source Code 3.1 all computations of Construction 1 are carried out in $\mathbb{F}_q$ and the construction is based solely on the examination of the non-zero coefficients of the polynomials f_i. Where Construction AD relies on the order of the initial polynomial f for the derivation of m_{β^n} from χ_{β^n} or a complete factorization of χ_{β^n} in $\mathbb{F}_q[X]$, the polynomials f_i are computed directly from f_{i-1} in each step of Construction 1.

Remark 3.25. (a) In Example 3.24 the degree of f_1, f_2, f_3 was strictly lower than the degree of the initial polynomial f. If we select positive integers n such that one of the following two conditions is satisfied, then all polynomials obtained with Construction 1 are of the same degree k as the initial polynomial f:

(i) $\gcd(n, k) = 1$

(ii) The order $\frac{e}{\gcd(e,n)}$ of the minimal polynomial of β^n does not divide $q^{\frac{k}{t}} - 1$ for any divisor t of k, whose prime factors divide $\gcd(n, k)$.

(b) If there exists a polynomial $g \in \mathbb{F}_q[X]$ such that $f = g\left(X^t\right)$ for a prime divisor t of n, then the minimal polynomial of β^n will be of lower degree. Observe that in this case the polynomial $f(X+a)$ for any element $a \in \mathbb{F}_q \backslash \{0\}$ will not be a composition with X^m for any positive integer $m > 1$ and could be used instead of f. This fact was proved in [KK22] for $t = 2$. For the convenience of the reader, we include the generalized proof here.

Proof. If $f(X) = \sum_{i=0}^{k} b_i X^i = g\left(X^t\right)$ for $t > 1$, then $b_{k-1} = 0$ and since f is monic, we have $b_k = 1$. Furthermore,

$$\begin{aligned} f(X+a) &= (X+a)^k + \underbrace{\sum_{i=0}^{k-2} b_i\ (X+a)^i}_{\deg(\ldots)<k-1} \\ &= \sum_{j=0}^{k} \binom{k}{j} a^j X^{k-j} + \sum_{i=0}^{k-2} b_i (X+a)^i \\ &= X^k + kaX^{k-1} + \sum_{j=2}^{k} \binom{k}{j} a^j X^{k-j} + \sum_{i=0}^{k-2} b_i (X+a)^i. \end{aligned}$$

Since $\gcd(k, q) = 1$, $\operatorname{char}(\mathbb{F}_q)$ does not divide k from which follows that $ka \neq 0$ and there cannot exist any positive integer $m > 1$ such that $f(X+a) = h\left(X^m\right)$ for a polynomial $h \in \mathbb{F}_q[X]$. □

Theorem 3.3 states that Construction AD yields all monic irreducible polynomials whose order divides the order of the initial polynomial f. This poses the question which of these polynomials can be constructed with Construction 1. Obviously, Construction 1 can be applied for every positive integer n such that $\operatorname{rad}(n) \mid (q-1)$. However, many of these integers yield the same polynomial. We mentioned before that Constructions AD and KK show a periodic behavior when applied recursively. Our new construction shows the same behavior.

The following construction generalizes Construction 2 from [KK22]. It is a recursive application of Construction 1 for a fixed prime integer n dividing $q-1$.

Construction 2. *Let n be a prime factor of $q-1$ and $f \in \mathbb{F}_q[X]$ be a monic irreducible polynomial of degree k. Further let $w = \nu_n\left(q^k - 1\right)$ be the n-adic valuation of $q^k - 1$. Set $f_0 := f$. For $i \geq 1$ compute the monic irreducible polynomial f_i in the following way:*
If there exists a polynomial $g \in \mathbb{F}_q[X]$ such that $f_{i-1}(X) = g\left(X^n\right)$, then $f_i = g$. Otherwise, compute

$$(-1)^{\deg(f_{i-1})\cdot(n+1)} \prod_{j=1}^{n} f_{i-1}\left(\zeta_n^j X\right) = f_i\left(X^n\right)$$

and extract f_i from the composition. If $f_i = f_l$ for an integer l such that $0 \leq l \leq w$ and $l < i$, then stop.

With the notation from Construction 2, suppose that the construction terminates for the polynomial f_{l+s} which is equal to f_l, for integers $s \geq 1$ and $0 \leq l \leq \nu_k(q^k - 1)$. Then we call the sequence

$$(f_0, f_1, \ldots, f_{l-1})$$

the *tail* of the construction and the sequence

$$(f_l, \ldots, f_{l+s-1})$$

the *orbit*. Note that the construction would yield the polynomials of the orbit repeatedly if we continued to iterate through the integers $i \geq l + s$. This is the periodic behavior which we mentioned before. Observe that the length of the tail is l and the length of the orbit s.

The following result shows that the two parameters l and s can be used to gain information on the order of the initial polynomial f.

Theorem 3.26. *With the notation from Construction 2, we suppose that Construction 2 terminated after a tail of length l and an orbit of length s.*

Then $\operatorname{ord}(f) = n^l \cdot r$ *and r must satisfy*

(i) $\gcd(n, r) = 1$,

(ii) $s = \frac{\operatorname{ord}_r(n)}{d}$ *for a divisor d of* $\deg(f_l)$,

(iii) Furthermore, for an integer $0 \leq j \leq \deg(f_l) - 1$, *$d$ must satisfy* $\operatorname{ord}_r\left(q^j\right) = d$ *and* $n^s \equiv q^j \mod r$.

Proof. Let $\beta \in \mathbb{F}_{q^k}$ be a root of f, that is, $f = m_\beta$ is the minimal polynomial of β over $\mathbb{F}_q$. Then with Construction 1 we know that $f_i = m_{\beta^{n^i}}$ for every $i \geq 0$. Further, let $\operatorname{ord}(f) = e$ and $e = n^v \cdot r$ with $\gcd(n, r) = 1$. Then with Theorem 3.2 the minimal polynomial of β^{n^i}, that is, the polynomial f_i, has order

$$\operatorname{ord}(f_i) = \begin{cases} \frac{e}{n^i} = n^{v-i} \cdot r & \text{for } 0 \leq i \leq v, \\ r & \text{for } i \geq v. \end{cases} \tag{3.10}$$

Since the order of the polynomials $(f_0, f_1, \ldots, f_{v-1})$ strictly decreases, these polynomials cannot appear twice in the sequence $(f_i)_{i\geq 0}$. Note that $v \leq w = \nu_n(q^k - 1)$. Thus, the polynomial f_v, which is the first polynomial of order r of the sequence $(f_i)_{i\geq 0}$, is an element of the sequence $(f_0, f_1, \ldots, f_w)$.

We need to examine $\mathbb{Z}_r^*$, the multiplicative group modulo r, to see that f_v is the first polynomial to appear twice in the sequence $(f_i)_{i\geq 0}$ and therefore $v = l$. The subgroup $\langle n \rangle$ of $\mathbb{Z}_r^*$ generated by n has order $\operatorname{ord}_r(n)$, which is the multiplicative order of n modulo r. This implies that $\beta^{n^{v+\operatorname{ord}_r(n)}} = \beta^{n^v}$ and obviously the minimal polynomials of β^{n^v} and $\beta^{n^{v+\operatorname{ord}_r(n)}}$ over $\mathbb{F}_q$ are equal. Thus, $f_v = f_{v+\operatorname{ord}_r(n)}$ and we have shown that f_v does appear again in the sequence.

However, the length s of the orbit is not always equal to $\operatorname{ord}_r(n)$. The polynomials f_{i_1} and f_{i_2} are equal if and only if $\beta^{n^{i_1}}$ and $\beta^{n^{i_2}}$ are $\mathbb{F}_q$-conjugates. Thus, it is possible that there exists a positive integer u smaller than $\operatorname{ord}_r(n)$ such that $\beta^{n^{v+u}}$ is an $\mathbb{F}_q$-conjugate of β^{n^v} and the minimal polynomial $f_{v+u} = m_{\beta^{n^{v+u}}}$ also is equal to $f_v = m_{\beta^{n^v}}$. To account for this, we choose $u \in \mathbb{N}$ to be the smallest positive integer that satisfies

$$\langle n \rangle \cap \langle q \rangle = \langle n^u \rangle = \langle q^j \rangle \leq \mathbb{Z}_r^* \quad \text{for an integer } 0 \leq j \leq \deg(f_v) - 1. \tag{3.11}$$

Note that since $\langle n^{\operatorname{ord}_r(n)} \rangle = \langle q^0 \rangle$ such an integer u exists and satisfies $u \leq \operatorname{ord}_r(n)$. Then $\beta^{n^{v+u}} = \left(\beta^{n^v}\right)^{q^j}$ and $f_{v+u} = f_v$. Moreover, the minimal polynomials of $\beta^{n^{v+i}}$ for $0 \leq i \leq u-1$ are distinct because we selected u to be the smallest positive integer to satisfy (3.11). Consequently, $v = l$, which shows that (i) holds, and the length s of the orbit equals u.

Set $d := |\langle q^j \rangle| = |\langle n^s \rangle|$ which is a divisor of $\deg(f_l)$, since $\langle q^j \rangle \leq \langle q \rangle$ and $|\langle q \rangle| = \deg(f_l)$. Then because of $\langle n^s \rangle$ being a subgroup of $\langle n \rangle$, we have $s = \frac{|\langle n \rangle|}{|\langle n^s \rangle|} = \frac{\operatorname{ord}_r(n)}{d}$ which shows that (ii) holds. (iii) follows directly from equation (3.10) and our definition of d. □

Remark 3.27. In the original version of [KK22] on ArXiV, [KK20], the number of distinct polynomials produced by [KK22, Construction 1] was given as $\operatorname{ord}_r(2)$ where $\operatorname{ord}(f) = 2^v r$ with $v \geq 0$ and $r \geq 1$ odd. As we can see from Theorem 3.26 this number is not correct,

since the authors did not take into consideration that the construction could also yield the minimal polynomials of $\mathbb{F}_q$-conjugates over $\mathbb{F}_q$. Similarly, in [KK22, Remark 1] the information about the order of the initial polynomial $C_0(X)$ obtained by the construction should be changed to: $2^l t$ where t is an odd divisor of $q^n - 1$ and $k - l = \frac{\mathrm{ord}_t(2)}{d}$ for a divisor d of n.

Note that with equation (3.10) in the proof of Corollary 3.26 the polynomials f_i for $0 \leq i \leq l-1$ of the tail of Construction 2 have order $n^{l-i} \cdot r$ and all polynomials of the orbit have order r.

If $p_1, \ldots, p_m$ are the distinct prime factors of $q-1$, and $\mathrm{ord}(f) = e = p_1^{v_1} \cdots p_m^{v_m} \cdot r$ with $\gcd(q, r) = 1$ and $v_1 \geq 0, \ldots, v_m \geq 0$. Then Construction 2 allows us to determine the p_i-adic valuations $v_1, \ldots, v_m$ of the order of f. Additionally, Corollary 3.26 ((ii)) and ((iii)) give further conditions on the factor r.

In the module B.12 `ConstructionClasses.sage` there exists an implementation of Construction 2 and Theorem 3.26 as the class `Construction2`. In Section 3.4 we explain in detail how to use it.

Example 3.28. Let $q = 4$ and $\mathbb{F}_4 = \mathbb{F}(a)$, where a is a root of the polynomial $X^2 + X + 1$. We can apply Construction 2 for the prime factor 3 of $4 - 1$. For the monic irreducible polynomial $f = X^6 + X^5 + X + a \in \mathbb{F}_4[X]$ the construction yields the following result:

p	$\nu_p(\mathrm{ord}(f))$	orbit length s
3	2	12

Since the order of f divides $4^6 - 1 = 4095 = 3^2 \cdot 5 \cdot 7 \cdot 13$, there are 24 possibilities. If we apply Theorem 3.26, we can remove 16 of them because they do not satisfy the 3-adic valuation. Then 5 more can be removed because the orbit length s does not satisfy Theorem 3.26 (ii), that is, s does not equal $\frac{\mathrm{ord}_r(3)}{d}$ for a divisor d of 6. For the remaining three possibilities s equals the order $\mathrm{ord}_r(p)$ and $j = 0$ satisfies Theorem 3.26(iii). Thus,

$$\mathrm{ord}(f) \in \{3^2 \cdot 5 \cdot 7, 3^2 \cdot 5 \cdot 13, 3^2 \cdot 5 \cdot 7 \cdot 13\}.$$

The polynomial is in fact of order $4095 = 3^2 \cdot 5 \cdot 7 \cdot 13$.

We consider the polynomial

$$f = X^6 + (a^2 + 1)X^3 + a + 1 \in \mathbb{F}_{16}[X],$$

where $\mathbb{F}_{16}$ equals $\mathbb{F}(a)$ and a is a root of the polynomial $X^4 + X + 1$. We can apply Construction 2 for the two prime factors 3 and 5 of $16 - 1$. We obtain:

p	$\nu_p(\mathrm{ord}(f))$	orbit length s
3	2	8
5	1	8

There exist 96 divisors of $16^6 - 1 = 3^2 \cdot 5 \cdot 7 \cdot 13 \cdot 17 \cdot 241$. 80 of them do not satisfy the p-valuations given by Construction 2. When considering $p = 3$ and the condition (ii) of Theorem 3.26, for 12 divisors the orbit length s is not of the form $\frac{\mathrm{ord}_r(3)}{d}$ for a divisor d of 6 and for one divisor (iii) is violated. From the remaining three candidates two can be eliminated when considering Theorem 3.26 (ii) for $p = 5$. Consequently, Theorem 3.26 yields the order of f, which is $\mathrm{ord}(f) = 765$.

For all polynomials that we tested `Construction 2` for, the algorithm could reduce the number of candidates significantly or give the exact order of the polynomial. However, the computation does not seem to be more efficient than existing algorithms, which compute the exact order. The biggest advantage is that all computations are done in $\mathbb{F}_q$.

3.4 How to find all polynomials generated by Construction 1 efficiently

In this section we present a procedure to generate all polynomials that can be obtained from a fixed monic irreducible polynomial f with Construction 1 efficiently. For this, we use the periodic behavior of Construction 2.

The ideas in this section have been presented before in [GK23] but we present them here in much more detail. In particular, we give all proofs which were not included in [GK23].

Let $p_1, \dots, p_m$ be the distinct prime factors of $q-1$. Then we can apply Construction 1 for any integer n that is an element of the set

$$\mathcal{N} := \{p_1^{i_1} \cdots p_m^{i_m} : i_1, \dots, i_m \geq 0\}.$$

Suppose that f is of degree k, has order e and $\beta \in \mathbb{F}_{q^k}$ is a root of f. Since the element β has multiplicative order e, for every $n \in \mathcal{N}$ holds $\beta^n = \beta^{n \bmod e}$. Thus, the set of polynomials that we can construct with the integers in $\mathcal{N}$ is

$$\mathcal{M} := \{m_{\beta^{n \bmod e}} : n = p_1^{i_1} \dots p_m^{i_m}, i_1, \dots, i_m \geq 0\}. \tag{3.12}$$

However, we would like to emphasize that the construction should not be restricted to the elements of $\mathcal{N}$ which are smaller than e, here denoted by $\mathcal{N}_{<e}$. An integer $n \in \mathcal{N}$, $n \geq e$, can yield a polynomial that cannot be constructed by choosing all elements of $\mathcal{N}_{<e}$. This is the case if $(n \bmod e)$ is not an element of $\mathcal{N}$ as can be seen from the following example:

Example 3.29. Let $\mathbb{F}_8 = \mathbb{F}(a)$ where a is a root of the monic irreducible polynomial $X^3 + X + 1 \in \mathbb{F}_2[X]$. We consider the primitive monic irreducible polynomial

$$f = X^5 + aX^4 + X^3 + aX^2 + (a^2 + a)X + a^2 \in \mathbb{F}_8[X]$$

of order $e = 32\,767 = 7 \cdot 31 \cdot 151$. There exists only the prime factor 7 of $8-1$ and we can apply Construction 1 for all elements of $\mathcal{N} = \{7^i : i \geq 0\}$. Note that this the same as an application of Construction 2. The construction yields a tail of length 1 and an orbit of length 150. Thus, m_{β^7} and $m_{\beta^{7^{151}}}$ are equal, where β is a root of f.

However, the smallest positive integer i such that 7^i is greater than or equal to e is 6, which implies that $\mathcal{N}_{<e} = \{1, 2, 3, 4, 5\}$. The positive integer $7^6 \pmod e = 117\,649 \pmod e = 19\,348 = 2^2 \cdot 7 \cdot 691$ is not an element of $\mathcal{N}$. Thus, if we had restricted ourselves to $\mathcal{N}_{<e}$, we would only have found 5 of the 151 possible polynomials.

The number of polynomials that we can construct with Construction 1, which is the size of $\mathcal{M}$, obviously depends on the size of $\mathcal{N}$ considered in $\mathbb{Z}_e$:

$$\mathcal{N}_{\bmod e} = \{p_1^{i_1} \cdots p_m^{i_m} \bmod e : i_1, \dots, i_m \geq 0\}. \tag{3.13}$$

Indeed, we have $\mathcal{M} = \{m_{\beta^n} : n \in \mathcal{N}_{\bmod e}\}$. Note that in general $|\mathcal{M}|$ is smaller than the size of $\mathcal{N}_{\bmod e}$, because in $\mathcal{N}_{\bmod e}$ exponents can belong to $\mathbb{F}_q$-conjugates which then yield the same polynomial multiple times.

The statement of the following lemma has been mentioned briefly in [GK23] but not been proven. We present its proof here.

Lemma 3.30. *Let $p_1, \dots, p_m$ be the distinct prime factors of $q-1$ and let $f \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree k and order e. Further, let $\beta \in \mathbb{F}_{q^k}$ be a root of f. Then the size $|\mathcal{M}|$ of the set $\mathcal{M} = \{m_{\beta^{n \bmod e}} : n = p_1^{i_1} \dots p_m^{i_m}, i_1, \dots, i_m \geq 0\}$ only depends on q and the order e of f.*

Proof. We can write $\mathcal{M} = \{m_{\beta^n} : n \in \mathcal{N}_{\bmod e}\}$, where $\mathcal{N}_{\bmod e}$ is defined as in (3.13). Obviously $|\mathcal{N}_{\bmod e}|$, the size of $\mathcal{N}_{\bmod e}$, only depends on q and e. Furthermore, two

distinct positive integers $n, n' \in \mathcal{N}_{\bmod e}$ yield the same polynomial $m_{\beta^n} = m_{\beta^{n'}} \in \mathcal{M}$ if and only if β^n and $\beta^{n'}$ are $\mathbb{F}_q$-conjugates. That is, if and only if there exists a positive integer $1 \leq j \leq k-1$ such that $\beta^n = \beta^{n' \cdot q^j}$. This is equivalent to $\beta^{n-n' \cdot q^j} = 1$ and thus also equivalent to $e \mid (n - n' \cdot q^j)$. The last statement only depends on q and e and Lemma 3.30 holds. □

We believe that it is not possible to give a closed formula for $|\mathcal{M}|$ in general since computing $|\mathcal{N}_{\bmod e}|$ is difficult. Indeed, it is related to determining the order of some prime numbers in $\mathbb{Z}_r^*$. In order to see this, suppose that $e = p_1^{v_1} \dots p_m^{v_m} \cdot r$ with $\gcd(q-1, r) = 1$ and $v_1, \dots, v_m \geq 0$. Then by the Chinese Remainder Theorem the ring $\mathbb{Z}_e$ is isomorphic to $\mathbb{Z}_{p_1^{v_1}} \times \dots \times \mathbb{Z}_{p_m^{v_m}} \times \mathbb{Z}_r$. To determine $|\mathcal{N}_{\bmod e}|$, in particular, we need to calculate the size of the multiplicative subgroup $\langle p_1, \dots, p_m \rangle$ in $\mathbb{Z}_r^*$.

Thus, it does not seem to be possible to determine a subset $\tilde{\mathcal{N}}$ of $\mathcal{N}$ such that every polynomial in $\mathcal{M}$ is contained exactly once in the sequence $\{m_{\beta^n}\}_{n \in \tilde{\mathcal{N}}}$. We are interested in finding a subset N of $\mathcal{N}$ such that the number of multiples in the sequence $\{m_{\beta^n}\}_{n \in N}$ is as small as possible. The following lemma shows that the periodic behavior of Construction 2 yields a naive upper bound on the exponents $i_1, \dots, i_m$:

Lemma 3.31. *Let $p_1, \dots, p_m$ be the distinct prime factors of $q-1$ and $f \in \mathbb{F}_q[X]$ a monic irreducible polynomial of degree k. Further, suppose that for every $1 \leq l \leq m$, Construction 2 applied to f for the prime p_l has a tail of length v_l and the orbit a length of s_l. Then*

$$\mathcal{M} \subseteq \{m_{\beta^n} : n = p_1^{i_1} \cdots p_m^{i_m}, i_1 \leq v_1 + s_1, \dots, i_m \leq v_m + s_m\}.$$

Proof. Let $n = p_1^{i_1} \cdots p_m^{i_m}$ such that $i_j > v_j + s_j$ for a positive integer $1 \leq j \leq m$ and $l = \min\{1 \leq j \leq m : i_j > v_j + s_j\}$. We prove that for $n_l = p_1^{i_1} \cdots p_l^{i_l}$ the polynomial $m_{\beta^{n_l}}$ is already contained in the set $\{m_{\beta^{n'}} : n' = p_1^{i'_1} \cdots p_l^{i'_l} : i'_j \leq v_j + s_j \text{ for all } 1 \leq j \leq l\}$. Then by induction, the statement of Lemma 3.31 is true.

The polynomial $\tilde{f} = m_{\beta^{n_{l-1}}}$, where $n_{l-1} = p_1^{i_1} \cdots p_{l-1}^{i_{l-1}}$ is of degree $\tilde{k} \leq k$ and order

$$\tilde{e} = \frac{e}{p_1^{\min\{\nu_{p_1}(e), i_1\}} \cdots p_{l-1}^{\min\{\nu_{p_{l-1}}(e), i_{l-1}\}}}.$$

The order $\tilde{e}$ divides the order e of f and satisfies $\nu_{p_l}(\tilde{e}) = \nu_{p_l}(e)$ because $p_1, \dots, p_m$ are distinct.

If we applied Construction 2 to $\tilde{f}$ for the prime p_l, we would also obtain a tail and an orbit. The length $\tilde{v}$ of the tail equals v_l, because $\nu_{p_l}(\tilde{e}) = \nu_{p_l}(e)$. Let $\tilde{r} = \frac{\tilde{e}}{\nu_{p_l}(\tilde{e})}$, then the length $\tilde{s}$ of the orbit is the smallest positive integer s such that

$$p_l^s \equiv q^j \pmod{\tilde{r}} \quad \text{for an integer } 0 \leq j \leq \tilde{k} - 1.$$

Thus, the positive integer $p_1^{i_1} \cdots p_l^{v_l + (i_l - v_l \pmod{\tilde{s}})}$ yields the same polynomial as the positive integer $n_l = p_1^{i_1} \cdots p_l^{i_l}$, namely $m_{\beta^{n_l}}$.

Recall that s_l is the smallest positive integer such that $p_l^{s_l} \equiv q^j \pmod{r}$ for an integer $0 \leq j \leq k-1$, where $r = \frac{e}{\nu_{p_l}(e)}$. From the fact that $\tilde{r}$ is a divisor of r follows that $s_l \geq \tilde{s}$. This implies that the exponent $v_l + (i_l - v_l \pmod{\tilde{s}})$ is less or equal than $v_l + s_l$ and the statement of Lemma 3.31 is true. □

Remark 3.32. If the order e of the initial polynomial $f = m_\beta$ is known, the values v_j and s_j can be determined directly with Theorem 3.26.

Using the ideas from the proof of Lemma 3.31, we give a more precise definition of the tails and orbits obtained from Construction 2:

Let $f = m_\beta$ be a monic irreducible polynomial over $\mathbb{F}_q$ of degree k and let $\beta \in \mathbb{F}_{q^k}$ be a root of f. Furthermore, let n be a positive integer and p a prime dividing $q-1$. If we apply Construction 2 to the polynomial m_{β^n} for the prime p, we obtain a tail of length $\nu_p(e)$ and an orbit whose length is a divisor of $\mathrm{ord}_{\frac{e}{\nu_p(e)}}(p)$, where $e = \mathrm{ord}(m_{\beta^n})$. We call this tail *the tail of n for p* and this orbit *the orbit of n for p*.

Let $p_1, \ldots, p_m$ be the distinct prime factors of $q-1$ and for all $1 \leq j \leq m$ let v_j be the length of the tail of 1 for p_j and s_j the length of the orbit of 1 for p_j. Moreover, let $1 \leq l \leq m$ and n be a positive integer such that $n = p_1^{i_1} \ldots p_{l-1}^{i_{l-1}} \cdot p_{l+1}^{i_{l+1}} \cdots p_m^{i_m}$ for $i_1, \ldots, i_{l-1}, i_{l+1}, \ldots, i_m \in \mathbb{N} \cup \{0\}$. Then Lemma 3.31 and its proof imply that the length of the tail of n for p_l equals v_l and the length of the orbit of n for p_l is less than or equal to s_l.

Example 3.33. Let $q = 31$ and we consider the monic irreducible polynomial $f = X^4 + 30X^3 + 9X^2 + 15X + 27 \in \mathbb{F}_{31}[X]$. The order of f equals $e = 307\,840 = 2^7 \cdot 5 \cdot 13 \cdot 37$ and Construction 2 applied for the distinct prime factors $p_1 = 2, p_2 = 3, p_3 = 5$ of 30 yields :

Table 3.1: Construction 2 for $p_1 = 2$, $p_2 = 3$ and $p_3 = 5$

p_j	$v_j = \nu_p(\mathrm{ord}(f))$	orbit length s_j
2	7	36
3	0	288
5	1	288

Then Lemma 3.31 states that for Construction 1 we only need to consider positive integers $n = 2^{i_1} \cdot 3^{i_2} \cdot 5^{i_3}$ such that $i_1 \leq 7 + 36 = 43,\ i_2 \leq 0 + 288,\ i_3 \leq 1 + 288 = 289$. However, in general these naive upper bounds are not reached as the following table shows for i_2. It specifies the lengths of the orbits of $n = 2^{i_1} \cdot 5^0 = 2^{i_1}$, $i_1 \geq 0$, for $p_2 = 3$. The length of the tail of $n = 2^{i_1}$ for $p_2 = 3$ equals $v_2 = 0$ for every $i_1 \geq 0$. The bound $v_2 + s_2 = 288$ is reached in only one of these orbits:

Table 3.2: Orbits of $n = 2^{i_1}$ for $p_2 = 3$

n	orbit length	orbit order
2^0	288	$2^7 \cdot 5 \cdot 13 \cdot 37$
2^1	144	$2^6 \cdot 5 \cdot 13 \cdot 37$
2^2	72	$2^5 \cdot 5 \cdot 13 \cdot 37$
2^3	36	$2^4 \cdot 5 \cdot 13 \cdot 37$
2^4	36	$2^3 \cdot 5 \cdot 13 \cdot 37$
2^5	36	$2^2 \cdot 5 \cdot 13 \cdot 37$
2^6	36	$2 \cdot 5 \cdot 13 \cdot 37$
2^7	36	$5 \cdot 13 \cdot 37$
2^8	36	$5 \cdot 13 \cdot 37$

2^9	36	$5 \cdot 13 \cdot 37$
2^{10}	36	$5 \cdot 13 \cdot 37$
2^{11}	36	$5 \cdot 13 \cdot 37$
2^{12}	36	$5 \cdot 13 \cdot 37$

Table 3.2 ends with $i_1 = 12$, because the polynomials in the orbit of 2^{13} for $p_2 = 3$ are the same as in the orbit of 2^7 for $p_2 = 3$. In particular, the distinct polynomials obtained from Construction 2 with the positive integers $n = 2^i$ for $13 \leq i \leq 43$ are already contained in the orbits shown in the table above.

Orbits consisting of the same polynomials can also be found in the orbits of the positive integers $n = 2^{i_1} \cdot 3^{i_2}$, $i_1, i_2 \geq 0$, for $p_3 = 5$. For all $i_1, i_2 \geq 0$ the tail of n for $p_3 = 5$ has length $1 = v_3$. The following table specifies the lengths of the orbits for $p_3 = 5$ and the order of the orbit polynomials. Only three of the orbits are of length $s_3 = 288$.

Table 3.3: Orbits of $X^4 + 30X^3 + 9X^2 + 15X + 27 \in \mathbb{F}_{31}[X]$

n	orbit length	orbit order	same orbit as
1	288	$2^7 \cdot 13 \cdot 37$	
2^1	144	$2^6 \cdot 13 \cdot 37$	
2^2	72	$2^5 \cdot 13 \cdot 37$	
2^3	36	$2^4 \cdot 13 \cdot 37$	
2^4	18	$2^3 \cdot 13 \cdot 37$	
2^5	18	$2^2 \cdot 13 \cdot 37$	
2^6	18	$2 \cdot 13 \cdot 37$	
2^7	18	$13 \cdot 37$	
2^8	18	$13 \cdot 37$	
2^9	18	$13 \cdot 37$	
2^{10}			2^7
3^1	288	$2^7 \cdot 13 \cdot 37$	
3^2	288	$2^7 \cdot 13 \cdot 37$	
3^3			1
$2^1 \cdot 3^1$	144	$2^6 \cdot 13 \cdot 37$	
$2^2 \cdot 3^1$	72	$2^5 \cdot 13 \cdot 37$	
$2^3 \cdot 3^1$	36	$2^4 \cdot 13 \cdot 37$	
$2^4 \cdot 3^1$	18	$2^3 \cdot 13 \cdot 37$	
$2^5 \cdot 3^1$	18	$2^2 \cdot 13 \cdot 37$	
$2^6 \cdot 3^1$	18	$2 \cdot 13 \cdot 37$	
$2^7 \cdot 3^1$			2^8
$2^1 \cdot 3^2$	144	$2^6 \cdot 13 \cdot 37$	

$2^2 \cdot 3^2$	72	$2^5 \cdot 13 \cdot 37$	
$2^3 \cdot 3^2$	36	$2^4 \cdot 13 \cdot 37$	
$2^4 \cdot 3^2$	18	$2^3 \cdot 13 \cdot 37$	
$2^5 \cdot 3^2$	18	$2^2 \cdot 13 \cdot 37$	
$2^6 \cdot 3^2$	18	$2 \cdot 13 \cdot 37$	
$2^7 \cdot 3^2$			2^9

The exponents (i_1, i_2) in

$$\{(0,3),(1,3),(0,4),(2,3),(1,4),(0,5),(3,3),(2,4),(1,5),$$
$$(0,6),(4,3),(3,4),(2,5),(1,6),(0,7),(7,1),(5,3),(4,4),(3,5),\ldots,$$
$$(1,36),(0,37),(1,37),(0,38),(1,38),\ldots,$$
$$(0,144),(0,145),(0,146),\ldots,(0,287)\}$$

yield one new tail polynomial each but their orbit is the same as one in the table above.

The following lemma shows that orbits of different integers n and n' for the same prime p are either completely distinct or equal.

Lemma 3.34. *Let f be a monic irreducible polynomial over $\mathbb{F}_q$ of degree k and $\beta \in \mathbb{F}_{q^k}$ a root of f. Further, let n, n' be two positive integers and p be a prime factor of $q-1$.*

Then the orbits of n and n' for p contain either exactly the same or no common polynomials. If the orbits contain the same polynomials, then these appear in the same or a shifted order.

Proof. Let $e = \mathrm{ord}(m_{\beta^n})$ and $e' = \mathrm{ord}(m_{\beta^{n'}})$. Note that for any polynomial g in the tail or orbit of n for p or n' for p holds that its successor is the polynomial m_{γ^p}, where γ is a root of g. Thus, we can define a successor mapping on the polynomials and if two polynomials g and g' are equal, all of their successors are also equal.

If one polynomial g of the orbits of n and n' for p is equal, both orbits contain all polynomials of the form $\{m_{\gamma^{p^i}} : i \in \mathbb{N}_0\}$, where γ is a root of g, because of the periodic behavior of Construction 2 . Thus, the whole orbit is equal. □

Based on the fact that with Lemma 3.34 orbits for one fixed prime factor p of $q-1$ are either equal or distinct, we define the following algorithm, where we fix the prime factor p_m of $q-1$ and compute only tails and orbits for p_m:

Algorithm 3. Let $p_1, \ldots, p_m$ be the distinct prime factors of $q-1$ and $f \in \mathbb{F}_q[X]$, $f \neq X$, a monic irreducible polynomial of degree k.
Suppose that for every $1 \leq j \leq m$, Construction 2 applied to f for the prime p_j has a tail of length v_j and an orbit of length s_j.

3.1. Set $\mathcal{T} := \{\}, \mathcal{O} := \{\}$ and $\mathcal{M} := \{\}$.
3.2. Choose $0 \leq i_j \leq v_j + s_j$ for $1 \leq j \leq m-1$.
3.3. Set $n := p_1^{i_1} \cdots p_{m-1}^{i_{m-1}}$.
3.4. Compute the polynomial m_{β^n} and add it to $\mathcal{M}$.
3.5. (Tail) For $1 \leq i_m < v_m$:
Compute $m_{\beta^{n \cdot p_m^{i_m}}}$ from $m_{\beta^{n \cdot p_m^{i_m-1}}}$ with Construction 1.
If $m_{\beta^{n \cdot p_m^{i_m}}}$ is already contained in $\mathcal{T}$, then stop.
Otherwise add $m_{\beta^{n \cdot p_m^{i_m}}}$ to $\mathcal{T}$ and to $\mathcal{M}$.
3.6. (Orbit) For $v_m \leq i_m \leq v_m + s_m$:

> Compute $m_{\beta^{n\cdot p_m^{i_m}}}$ from $m_{\beta^{n\cdot p_m^{i_m-1}}}$ with Construction 1.
> **If** $m_{\beta^{n\cdot p_m^{i_m}}}$ is already contained in $\mathcal{O}$, then stop.
> **If** $m_{\beta^{n\cdot p_m^{i_m}}} = m_{\beta^{n\cdot p_m^{v_m}}}$, then add $m_{\beta^{n\cdot p_m^{v_m}}}$ to $\mathcal{O}$ as orbit representative and all orbit polynomials $\{m_{\beta^{n\cdot p_m^{i}}} : v_m \leq i < i_m\}$ to $\mathcal{M}$.

Let

$$\mathcal{S} = \{m_{\beta^n} : n = p_1^{i_1} \cdots p_{m-1}^{i_{m-1}}, 0 \leq i_j \leq v_j + s_j \text{ for all } 1 \leq j \leq m-1\}$$

be the set of all starting polynomials of Algorithm 3. Then from Lemma 3.31 and Lemma 3.34 follows that the union of all $\mathcal{M}$ for $m_{\beta^n} \in \mathcal{S}$ equals the set $\mathcal{M}$ as defined in (3.12).

We present an implementation of Algorithm 3 in the class `Construction1` from the module B.12 `ConstructionClasses.sage`. In Appendix B.5.1 we explain in detail how to use our implementation.

In our implementation of Algorithm 3 steps **3.2.** to **3.4.** are implemented so that every polynomial $m_{\beta^n} \in \mathcal{S}$ is obtained from another $m_{\beta^{n'}} \in \mathcal{S}$ with exactly one application of `construction_mbetap` (see Source Code 3.1). This is realized as follows: Every polynomial $m_{\beta^n} \in \mathcal{S}$ is stored together with the list $[i_1, \ldots, i_{m-1}]$ of its exponents and a pointer to an index $1 \leq j \leq m-1$ of the list. Then the polynomials $m_{\beta^{n\cdot p_l}}$ for $j \leq l \leq m-1$ are computed from m_{β^n}. They are stored together with the list $[i_1, \ldots, i_l + 1, \ldots, i_m]$ and the pointer l. The same procedure is applied recursively to each of them. The use of the pointer here is crucial. For example the polynomial $m_{\beta^{2^2\cdot 3^2}}$ from Table 3.3 could be computed from $m_{\beta^{2^2\cdot 3^1}}$ and from $m_{\beta^{2^1\cdot 3^2}}$ with one application of `construction_mbetap`. However, in this program the pointer can only "move" to the right and $m_{\beta^{2^1\cdot 3^2}}$ would not be considered a predecessor of $m_{\beta^{2^2\cdot 3^2}}$. With this procedure all computations of step **3.4.** are done exactly once.

The computation of the upper bounds $v_l + s_l$ for $1 \leq i \leq m-1$ are not done beforehand, but are included in the program itself. They are stored and updated in the dictionary `max_exponents` (see lines 203ff. in Source Code B.12).

The statement of Lemma 3.34 is applied in step **3.6.**. Here we use the fact that either the orbit of n for p_m is the same as an orbit that has been computed before or all orbit polynomials are new polynomials.

In [GK23] we had a false intuition and claimed that an analogue of Lemma 3.34 would hold for tails. More precisely, we claimed that two integers n and n' either have a completely distinct or an equal tail. This statement is not true as the following example shows.

Example 3.35. Let $q = 31$ and we consider again the monic irreducible polynomial $f = X^4 + 30X^3 + 9X^2 + 15X + 27 \in \mathbb{F}_{31}[X]$. The following table gives the tails of $n = 3 \cdot 5^5$ and $n' = 3^4 \cdot 5^2$ for $p = 2$:

Table 3.4: Counterexample for completely equal or distinct tails

i	$m_{\beta^{n\cdot 2^i}}$	$m_{\beta^{n'\cdot 2^i}}$
0	$X^4 + 17X^3 + 7X^2 + 25X + 30$	$X^4 + 16X^3 + 29X^2 + 20X + 30$
1	$X^4 + 4X^3 + 3X^2 + 12X + 1$	$X^4 + 19X^3 + 13X^2 + 7X + 1$
2	$X^4 + 21X^3 + 8X^2 + 17X + 1$	$X^4 + 6X^3 + 29X^2 + 8X + 1$
3	$X^4 + 9X^3 + 3X^2 + 6X + 1$	$X^4 + 22X^3 + 3X^2 + 25X + 1$

4	$X^4 + 18X^3 + 27X^2 + X + 1$	$X^4 + 18X^3 + 27X^2 + X + 1$
5	$X^4 + 9X^3 + 13X^2 + 22X + 1$	$X^4 + 9X^3 + 13X^2 + 22X + 1$
6	$X^4 + 7X^3 + 23X^2 + 7X + 1$	$X^4 + 7X^3 + 23X^2 + 7X + 1$

Even though the first four polynomials for $0 \leq i \leq 3$ are distinct, the fourth polynomial and all its successors are equal. Thus, the statement from [GK23] is false. However, the presented numerical results were correct. Since the tails were all of length 1, they were either equal or completely distinct.

We can use the fact that all successors are equal, if one polynomial in the computations for two integers n and n' is equal, to our advantage. The following lemma shows that this fact reduces the number of orbits that need to be considered by our implementation.

Lemma 3.36. *Let f be a monic irreducible polynomial over $\mathbb{F}_q$ of degree k and $\beta \in \mathbb{F}_{q^k}$ a root of f. Further, let p be a prime factor of $q-1$ and n, n' be two positive integers such that for $0 \leq i, i'$ the two polynomials $m_{\beta^{n \cdot p^i}}$ and $m_{\beta^{n' \cdot p^{i'}}}$ are equal.*

Then for any positive integer $m \in \mathbb{N}$, also the polynomials $m_{\beta^{m \cdot n \cdot p^i}}$ and $m_{\beta^{m \cdot n' \cdot p^{i'}}}$ are equal.

In particular, if the orbits of n and n' for p are equal, also the orbits of $m \cdot n$ and $m \cdot n'$ are equal for any positive integer m.

Proof. Since $m_{\beta^{n \cdot p^i}} = m_{\beta^{n' \cdot p^{i'}}}$, there exists an integer $0 \leq j \leq \deg(m_{\beta^{n \cdot p^i}})$ such that $\beta^{n \cdot p^i} = \beta^{n' \cdot p^{i'} \cdot q^j}$. For any positive integer $m \in \mathbb{N}$ follows that $\beta^{m \cdot n \cdot p^i} = \beta^{m \cdot n' \cdot p^{i'} \cdot q^j}$ and the polynomials $m_{\beta^{m \cdot n \cdot p^i}}$ and $m_{\beta^{m \cdot n' \cdot p^{i'}}}$ are equal.

If the orbits of n and n' are the same for p, then there exist positive integers $i, i' \geq \nu_p(\mathrm{ord}(f))$ such that $m_{\beta^{n \cdot p^i}} = m_{\beta^{n' \cdot p^{i'}}}$. Then for any positive integer m, the polynomials $m_{\beta^{m \cdot n \cdot p^i}}$ and $m_{\beta^{m \cdot n' \cdot p^{i'}}}$ are also equal. Since $i, i' \geq \nu_p(\mathrm{ord}(f))$, these polynomials are in the orbits of $m \cdot n$ and $m \cdot n'$. Because of the periodic behavior of Construction 2 the whole orbits are equal. □

Note that Lemma 3.36 explains Table 3.3. We see that the orbits of 2^7 and 2^{10} are equal, then with Lemma 3.36 follows directly that the orbits of 2^{11} and 2^8 are equal. The orbit of 2^8 has already been computed and does not need to be computed again. Moreover, for any $i \geq 10$ the orbit of 2^i is the same as the orbit of 2^{7+r}, where $i = 7 + (10-7) \cdot m + r$ and $0 \leq r < (10-7)$. Furthermore, from Lemma 3.36 follows that for any positive integer n the orbits of $2^7 \cdot n$ and $2^{10} \cdot n$ are equal. Thus, 10 is an upper bound on the exponent i_1 of 2 for the computation of new orbits.

In the implementation `Construction1` in `ConstructionClasses.sage` the upper bounds on the exponents for the construction of new orbits are stored and updated in the dictionary `new_orbit_exponents` (see lines 249ff. in Source Code B.12).

The following lemma generalizes the ideas used for the `new_orbit_exponents`:

Lemma 3.37. *Let f be a monic irreducible polynomial over $\mathbb{F}_q$ of degree k and $\beta \in \mathbb{F}_{q^k}$ a root of f. Further, let $p_1, \ldots, p_m$ be the distinct prime factors of $q-1$ and let $n' = p_1^{i_1'} \cdots p_{m-1}^{i_{m-1}'}$ and $n'' = p_1^{i_1''} \cdots p_{m-1}^{i_{m-1}''}$ be positive integers such that $i_l' \leq i_l''$ for all $1 \leq l \leq m-1$. Suppose that the orbits of n' and n'' for p_m are equal and suppose that the positive integer n is of the form*

$$n = p_1^{i_1} \cdots p_{m-1}^{i_{m-1}} \text{ with } i_l = i_l' + (i_l'' - i_l') \cdot d_l + r_l \text{ for all } 1 \leq l \leq m-1,$$

where $d_l \geq 0$ and $0 \leq r_l \leq (i_l'' - i_l')$ for all $1 \leq l \leq m-1$. Then the orbit of n for p_m is equal to the orbit of an integer $\tilde{n} \in N(n', n'')$ for p_m, where

$$N(n', n'') := \left\{ \tilde{n} = p_1^{\tilde{i}_1} \cdots p_{m-1}^{\tilde{i}_{m-1}} \mid i_l' \leq \tilde{i}_l \leq i_l'' \text{ for all } 1 \leq l \leq m-1 \right\}.$$

Proof. From Lemma 3.36 follows that the orbit of

$$p_1^{i_1'+(i_1''-i_1)\cdot d_1} \cdots p_{m-1}^{i_{m-1}'+(i_{m-1}''-i_{m-1}')\cdot d_{m-1}}$$

is equal to the orbit of n' and n''. Consequently, the orbit of n is equal to the orbit of $\tilde{n} := p_1^{i_1'+r_1} \cdots p_{m-1}^{i_{m-1}'+r_{m-1}}$ and $\tilde{n}$ is an element of the set $N(n', n'')$. □

The statement of Lemma 3.37 could be used to further reduce the number of duplicates during the computation of step **3.6.** in Algorithm 3. All pairs (n', n'') whose orbits for p_m are equal would have to be stored and for every positive integer n the program needs to check whether $n \in N(n', n'')$ for any of the stored pairs (n', n''). It is not clear whether this actually speeds up the computation time of the program.

In Chapter 1 we mentioned that different applications call for irreducible polynomials with different properties. Especially the construction of large finite fields calls for irreducible polynomials of a low weight. When working simultaneously in a finite field $\mathbb{F}_{q^k}$ as an extension of $\mathbb{F}_q$ and also as a vector space over $\mathbb{F}_q$ it might be useful to select a normal polynomial and this concept was extended to so-called k-normal polynomials in [Huc+13]. The class `RichPolynomial` contains the functions `get_weight` and `get_knormal`. Using these functions the implementation B.12 of Algorithm 3 yields an overview table of the weight and k-normality of the constructed polynomials as output.

Example 3.38. We consider the polynomials

$$\begin{aligned} f_1 &= X^8 + X^5 + X^3 + X^2 + a, \\ f_2 &= X^9 + (a^2+a)X^8 + (a^3+a^2)X^7 + aX^6 + X^5 + (a^3+a^2+a)X^4 \\ &\quad + (a^2+a+1)X^3 + a^2X^2 + a^3X + a^3 + a^2 + a \end{aligned}$$

over $\mathbb{F}_{16} = \mathbb{F}(a)$, where a is a root of the monic irreducible polynomial $X^4 + X + 1$ over $\mathbb{F}_2$.

The polynomial f_1 is primitive and has order $4\,294\,967\,295 = 3 \cdot 5 \cdot 17 \cdot 257 \cdot 65\,537$. Construction 1 with f_1 as initial polynomial yields 1 114 113 monic irreducible polynomials of degree 8. Considering the orbits of the positive integers $n = 3^j$, $j \geq 0$, for 5 there are 33 orbits of 32 768 polynomials each. The orbit of $n = 1$ for 5 contains 32 768 of the 67 108 864 monic irreducible polynomials of order $3 \cdot 17 \cdot 257 \cdot 65537$ over $\mathbb{F}_{16}$ and the other 32 orbits of $n = 3^j$ for 5, $1 \leq j \leq 32$, yield 1 048 576 of the 33 554 432 monic irreducible polynomials of order $17 \cdot 257 \cdot 65\,537$ over $\mathbb{F}_{16}$. f_1 has 5 non-zero coefficients and yields a weight distribution of $4^6 5^{384} 6^{7225} 7^{65997} 8^{331084} 9^{709417}$, which means that there exist 6 polynomials with smaller weight and 384 polynomials with the same weight. Hence, from these we could try to choose polynomials with other required properties that our initial polynomial might lack.

The polynomial f_2 has order $68\,719\,476\,735 = 3^3 \cdot 5 \cdot 7 \cdot 13 \cdot 19 \cdot 37 \cdot 73 \cdot 109$ and yields 4 644 monic irreducible polynomials over $\mathbb{F}_{16}$ of degree 9 with the weight distribution $6^2 7^{47} 8^{373} 9^{1401} 10^{2821}$. Even though the number of constructed polynomials is not very large, we could find polynomials of weight 6, 7 and 8. Considering the orbits of the positive integers $n = 3^j$, $j \geq 0$, for 5 there are 21 orbits of 216 polynomials each and the construction yields 3 888 of the 40 310 784 polynomials of order $509\,033\,161 = 7 \cdot 13 \cdot 19 \cdot 37 \cdot 73 \cdot 109$.

Tables 3.5 and 3.6 show that Construction 1 also yields a large number of k-normal polynomials for small values of k which could be used for respective applications. Since the number of k-polynomials decreases with k increasing, this distribution of k-normality is to be expected (see [Huc+13]).

Table 3.5: Weight and k-normality distribution for f_1

Weight	Total	0-normal	1-normal	2-normal	3-normal
4	6	1	5	0	0
5	384	139	240	5	0
6	7 225	4 160	2 927	136	2
7	65 997	47 088	17 746	1 119	44
8	331 084	283 554	44 713	2 625	192
9	709 417	709 417	0	0	0

Table 3.6: Weight and k-normality distribution for f_2

Weight	Total	0-normal	1-normal	2-normal	3-normal	4-normal
6	2	1	1	0	0	0
7	47	28	15	4	0	0
8	373	256	102	14	1	0
9	1 401	1 091	290	18	0	2
10	2 821	2 475	339	5	2	0

3.5 Future work

Construction AD, Construction KK and our new construction Construction 1 all were a reversal of the classical composition method for the rational function $Q = \frac{X^n}{1}$ and selected positive integers n. Gohar M. Kyureghyan suggested that the method could also be applied for arbitrary rational functions $Q = \frac{g}{h} \in \mathbb{F}_q(X)$.

For an arbitrary rational function $Q = \frac{g}{h} \in \mathbb{F}_q(X)$ the reverse composition method can be described as follows:

Step 1. Choose an irreducible polynomial $f_1 \in \mathbb{F}_q[X]$.

Step 2. Compute other irreducible polynomials $f_2, \ldots, f_m \in \mathbb{F}_q[X]$ from f_1 such that

$$f_1 \cdot f_2 \cdots f_m = f^Q$$

for a polynomial $f \in \mathbb{F}_q[X]$.

Step 3. Extract f from the composition f^Q.

Step 4. Either f is irreducible or an irreducible polynomial can easily be computed from f.

For this construction to work, the rational function Q needs to satisfy the following conditions:

(i) The factorization of f^Q into irreducible factors $f_1 \cdots f_m$ over $\mathbb{F}_q$ is known for any polynomial $f \in \mathbb{F}_q[X]$.

(ii) When one factor f_1 of the factorization is given, the other factors $f_2, \ldots, f_m$ can be easily computed.

(iii) The polynomial f can easily be retrieved from the composition f^Q.

(iv) It is easy to decide whether f is irreducible or not, or an irreducible polynomial can easily be computed from f.

The properties of the polynomials obtained from the reverse composition method depend on the rational function Q. For example, our construction with $Q = \frac{X^n}{1}$ yielded only irreducible polynomials of the same or a lower order. Other choices of Q will yield other interesting irreducible polynomials with different properties.

Thus, it might be interesting and useful to study other factorizations of compositions f^Q and to derive methods for conditions (ii), (iii) and (iv) for them.

Bibliography

[ADM18] Mahmood Alizadeh, Mohammad Reza Darafsheh, and Saeid Mehrabi. "On the k-normal elements and polynomials over finite fields". In: *Italian Journal of Pure and Applied Mathematics* 39 (2018), pp. 451–464.

[AHJ10] Nabihah Ahmad, Rezaul Hasan, and Warsuzarina Mat Jubadi. "Design of AES S-Box using combinational logic optimization". In: *2010 IEEE Symposium on Industrial Electronics and Applications (ISIEA)*. IEEE. 2010, pp. 696–699.

[AK11] Sergey Abrahamyan and Melsik Kyureghyan. "A recurrent method for constructing irreducible polynomials over finite fields". In: *International Workshop on Computer Algebra in Scientific Computing*. Springer. 2011, pp. 1–9.

[Alb56] A. Adrian Albert. *Fundamental Concepts of Higher Algebra*. Chicago: University of Chicago Press, 1956.

[Als18] Zynab Alshareef. "Composed Product and Factorization of Cyclotomic Polynomials over Finite Fields". PhD thesis. Carleton University, 2018.

[AM15] Mahmood Alizadeh and Saeid Mehrabi. "Construction of self-reciprocal normal polynomials over finite fields of even characteristic". In: *Turkish Journal of Mathematics* 39.2 (2015), pp. 259–267.

[AP99] S.K. Arora and Manju Pruthi. "Minimal cyclic codes of length $2p^n$". In: *Finite fields and their applications* 5.2 (1999), pp. 177–187.

[Ava+20] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. *CRYSTALS-Kyber*. `https://pq-crystals.org/kyber/resources.shtml`. 2020.

[Ber15] Elwyn R. Berlekamp. *Algebraic coding theory (revised edition)*. World Scientific, 2015.

[Ber67] Elwyn R. Berlekamp. "Factoring polynomials over finite fields". In: *Bell System Technical Journal* 46.8 (1967), pp. 1853–1859.

[Ber70] Elwyn R. Berlekamp. "Factoring polynomials over large finite fields". In: *Mathematics of Computation* 24.111 (1970), pp. 713–735.

[Bey77] F. Rudolf Beyl. "Cyclic subgroups of the prime residue group". In: *The American Mathematical Monthly* 84.1 (1977), pp. 46–48.

[BGM93] Ian F. Blake, Shuhong Gao, and Ronald C. Mullin. "Explicit factorization of $X^{2^k}+1$ over $\mathbb{F}_p$ with Prime $p \equiv 3 \mod 4$". In: *Applicable Algebra in Engineering, Communication and Computing* 4.2 (1993), pp. 89–94.

[BR03] Gurmeet K. Bakshi and Madhu Raka. "Minimal cyclic codes of length $p^n q$". In: *Finite Fields and Their Applications* 9.4 (2003), pp. 432–448.

[But55] M.C.R. Butler. "The irreducible factors of $f(x^m)$ over a finite field". In: *Journal of the London Mathematical Society* 1.4 (1955), pp. 480–482.

[Can05] David Canright. "A very compact S-box for AES". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 2005, pp. 441–455.

[Cas+91] Guy Castagnoli, James L. Massey, Philipp A. Schoeller, and Niklaus Von Seemann. "On repeated-root cyclic codes". In: *IEEE Transactions on Information Theory* 37.2 (1991), pp. 337–342.

[CB08] David Canright and Lejla Batina. "A very compact "perfectly masked" S-box for AES". In: *Applied Cryptography and Network Security: 6th International Conference, ACNS 2008, New York, NY, USA, June 3-6, 2008. Proceedings 6*. Springer. 2008, pp. 446–459.

[CDL14] Bocong Chen, Hai Q. Dinh, and Hongwei Liu. "Repeated-root constacyclic codes of length lp^s and their duals". In: *Discrete Applied Mathematics* 177 (2014), pp. 60–70.

[CDL15] Bocong Chen, Hai Q. Dinh, and Hongwei Liu. "Repeated-root constacyclic codes of length $2l^m p^n$". In: *Finite fields and their applications* 33 (2015), pp. 137–159.

[Cha97] Robin Chapman. "Completely normal elements in iterated quadratic extensions of finite fields". In: *Finite Fields and Their Applications* 3.1 (1997), pp. 1–10.

[Che+12] Bocong Chen, Yun Fan, Liren Lin, and Hongwei Liu. "Constacyclic codes over finite fields". In: *Finite Fields and Their Applications* 18.6 (2012), pp. 1217–1231.

[CLT13] Bocong Chen, Liangchen Li, and Rouziwanguli Tuerhong. "Explicit factorization of $X^{2^m p^n} - 1$ over a finite field". In: *Finite Fields and Their Applications* 24 (2013), pp. 95–104.

[CLZ15] Bocong Chen, Hongwei Liu, and Guanghui Zhang. "A class of minimal cyclic codes over finite fields". In: *Designs, Codes and Cryptography* 74.2 (2015), pp. 285–300.

[Coh69] Stephen D. Cohen. "On irreducible polynomials of certain types in finite fields". In: *Mathematical Proceedings of the Cambridge Philosophical Society*. Vol. 66. 2. Cambridge University Press. 1969, pp. 335–344.

[Coh92] Stephen D. Cohen. "The explicit construction of irreducible polynomials over finite fields". In: *Designs, Codes and Cryptography* 2 (1992), pp. 169–174.

[Cui+11] Jie Cui, Liusheng Huang, Hong Zhong, Chinchen Chang, and Wei Yang. "An improved AES S-box and its performance analysis". In: *International Journal of Innovative Computing, Information and Control* 7.5 (2011), pp. 2291–2302.

[CZ81] David G. Cantor and Hans Zassenhaus. "A new algorithm for factoring polynomials over finite fields". In: *Mathematics of Computation* 36.154 (1981), pp. 587–592.

[Day65] David E. Daykin. "Generation of irreducible polynomials over a finite field". In: *The American Mathematical Monthly* 72.6 (1965), pp. 646–648.

[Din+19] Hai Q. Dinh, Xiaoqiang Wang, Hongwei Liu, and Songsak Sriboonchitta. "On the Hamming distances of repeated-root constacyclic codes of length $4p^s$". In: *Discrete Mathematics* 342.5 (2019), pp. 1456–1470.

[Din+21] Hai Q. Dinh, Xiaoqiang Wang, Hongwei Liu, and Woraphon Yamaka. "Hamming distances of constacyclic codes of length $3p^s$ and optimal codes with respect to the Griesmer and Singleton bounds". In: *Finite Fields and Their Applications* 70 (2021), p. 101794.

[Din08] Hai Q. Dinh. "On the linear ordering of some classes of negacyclic and cyclic codes and their distance distributions". In: *Finite Fields and Their Applications* 14.1 (2008), pp. 22–40.

[Din12] Hai Q. Dinh. "Repeated-root constacyclic codes of length $2p^s$". In: *Finite Fields and Their Applications* 18.1 (2012), pp. 133–143.

[Din13a] Hai Q. Dinh. "On repeated-root constacyclic codes of length $4p^s$". In: *Asian-European Journal of Mathematics* 6.02 (2013), p. 1350020.

[Din13b] Hai Q. Dinh. "Structure of repeated-root constacyclic codes of length $3p^s$ and their duals". In: *Discrete Mathematics* 313.9 (2013), pp. 983–991.

[DR19] Hai Q. Dinh and Saroj Rani. "Structure of some classes of repeated-root constacyclic codes of length $2^k l^m p^n$". In: *Discrete Mathematics* 342.12 (2019), p. 111609.

[DWS20] Hai Q. Dinh, Xiaoqiang Wang, and Jirakom Sirisrisakulchai. "On the Hamming Distances of Constacyclic Codes of Length $5p^s$". In: *IEEE Access* 8 (2020), pp. 46242–46254.

[FY07] Robert W. Fitzgerald and Joseph L. Yucas. "Explicit factorizations of Cyclotomic and Dickson polynomials over finite fields". In: *Arithmetic of Finite Fields: First International Workshop, WAIFI 2007, Madrid, Spain, June 21-22, 2007. Proceedings 1*. Springer. 2007, pp. 1–10.

[Gao93] Shuhong Gao. "Normal bases over finite fields." PhD thesis. University of Waterloo Waterloo, Canada, 1993.

[Gau76] Carl Friedrich Gauss. *Analysis residuorum: Caput octavum. Disquisitiones generales de con- gruentiis, Werke*. Vol. 2. Göttingen: Konigl. Gesellschaft der Wissenschaften, 1876, pp. 212–240.

[Gau89] Carl Friedrich Gauss. *Untersuchungen über Hohere Arithmetik (H. Maser, ed.)* Berlin: Springer, 1889, pp. 602–629.

[GK23] Anna-Maurin Graner and Gohar M. Kyureghyan. "Constructing irreducible polynomials recursively with a reverse composition method". In: *Designs, Codes and Cryptography* (2023).

[Gra23] Anna-Maurin Graner. *Closed formulas for the factorization of $X^n - 1$, the n-the cyclotomic polynomial, $X^n - a$ and $f(X^n)$ over a finite field for arbitrary positive integers n*. 2023. arXiv: `2306.11183 [math.NT]`.

[GYL21] Ying Gao, Qin Yue, and Fengwei Li. "The minimum Hamming distances of repeated-root cyclic codes of length $6p^s$ and their MDS codes". In: *Journal of Applied Mathematics and Computing* 65.1-2 (2021), pp. 107–123.

[Ham50] Richard W. Hamming. "Error detecting and error correcting codes". In: *The Bell system technical journal* 29.2 (1950), pp. 147–160.

[HJ12] Razi Hosseinkhani and H. Haj Seyyed Javadi. "Using cipher key to generate dynamic S-box in AES cipher system". In: *International Journal of Computer Science and Security (IJCSS)* 6.1 (2012), pp. 19–28.

[Huc+13] Sophie Huczynska, Gary L. Mullen, Daniel Panario, and David Thomson. "Existence and properties of k-normal elements over finite fields". In: *Finite Fields and Their Applications* 24 (2013), pp. 170–183.

[Hug00] Garry Hughes. "Constacyclic codes, cocycles and a $u + v|u - v$ construction". In: *IEEE Transactions on Information Theory* 46.2 (2000), pp. 674–680.

[KK09] Kazys Kazlauskas and Jaunius Kazlauskas. "Key-dependent S-box generation in AES block cipher system". In: *Informatica* 20.1 (2009), pp. 23–34.

[KK11] Melsik K. Kyuregyan and Gohar M. Kyureghyan. "Irreducible Compositions of Polynomials over Finite Fields". In: *Designs Codes Cryptography* 61.3 (2011), pp. 301–314.

[KK20] Gohar M. Kyureghyan and Melsik K. Kyureghyan. *A recurrent construction of irreducible polynomials of fixed degree over finite fields*. 2020. arXiv: `2005.05285 [math.NT]`.

[KK22] Gohar M. Kyureghyan and Melsik K. Kyureghyan. "A recurrent construction of irreducible polynomials of fixed degree over finite fields". In: *Applicable Algebra in Engineering, Communication and Computing* 33.2 (2022), pp. 163–171.

[KR08] G.N. Krishnamurthy and V. Ramaswamy. "Making AES stronger: AES with key dependent S-box". In: *IJCSNS International Journal of Computer Science and Network Security* 8.9 (2008), pp. 388–398.

[KS20] Ryul Kim and Hyang-Sim Son. "Recursive constructions of k-normal polynomials using some rational transformations over finite fields". In: *Journal of Algebra and its Applications* 19.11 (2020), p. 2050210.

[KS95] Erich Kaltofen and Victor Shoup. "Subquadratic-time factoring of polynomials over finite fields". In: *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*. 1995, pp. 398–406.

[Kyu02] Melsik K. Kyuregyan. "Recurrent methods for constructing irreducible polynomials over GF(2^s)". In: *Finite Fields and Their Applications* 8.1 (2002), pp. 52–68.

[Kyu03] Melsik K. Kyuregyan. "Recurrent methods for constructing irreducible polynomials over $\mathbb{F}_q$ of odd characteristics". In: *Finite Fields and Their Applications* 9.1 (2003), pp. 39–58.

[Kyu04] Melsik K. Kyuregyan. "Iterated constructions of irreducible polynomials over finite fields with linearly independent roots". In: *Finite fields and their applications* 10.3 (2004), pp. 323–341.

[Kyu06] Melsik K. Kyuregyan. "Recurrent methods for constructing irreducible polynomials over $\mathbb{F}_q$ of odd characteristics, II". In: *Finite Fields and Their Applications* 12.3 (2006), pp. 357–378.

[LC17] Fen Li and Xiwang Cao. "A class of minimal cyclic codes over finite fields". In: *Discrete Mathematics* 340.1 (2017), pp. 3197–3206.

[Lin91] Jacobus H. van Lint. "Repeated-root cyclic codes". In: *IEEE Transactions on Information Theory* 37.2 (1991), pp. 343–345.

[Liu+05] Jingmei Liu, Baodian Wei, Xiangguo Cheng, and Xinmei Wang. "An AES S-box to increase complexity and cryptographic analysis". In: *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*. Vol. 1. IEEE. 2005, pp. 724–728.

[Liu+16] Li Liu, Lanqiang Li, Xiaoshan Kai, and Shixin Zhu. "Repeated-root constacyclic codes of length $3lp^s$ and their dual codes". In: *Finite Fields and Their Applications* 42 (2016), pp. 269–295.

[Liu+17] Li Liu, Lanqiang Li, Liqi Wang, and Shixin Zhu. "Repeated-root constacyclic codes of length n^lp^s". In: *Discrete Mathematics* 340.9 (2017), pp. 2250–2261.

[LN94] Rudolf Lidl and Harald Niederreiter. *Introduction to finite fields and their applications*. Cambridge university press, 1994.

[Lóp+13] Sergio R. López-Permouth, Hakan Özadam, Ferruh Özbudak, and Steve Szabo. "Polycyclic codes over Galois rings with applications to repeated-root constacyclic codes". In: *Finite Fields and Their Applications* 19.1 (2013), pp. 16–38.

[LY18] Fengwei Li and Qin Yue. "The primitive idempotents and weight distributions of irreducible constacyclic codes". In: *Designs, Codes and Cryptography* 86 (2018), pp. 771–784.

[LYL14] Fengwei Li, Qin Yue, and Chengju Li. "The minimum Hamming distances of irreducible cyclic codes". In: *Finite Fields and Their Applications* 29 (2014), pp. 225–242.

[LYL15] Fengwei Li, Qin Yue, and Chengju Li. "Irreducible cyclic codes of length $4p^n$ and $8p^n$". In: *Finite Fields and Their Applications* 34 (2015), pp. 208–234.

[McN95] G. McNay. "Topics in finite fields". In: *PhD, University of Glasgow, Glasgow, UK* (1995).

[Mey90] Helmut Meyn. "On the construction of irreducible self-reciprocal polynomials over finite fields". In: *Applicable algebra in engineering, communication and computing* 1.1 (1990), pp. 43–53.

[Mey95] Helmut Meyn. "Explicit n-polynomials of 2-power degree over finite fields, I". In: *Designs, Codes and Cryptography* 6.2 (1995), pp. 107–116.

[Mey96] Helmut Meyn. "Factorization of the Cyclotomic Polynomial $x^{2^n}+1$ over Finite Fields". In: *Finite Fields and Their Applications* 2.4 (1996), pp. 439–442.

[MGO15] Fabio Enrique Brochero Martínez, Carmen Rosa Giraldo Vergara, and Lilian Batista de Oliveira. "Explicit factorization of $x^n-1 \in \mathbb{F}_q[X]$". In: *Designs, Codes and Cryptography* 77 (2015), pp. 277–286.

[MR18] Fabio Enrique Brochero Martínez and Lucas Reis. "Factoring polynomials of the form $f(X^n) \in \mathbb{F}_q[X]$". In: *Finite Fields and Their Applications* 49 (2018), pp. 166–179.

[MRS19] Fabio Enrique Brochero Martínez, Lucas Reis, and Lays Silva-Jesus. "Factorization of composed polynomials and applications". In: *Discrete Mathematics* 342.12 (2019), p. 111603.

[MYM10] Ronald C. Mullin, Joseph L. Yucas, and Gary Lee Mullen. "A generalized counting and factoring method for polynomials over finite fields". In: *Journal of Combinatorial Mathematics and Combinatorial Computing* 72 (2010), pp. 121–143.

[ÖÖ09] Hakan Özadam and Ferruh Özbudak. "The minimum Hamming distance of cyclic codes of length $2p^s$". In: *International Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*. Springer. 2009, pp. 92–100.

[Pel81] A.E. Pellet. "Sur les fonctions irréductibles suivant un module premier". In: *CR Acad. Sci. Paris* 93.1065-1066 (1881), pp. 31–34.

[Pel89] A.E. Pellet. "Sur les fonctions réduites suivant un module premier". In: *Bulletin de la Société Mathématique de France* 17 (1889), pp. 156–167.

[PRW20] Daniel Panario, Lucas Reis, and Qiang Wang. "Construction of irreducible polynomials through rational transformations". In: *Journal of Pure and Applied Algebra* 224.5 (2020), p. 106241.

[Rak+22] Supakarn Rakphon, Wutichai Chongchitmate, Jirayu Phuto, and Chakkrid Klin-eam. "Explicit factorization of $x^{l_1^{m_1} l_2^{m_2} l_3^{m_3}} - a$ and a-constacyclic codes over a finite field". In: *Communications in Algebra* 50.11 (2022), pp. 4725–4745.

[Ran19] Saroj Rani. "Structure of repeated-root constacyclic codes of length $8l^m p^n$". In: *Asian-European Journal of Mathematics* 12.04 (2019), p. 1950050.

[SAS22] P.L. Sharma, Ashima, and Arun Kumar Sharma. "Recursive construction of normal polynomials over finite fields". In: *Journal of Discrete Mathematical Sciences and Cryptography* 25.8 (2022), pp. 2645–2660.

[Sem89] Igor A. Semaev. "Construction of polynomials irreducible over a finite field with linearly independent roots". In: *Mathematics of the USSR-Sbornik* 63.2 (1989), p. 507.

[Ser66] Joseph Alfred Serret. *Cours d'algèbre supérieure*. Vol. 1. Gauthier-Villars, 1866.

[SF20] Zexia Shi and Fang-Wei Fu. "The primitive idempotents of irreducible constacyclic codes and LCD cyclic codes". In: *Cryptography and Communications* 12.1 (2020), pp. 29–52.

[Sha15] Anuradha Sharma. "Repeated-root constacyclic codes of length $l^t p^s$ and their dual codes". In: *Cryptography and Communications* 7 (2015), pp. 229–255.

[Sho95] Victor Shoup. "A new polynomial factorization algorithm and its implementation". In: *Journal of Symbolic Computation* 20.4 (1995), pp. 363–397.

[SR16] Anuradha Sharma and Saroj Rani. "Repeated-root constacyclic codes of length $4l^m p^n$". In: *Finite Fields and Their Applications* 40 (2016), pp. 163–200.

[ST01] National Institute of Standards and Technology. *Advanced Encryption Standard (AES)*. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf. 2001.

[Ste01] Greg Stein. "Using the theory of cyclotomy to factor cyclotomic polynomials over finite fields". In: *Mathematics of computation* 70.235 (2001), pp. 1237–1251.

[TBD08] Minh Triet Tran, Doan Khanh Bui, and Anh Duc Duong. "Gray S-box for advanced encryption standard". In: *2008 international conference on computational intelligence and security*. Vol. 1. IEEE. 2008, pp. 253–258.

[Ton16] Hongxi Tong. "Repeated-root constacyclic codes of length $kl^a p^b$ over a finite field". In: *Finite Fields and Their Applications* 41 (2016), pp. 159–173.

[TW13] Aleksandr Tuxanidy and Qiang Wang. "Composed products and factors of cyclotomic polynomials over finite fields". In: *Designs, codes and cryptography* 69.2 (2013), pp. 203–231.

[Ugo13] Simone Ugolini. "Sequences of binary irreducible polynomials". In: *Discrete Mathematics* 313.22 (2013), pp. 2656–2662.

[Ugo15] Simone Ugolini. "Sequences of irreducible polynomials without prescribed coefficients over odd prime fields". In: *Designs, Codes and Cryptography* 75 (2015), pp. 145–155.

[Ugo16] Simone Ugolini. "On an iterated construction of irreducible polynomials over finite fields of even characteristic by Kyuregyan". In: *Czechoslovak Mathematical Journal* 66 (2016), pp. 243–250.

[Van02] Mark Van Hoeij. "Factoring polynomials and the knapsack problem". In: *Journal of Number theory* 95.2 (2002), pp. 167–189.

[VG90] Joachim Von Zur Gathen and Mark Giesbrecht. "Constructing normal bases in finite fields". In: *J. Symb. Comput* 10 (1990), pp. 547–570.

[VS92] Joachim Von Zur Gathen and Victor Shoup. "Computing Frobenius maps and factoring polynomials". In: *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*. 1992, pp. 97–105.

[Wie88] Doug Wiedemann. "An iterated quadratic extension of GF(2)". In: *Fibonacci Quart* 26.4 (1988), pp. 290–295.

[Wu+17] Hongfeng Wu, Li Zhu, Rongquan Feng, and Siman Yang. "Explicit factorizations of cyclotomic polynomials over finite fields". In: *Designs, Codes and Cryptography* 83 (2017), pp. 197–217.

[WW12] Liping Wang and Qiang Wang. "On explicit factors of cyclotomic polynomials over finite fields". In: *Designs, Codes and Cryptography* 63 (2012), pp. 87–104.

[WY18] Yansheng Wu and Qin Yue. "Factorizations of binomial polynomials and enumerations of LCD and self-dual constacyclic codes". In: *IEEE Transactions on Information Theory* 65.3 (2018), pp. 1740–1751.

[WY21] Yansheng Wu and Qin Yue. "Further factorization of $x^n - 1$ over a finite field (II)". In: *Discrete Mathematics, Algorithms and Applications* 13.06 (2021), p. 2150070.

[WYF18] Yansheng Wu, Qin Yue, and Shuqin Fan. "Further factorization of $x^n - 1$ over a finite field". In: *Finite Fields and Their Applications* 54 (2018), pp. 197–215.

Appendix A

Numerical results

Example A.1 (Example 2.46 extended). We present the details of the computation in Example 2.46 which have been omitted in Chapter 2.

14. For $0 \leq j \leq 7 = {d_1}^{(s)} - 1$ we compute $b_j = (\zeta^j_{{d_1}^{(s)}} b)^r$:

$$
\begin{aligned}
b_0 &= 3\beta^{11} + 2\beta^{10} + 3\beta^9 + \beta^7 + 2\beta^6 + 3\beta^5 + 3\beta^4 + 3\beta^3 + \beta^2 + \beta + 4,\\
b_1 &= 4\beta^8 + \beta^7 + 4\beta^6 + 4\beta^4 + 2\beta^3 + 2\beta^2 + 1,\\
b_2 &= \beta^{11} + 4\beta^{10} + \beta^9 + 2\beta^7 + 4\beta^6 + \beta^5 + \beta^4 + \beta^3 + 2\beta^2 + 2\beta + 3,\\
b_3 &= 3\beta^8 + 2\beta^7 + 3\beta^6 + 3\beta^4 + 4\beta^3 + 4\beta^2 + 2,\\
b_4 &= 2\beta^{11} + 3\beta^{10} + 2\beta^9 + 4\beta^7 + 3\beta^6 + 2\beta^5 + 2\beta^4 + 2\beta^3 + 4\beta^2 + 4\beta + 1,\\
b_5 &= \beta^8 + 4\beta^7 + \beta^6 + \beta^4 + 3\beta^3 + 3\beta^2 + 4,\\
b_6 &= 4\beta^{11} + \beta^{10} + 4\beta^9 + 3\beta^7 + \beta^6 + 4\beta^5 + 4\beta^4 + 4\beta^3 + 3\beta^2 + 3\beta + 2,\\
b_7 &= 2\beta^8 + 3\beta^7 + 2\beta^6 + 2\beta^4 + \beta^3 + \beta^2 + 3.
\end{aligned}
$$

17. On the next six pages we present the complete factorization of the binomial

$$X^{2^5 \cdot 7^2 \cdot 13} - 4 \in \mathbb{F}_5[X].$$

For every $j \in \{0, 1\}$, every $v \in \{1, 7\}$, every $i \in \{0, 1, 2, 4, 7, 8, 13, 14, 16, 17, 28\}$ such that $\gcd(i, v) = 1$ and every $m \in \{0, \ldots, \gcd(t_i, 4) - 1\}$ we give the monic irreducible factor $S_{(j,v,i,m)}$ as specified in Theorem 2.41:

$$\sum_{l=0}^{c_i} X^{4 \cdot v \cdot l} \cdot (-1)^{c_i - l} \sum_{\substack{\mathcal{U} \subseteq \{0, \ldots, c_i - 1\} \\ |\mathcal{U}| = c_i - l}} \prod_{u \in \mathcal{U}} (\zeta_{91}^{i \cdot q^m} \cdot b_j^v)^{q^u}.$$

$$
\begin{aligned}
&(0,1,0,0) : X^{16} + 3\\
&(1,1,0,0) : X^{16} + 2\\
&(0,1,1,0) : X^{48} + 3X^{44} + 3X^{40} + 3X^{32} + 3X^{28} + 2X^{24} + X^{16} + 4X^{12} + X^8 + X^4 + 2\\
&(1,1,1,0) : X^{48} + 2X^{44} + 4X^{40} + 2X^{32} + 4X^{28} + X^{24} + 2X^{20} + 2X^{16} + 2X^{12} + 4X^8 + 3\\
&(0,1,1,1) : X^{48} + 4X^{44} + 2X^{40} + 3X^{32} + 4X^{28} + 3X^{24} + X^{16} + 2X^{12} + 4X^8 + 2X^4 + 2\\
&(1,1,1,1) : X^{48} + 4X^{44} + X^{40} + 2X^{32} + 3X^{28} + 4X^{24} + X^{20} + 2X^{16} + 4X^{12} + X^8 + 3\\
&(0,1,1,2) : X^{48} + 2X^{44} + 3X^{40} + 3X^{32} + 2X^{28} + 2X^{24} + X^{16} + X^{12} + X^8 + 4X^4 + 2\\
&(1,1,1,2) : X^{48} + 3X^{44} + 4X^{40} + 2X^{32} + X^{28} + X^{24} + 3X^{20} + 2X^{16} + 3X^{12} + 4X^8 + 3\\
&(0,1,1,3) : X^{48} + X^{44} + 2X^{40} + 3X^{32} + X^{28} + 3X^{24} + X^{16} + 3X^{12} + 4X^8 + 3X^4 + 2\\
&(1,1,1,3) : X^{48} + X^{44} + X^{40} + 2X^{32} + 2X^{28} + 4X^{24} + 4X^{20} + 2X^{16} + X^{12} + X^8 + 3\\
&(0,7,1,0) : X^{336} + 3X^{308} + 3X^{280} + 3X^{224} + 3X^{196} + 2X^{168} + X^{112} + 4X^{84} + X^{56} + X^{28} + 2\\
&(1,7,1,0) : X^{336} + 2X^{308} + 4X^{280} + 2X^{224} + 4X^{196} + X^{168} + 2X^{140} + 2X^{112} + 2X^{84} + 4X^{56} + 3\\
&(0,7,1,1) : X^{336} + 4X^{308} + 2X^{280} + 3X^{224} + 4X^{196} + 3X^{168} + X^{112} + 2X^{84} + 4X^{56} + 2X^{28} + 2\\
&(1,7,1,1) : X^{336} + 4X^{308} + X^{280} + 2X^{224} + 3X^{196} + 4X^{168} + X^{140} + 2X^{112} + 4X^{84} + X^{56} + 3\\
&(0,7,1,2) : X^{336} + 2X^{308} + 3X^{280} + 3X^{224} + 2X^{196} + 2X^{168} + X^{112} + X^{84} + X^{56} + 4X^{28} + 2\\
&(1,7,1,2) : X^{336} + 3X^{308} + 4X^{280} + 2X^{224} + X^{196} + X^{168} + 3X^{140} + 2X^{112} + 3X^{84} + 4X^{56} + 3\\
&(0,7,1,3) : X^{336} + X^{308} + 2X^{280} + 3X^{224} + X^{196} + 3X^{168} + X^{112} + 3X^{84} + 4X^{56} + 3X^{28} + 2\\
&(1,7,1,3) : X^{336} + X^{308} + X^{280} + 2X^{224} + 2X^{196} + 4X^{168} + 4X^{140} + 2X^{112} + X^{84} + X^{56} + 3\\
&(0,1,2,0) : X^{48} + 3X^{44} + 2X^{40} + X^{36} + 4X^{32} + 3X^{28} + 4X^{24} + X^{20} + 4X^{16} + 4X^{12} + 2X^4 + 2\\
&(1,1,2,0) : X^{48} + 2X^{36} + 2X^{32} + 2X^{24} + 2X^{16} + X^{12} + 4X^8 + 4X^4 + 3\\
&(0,1,2,1) : X^{48} + 4X^{44} + 3X^{40} + 2X^{36} + 4X^{32} + 4X^{28} + X^{24} + 2X^{20} + 4X^{16} + 2X^{12} + 4X^4 + 2\\
&(1,1,2,1) : X^{48} + X^{36} + 2X^{32} + 3X^{24} + 2X^{16} + 2X^{12} + X^8 + 2X^4 + 3\\
&(0,1,2,2) : X^{48} + 2X^{44} + 2X^{40} + 4X^{36} + 4X^{32} + 2X^{28} + 4X^{24} + 4X^{20} + 4X^{16} + X^{12} + 3X^4 + 2
\end{aligned}
$$

$(1,1,2,2) : X^{48} + 3X^{36} + 2X^{32} + 2X^{24} + 2X^{16} + 4X^{12} + 4X^8 + X^4 + 3$

$(0,1,2,3) : X^{48} + X^{44} + 3X^{40} + 3X^{36} + 4X^{32} + X^{28} + X^{24} + 3X^{20} + 4X^{16} + 3X^{12} + X^4 + 2$

$(1,1,2,3) : X^{48} + 4X^{36} + 2X^{32} + 3X^{24} + 2X^{16} + 3X^{12} + X^8 + 3X^4 + 3$

$(0,7,2,0) : X^{336} + 3X^{308} + 2X^{280} + X^{252} + 4X^{224} + 3X^{196} + 4X^{168} + X^{140} + 4X^{112} + 4X^{84} + 2X^{28} + 2$

$(1,7,2,0) : X^{336} + 2X^{252} + 2X^{224} + 2X^{168} + 2X^{112} + X^{84} + 4X^{56} + 4X^{28} + 3$

$(0,7,2,1) : X^{336} + 4X^{308} + 3X^{280} + 2X^{252} + 4X^{224} + 4X^{196} + X^{168} + 2X^{140} + 4X^{112} + 2X^{84} + 4X^{28} + 2$

$(1,7,2,1) : X^{336} + X^{252} + 2X^{224} + 3X^{168} + 2X^{112} + 2X^{84} + X^{56} + 2X^{28} + 3$

$(0,7,2,2) : X^{336} + 2X^{308} + 2X^{280} + 4X^{252} + 4X^{224} + 2X^{196} + 4X^{168} + 4X^{140} + 4X^{112} + X^{84} + 3X^{28} + 2$

$(1,7,2,2) : X^{336} + 3X^{252} + 2X^{224} + 2X^{168} + 2X^{112} + 4X^{84} + 4X^{56} + X^{28} + 3$

$(0,7,2,3) : X^{336} + X^{308} + 3X^{280} + 3X^{252} + 4X^{224} + X^{196} + X^{168} + 3X^{140} + 4X^{112} + 3X^{84} + X^{28} + 2$

$(1,7,2,3) : X^{336} + 4X^{252} + 2X^{224} + 3X^{168} + 2X^{112} + 3X^{84} + X^{56} + 3X^{28} + 3$

$(0,1,4,0) : X^{48} + 2X^{44} + 2X^{40} + 3X^{32} + 2X^{28} + X^{24} + X^{20} + 4X^{12} + 2X^8 + X^4 + 2$

$(1,1,4,0) : X^{48} + 4X^{44} + 3X^{40} + 4X^{36} + 4X^{32} + 4X^{28} + X^{24} + 2X^{16} + 3X^{12} + 3X^4 + 3$

$(0,1,4,1) : X^{48} + X^{44} + 3X^{40} + 3X^{32} + X^{28} + 4X^{24} + 2X^{20} + 2X^{12} + 3X^8 + 2X^4 + 2$

$(1,1,4,1) : X^{48} + 3X^{44} + 2X^{40} + 2X^{36} + 4X^{32} + 3X^{28} + 4X^{24} + 2X^{16} + X^{12} + 4X^4 + 3$

$(0,1,4,2) : X^{48} + 3X^{44} + 2X^{40} + 3X^{32} + 3X^{28} + X^{24} + 4X^{20} + X^{12} + 2X^8 + 4X^4 + 2$

$(1,1,4,2) : X^{48} + X^{44} + 3X^{40} + X^{36} + 4X^{32} + X^{28} + X^{24} + 2X^{16} + 2X^{12} + 2X^4 + 3$

$(0,1,4,3) : X^{48} + 4X^{44} + 3X^{40} + 3X^{32} + 4X^{28} + 4X^{24} + 3X^{20} + 3X^{12} + 3X^8 + 3X^4 + 2$

$(1,1,4,3) : X^{48} + 2X^{44} + 2X^{40} + 3X^{36} + 4X^{32} + 2X^{28} + 4X^{24} + 2X^{16} + 4X^{12} + X^4 + 3$

$(0,7,4,0) : X^{336} + 2X^{308} + 2X^{280} + 3X^{224} + 2X^{196} + X^{168} + X^{140} + 4X^{84} + 2X^{56} + X^{28} + 2$

$(1,7,4,0) : X^{336} + 4X^{308} + 3X^{280} + 4X^{252} + 4X^{224} + 4X^{196} + X^{168} + 2X^{112} + 3X^{84} + 3X^{28} + 3$

$(0,7,4,1) : X^{336} + X^{308} + 3X^{280} + 3X^{224} + X^{196} + 4X^{168} + 2X^{140} + 2X^{84} + 3X^{56} + 2X^{28} + 2$

$(1,7,4,1) : X^{336} + 3X^{308} + 2X^{280} + 2X^{252} + 4X^{224} + 3X^{196} + 4X^{168} + 2X^{112} + X^{84} + 4X^{28} + 3$

$(0,7,4,2) : X^{336} + 3X^{308} + 2X^{280} + 3X^{224} + 3X^{196} + X^{168} + 4X^{140} + X^{84} + 2X^{56} + 4X^{28} + 2$

$$(1,7,4,2) : X^{336} + X^{308} + 3X^{280} + X^{252} + 4X^{224} + X^{196} + X^{168} + 2X^{112} + 2X^{84} + 2X^{28} + 3$$
$$(0,7,4,3) : X^{336} + 4X^{308} + 3X^{280} + 3X^{224} + 4X^{196} + 4X^{168} + 3X^{140} + 3X^{84} + 3X^{56} + 3X^{28} + 2$$
$$(1,7,4,3) : X^{336} + 2X^{308} + 2X^{280} + 3X^{252} + 4X^{224} + 2X^{196} + 4X^{168} + 2X^{112} + 4X^{84} + X^{28} + 3$$
$$(0,1,7,0) : X^{16} + 3X^8 + 4X^4 + 3$$
$$(1,1,7,0) : X^{16} + 4X^{12} + 4X^8 + 2$$
$$(0,1,7,1) : X^{16} + 2X^8 + 3X^4 + 3$$
$$(1,1,7,1) : X^{16} + 3X^{12} + X^8 + 2$$
$$(0,1,7,2) : X^{16} + 3X^8 + X^4 + 3$$
$$(1,1,7,2) : X^{16} + X^{12} + 4X^8 + 2$$
$$(0,1,7,3) : X^{16} + 2X^8 + 2X^4 + 3$$
$$(1,1,7,3) : X^{16} + 2X^{12} + X^8 + 2$$
$$(0,1,8,0) : X^{48} + 3X^{40} + 4X^{36} + 4X^{32} + 4X^{28} + 2X^{24} + 3X^{20} + 4X^{16} + 3X^8 + 4X^4 + 2$$
$$(1,1,8,0) : X^{48} + 2X^{44} + 3X^{40} + 3X^{36} + 3X^{32} + X^{24} + X^{20} + 4X^{16} + 4X^8 + X^4 + 3$$
$$(0,1,8,1) : X^{48} + 2X^{40} + 3X^{36} + 4X^{32} + 2X^{28} + 3X^{24} + X^{20} + 4X^{16} + 2X^8 + 3X^4 + 2$$
$$(1,1,8,1) : X^{48} + 4X^{44} + 2X^{40} + 4X^{36} + 3X^{32} + 4X^{24} + 3X^{20} + 4X^{16} + X^8 + 3X^4 + 3$$
$$(0,1,8,2) : X^{48} + 3X^{40} + X^{36} + 4X^{32} + X^{28} + 2X^{24} + 2X^{20} + 4X^{16} + 3X^8 + X^4 + 2$$
$$(1,1,8,2) : X^{48} + 3X^{44} + 3X^{40} + 2X^{36} + 3X^{32} + X^{24} + 4X^{20} + 4X^{16} + 4X^8 + 4X^4 + 3$$
$$(0,1,8,3) : X^{48} + 2X^{40} + 2X^{36} + 4X^{32} + 3X^{28} + 3X^{24} + 4X^{20} + 4X^{16} + 2X^8 + 2X^4 + 2$$
$$(1,1,8,3) : X^{48} + X^{44} + 2X^{40} + X^{36} + 3X^{32} + 4X^{24} + 2X^{20} + 4X^{16} + X^8 + 2X^4 + 3$$
$$(0,7,8,0) : X^{336} + 3X^{280} + 4X^{252} + 4X^{224} + 4X^{196} + 2X^{168} + 3X^{140} + 4X^{112} + 3X^{56} + 4X^{28} + 2$$
$$(1,7,8,0) : X^{336} + 2X^{308} + 3X^{280} + 3X^{252} + 3X^{224} + X^{168} + X^{140} + 4X^{112} + 4X^{56} + X^{28} + 3$$
$$(0,7,8,1) : X^{336} + 2X^{280} + 3X^{252} + 4X^{224} + 2X^{196} + 3X^{168} + X^{140} + 4X^{112} + 2X^{56} + 3X^{28} + 2$$
$$(1,7,8,1) : X^{336} + 4X^{308} + 2X^{280} + 4X^{252} + 3X^{224} + 4X^{168} + 3X^{140} + 4X^{112} + X^{56} + 3X^{28} + 3$$
$$(0,7,8,2) : X^{336} + 3X^{280} + X^{252} + 4X^{224} + X^{196} + 2X^{168} + 2X^{140} + 4X^{112} + 3X^{56} + X^{28} + 2$$

$(1,7,8,2) : X^{336} + 3X^{308} + 3X^{280} + 2X^{252} + 3X^{224} + X^{168} + 4X^{140} + 4X^{112} + 4X^{56} + 4X^{28} + 3$

$(0,7,8,3) : X^{336} + 2X^{280} + 2X^{252} + 4X^{224} + 3X^{196} + 3X^{168} + 4X^{140} + 4X^{112} + 2X^{56} + 2X^{28} + 2$

$(1,7,8,3) : X^{336} + X^{308} + 2X^{280} + X^{252} + 3X^{224} + 4X^{168} + 2X^{140} + 4X^{112} + X^{56} + 2X^{28} + 3$

$(0,1,13,0) : X^{48} + 4X^{40} + 4X^{32} + 2X^{24} + 2X^{16} + X^8 + 2$

$(1,1,13,0) : X^{48} + 3X^{40} + X^{32} + X^{24} + 2X^{16} + 2X^8 + 3$

$(0,1,13,1) : X^{48} + X^{40} + 4X^{32} + 3X^{24} + 2X^{16} + 4X^8 + 2$

$(1,1,13,1) : X^{48} + 2X^{40} + X^{32} + 4X^{24} + 2X^{16} + 3X^8 + 3$

$(0,7,13,0) : X^{336} + 4X^{280} + 4X^{224} + 2X^{168} + 2X^{112} + X^{56} + 2$

$(1,7,13,0) : X^{336} + 3X^{280} + X^{224} + X^{168} + 2X^{112} + 2X^{56} + 3$

$(0,7,13,1) : X^{336} + X^{280} + 4X^{224} + 3X^{168} + 2X^{112} + 4X^{56} + 2$

$(1,7,13,1) : X^{336} + 2X^{280} + X^{224} + 4X^{168} + 2X^{112} + 3X^{56} + 3$

$(0,1,14,0) : X^{16} + 2X^{12} + 4X^8 + 4X^4 + 3$

$(1,1,14,0) : X^{16} + 4X^{12} + 2X^8 + 3X^4 + 2$

$(0,1,14,1) : X^{16} + X^{12} + X^8 + 3X^4 + 3$

$(1,1,14,1) : X^{16} + 3X^{12} + 3X^8 + 4X^4 + 2$

$(0,1,14,2) : X^{16} + 3X^{12} + 4X^8 + X^4 + 3$

$(1,1,14,2) : X^{16} + X^{12} + 2X^8 + 2X^4 + 2$

$(0,1,14,3) : X^{16} + 4X^{12} + X^8 + 2X^4 + 3$

$(1,1,14,3) : X^{16} + 2X^{12} + 3X^8 + X^4 + 2$

$(0,1,16,0) : X^{48} + 3X^{44} + 3X^{40} + 2X^{36} + 4X^{32} + 4X^{24} + 4X^{16} + 4X^{12} + 2$

$(1,1,16,0) : X^{48} + 4X^{44} + 3X^{36} + 2X^{32} + 2X^{28} + 2X^{24} + X^{20} + 2X^{16} + 2X^{12} + X^8 + X^4 + 3$

$(0,1,16,1) : X^{48} + 4X^{44} + 2X^{40} + 4X^{36} + 4X^{32} + X^{24} + 4X^{16} + 2X^{12} + 2$

$(1,1,16,1) : X^{48} + 3X^{44} + 4X^{36} + 2X^{32} + 4X^{28} + 3X^{24} + 3X^{20} + 2X^{16} + 4X^{12} + 4X^8 + 3X^4 + 3$

$(0,1,16,2) : X^{48} + 2X^{44} + 3X^{40} + 3X^{36} + 4X^{32} + 4X^{24} + 4X^{16} + X^{12} + 2$

$$(1,1,16,2) : X^{48} + X^{44} + 2X^{36} + 2X^{32} + 3X^{28} + 2X^{24} + 4X^{20} + 2X^{16} + 3X^{12} + X^{8} + 4X^{4} + 3$$

$$(0,1,16,3) : X^{48} + X^{44} + 2X^{40} + X^{36} + 4X^{32} + X^{24} + 4X^{16} + 3X^{12} + 2$$

$$(1,1,16,3) : X^{48} + 2X^{44} + X^{36} + 2X^{32} + X^{28} + 3X^{24} + 2X^{20} + 2X^{16} + X^{12} + 4X^{8} + 2X^{4} + 3$$

$$(0,7,16,0) : X^{336} + 3X^{308} + 3X^{280} + 2X^{252} + 4X^{224} + 4X^{168} + 4X^{112} + 4X^{84} + 2$$

$$(1,7,16,0) : X^{336} + 4X^{308} + 3X^{252} + 2X^{224} + 2X^{196} + 2X^{168} + X^{140} + 2X^{112} + 2X^{84} + X^{56} + X^{28} + 3$$

$$(0,7,16,1) : X^{336} + 4X^{308} + 2X^{280} + 4X^{252} + 4X^{224} + X^{168} + 4X^{112} + 2X^{84} + 2$$

$$(1,7,16,1) : X^{336} + 3X^{308} + 4X^{252} + 2X^{224} + 4X^{196} + 3X^{168} + 3X^{140} + 2X^{112} + 4X^{84} + 4X^{56} + 3X^{28} + 3$$

$$(0,7,16,2) : X^{336} + 2X^{308} + 3X^{280} + 3X^{252} + 4X^{224} + 4X^{168} + 4X^{112} + X^{84} + 2$$

$$(1,7,16,2) : X^{336} + X^{308} + 2X^{252} + 2X^{224} + 3X^{196} + 2X^{168} + 4X^{140} + 2X^{112} + 3X^{84} + X^{56} + 4X^{28} + 3$$

$$(0,7,16,3) : X^{336} + X^{308} + 2X^{280} + X^{252} + 4X^{224} + X^{168} + 4X^{112} + 3X^{84} + 2$$

$$(1,7,16,3) : X^{336} + 2X^{308} + X^{252} + 2X^{224} + X^{196} + 3X^{168} + 2X^{140} + 2X^{112} + X^{84} + 4X^{56} + 2X^{28} + 3$$

$$(0,1,17,0) : X^{48} + 3X^{44} + 2X^{36} + 4X^{32} + 3X^{24} + X^{20} + 3X^{16} + 4X^{12} + 4X^{8} + X^{4} + 2$$

$$(1,1,17,0) : X^{48} + 4X^{44} + 4X^{40} + 4X^{36} + 4X^{28} + 2X^{24} + 2X^{20} + 4X^{16} + 4X^{8} + 2X^{4} + 3$$

$$(0,1,17,1) : X^{48} + 4X^{44} + 4X^{36} + 4X^{32} + 2X^{24} + 2X^{20} + 3X^{16} + 2X^{12} + X^{8} + 2X^{4} + 2$$

$$(1,1,17,1) : X^{48} + 3X^{44} + X^{40} + 2X^{36} + 3X^{28} + 3X^{24} + X^{20} + 4X^{16} + X^{8} + X^{4} + 3$$

$$(0,1,17,2) : X^{48} + 2X^{44} + 3X^{36} + 4X^{32} + 3X^{24} + 4X^{20} + 3X^{16} + X^{12} + 4X^{8} + 4X^{4} + 2$$

$$(1,1,17,2) : X^{48} + X^{44} + 4X^{40} + X^{36} + X^{28} + 2X^{24} + 3X^{20} + 4X^{16} + 4X^{8} + 3X^{4} + 3$$

$$(0,1,17,3) : X^{48} + X^{44} + X^{36} + 4X^{32} + 2X^{24} + 3X^{20} + 3X^{16} + 3X^{12} + X^{8} + 3X^{4} + 2$$

$$(1,1,17,3) : X^{48} + 2X^{44} + X^{40} + 3X^{36} + 2X^{28} + 3X^{24} + 4X^{20} + 4X^{16} + X^{8} + 4X^{4} + 3$$

$$(0,7,17,0) : X^{336} + 3X^{308} + 2X^{252} + 4X^{224} + 3X^{168} + X^{140} + 3X^{112} + 4X^{84} + 4X^{56} + X^{28} + 2$$

$$(1,7,17,0) : X^{336} + 4X^{308} + 4X^{280} + 4X^{252} + 4X^{196} + 2X^{168} + 2X^{140} + 4X^{112} + 4X^{56} + 2X^{28} + 3$$

$$(0,7,17,1) : X^{336} + 4X^{308} + 4X^{252} + 4X^{224} + 2X^{168} + 2X^{140} + 3X^{112} + 2X^{84} + X^{56} + 2X^{28} + 2$$

$$(1,7,17,1) : X^{336} + 3X^{308} + X^{280} + 2X^{252} + 3X^{196} + 3X^{168} + X^{140} + 4X^{112} + X^{56} + X^{28} + 3$$

$$(0,7,17,2) : X^{336} + 2X^{308} + 3X^{252} + 4X^{224} + 3X^{168} + 4X^{140} + 3X^{112} + X^{84} + 4X^{56} + 4X^{28} + 2$$

$(1,7,17,2) : X^{336} + X^{308} + 4X^{280} + X^{252} + X^{196} + 2X^{168} + 3X^{140} + 4X^{112} + 4X^{56} + 3X^{28} + 3$

$(0,7,17,3) : X^{336} + X^{308} + X^{252} + 4X^{224} + 2X^{168} + 3X^{140} + 3X^{112} + 3X^{84} + X^{56} + 3X^{28} + 2$

$(1,7,17,3) : X^{336} + 2X^{308} + X^{280} + 3X^{252} + 2X^{196} + 3X^{168} + 4X^{140} + 4X^{112} + X^{56} + 4X^{28} + 3$

$(0,1,28,0) : X^{16} + 3X^{12} + 3X^8 + 2X^4 + 3$

$(1,1,28,0) : X^{16} + 2X^{12} + 4X^8 + 2X^4 + 2$

$(0,1,28,1) : X^{16} + 4X^{12} + 2X^8 + 4X^4 + 3$

$(1,1,28,1) : X^{16} + 4X^{12} + X^8 + X^4 + 2$

$(0,1,28,2) : X^{16} + 2X^{12} + 3X^8 + 3X^4 + 3$

$(1,1,28,2) : X^{16} + 3X^{12} + 4X^8 + 3X^4 + 2$

$(0,1,28,3) : X^{16} + X^{12} + 2X^8 + X^4 + 3$

$(1,1,28,3) : X^{16} + X^{12} + X^8 + 4X^4 + 2$

Appendix B

Source code

B.1 The classes `RichFiniteField` and `RichPolynomial`

The SageMath/Python programs presented in this chapter are all based on the two modules `RichFiniteFieldClass.sage` and `RichPolynomialClass.sage`. In Section 1.3 we explain how to preparse them. When preparsed, the Python modules `RichFiniteFieldClass.py` and `RichPolynomialClass.py` can be loaded with the following commands:

```
from RichFiniteFieldClass import *
from RichPolynomialClass import *
```

The script B.3 `RichTutorial.sage` shows how to use the two modules in more detail.

The following file B.1 contains the specification of the custom Python class `RichFiniteField`. It contains the base class `RichFiniteField` and its kid `RichExtensionField`.

The base class `RichFiniteField` consists of a SageMath finite field `GF(q)` of size q, a polynomial ring over this finite field and functions returning the prime factors of $q-1$, a primitive element in $\mathbb{F}_q$ and primitive n-th roots of unity in $\mathbb{F}_q$ for positive integers n such that $n \mid q-1$.

The class `RichExtensionField` stores a SageMath finite field of size q^n for a positive integer n. It inherits from `RichFiniteField` and additionally stores its base field $\mathbb{F}_q$ as a `RichFiniteField` instance, a `RelativeFiniteFieldExtension` and the homomorphism between the base field and the extension field. The functions defined in this class can cast elements from one of the two fields to the other and can cast `RichPolynomials` over one of the two fields to the other. Furthermore, for elements of the extension field, the function `minimal_polynomial_over_basefield(self,el)` returns the minimal polynomial of `el` over the base field $\mathbb{F}_q$. In SageMath only minimal polynomials over the prime field $\mathbb{F}_{\mathrm{char}(\mathbb{F}_q)}$ are defined. Likewise `characteristic_polynomial_over_basefield(self,el)` computes the characteristic polynomial of `el` over the base field.

Source Code B.1: RichFiniteFieldClass.sage

```
1  # ****************************************************************************
2  #       Copyright (C) 2023 Anna-Maurin Graner
3  #
4  # This program is free software: you can redistribute it and/or modify
5  # it under the terms of the GNU General Public License as published by
6  # the Free Software Foundation, either version 3 of the License, or
7  # (at your option) any later version.
8  #                  https://www.gnu.org/licenses/
9  # ****************************************************************************
10
```

```
# This file contains the following Python Classes for computations with polynomials over finite fields and their extension fields
# - RichFiniteField
# - RichExtensionField (kid of RichFiniteField)

# The class stores a finite field together with its polynomial ring so that these two structures can be used together easily

from sage.coding.relative_finite_field_extension import *
import RichPolynomialClass

sep = "\t"

# CLASSES

# The Class RichFiniteField stores a Sagemath finite field instance _FF and a polynomial ring over this finite field
class RichFiniteField:
    def __init__(self, qq, charroot, charvar):
        self.q = qq
        if q in Primes():
            self.F = GF(self.q)
        else:
            self.F = GF(self.q, charroot)#, modulus="primitive")
        self.gen = (self.F).gen()
        self.zero = (self.F)(0)
        self.one = (self.F)(1)
        self._PRF = PolynomialRing(self.F,charvar)
        self.charvar = charvar
        self.x = (self._PRF).gen()
        return

    def __str__(self):
        return "RichFiniteField of size q = "+str(self.q)+" = "+str(factor(self.q))+"\n"+sep+"q-1 = "+str(self.q-1)+" = "+str(factor(self.q-1))+"\n"+sep+"modulus: "+sep+str((self.F).modulus())+"\n"+sep+"generator: "+sep+str(self.gen)+"\n"+sep+"polynomial ring in: "+sep+str(self.charvar)

    # public (redirection): returns the modulus of the finite field
    def get_modulus(self):
        return (self.F).modulus()

    # public: returns a list of the prime factors of q-1
    def get_primefactors(self):
        if not hasattr(self, "_primefactors"):
            self._add_primefactors()
        return self._primefactors

    # private: adds a list of the prime factors of q-1
    def _add_primefactors(self):
        self._primefactors = [fact[0] for fact in factor(self.q-1)]
        #self._primefactors.reverse()
        return

    # public: returns a primitive element of F_q
    def get_primitive_element(self):
        if not hasattr(self, "_primitive_element"):
            self._add_primitive_element()
        return self._primitive_element

    # private: adds a primitive element in F_q
    def _add_primitive_element(self):
        self._primitive_element = (self.F).multiplicative_generator()
```

```
68
69      # public: returns a primitive n-th root of unity if n divides q-1
70      def get_unityroot(self, n):
71          if (self.q-1)%n == 0:
72              return (self.get_primitive_element())^(int((self.q-1)/n))
73          else:
74              raise ValueError(str(n)+" does not divide "+str(self.q)+"-1.
    Thus, there is no primitive "+str(n)+"-th root of unity in F_{"+str(
    self.q)+"}.")
75          return
76
77      # public: returns a dictionary of one n-th primitive root of unity for
    all divisors n of q-1
78      def get_unityroots(self):
79          self._add_unityroots()
80          return self._unityroots
81
82      # private: add the dictionary of n-th primitive roots of unity for all
    divisors n of q-1
83      def _add_unityroots(self):
84          if not hasattr(self, "_unityroots"):
85              self._unityroots = dict()
86
87              for d in (self.q-1).divisors():
88                  self._unityroots[d] = self.get_unityroot(d)
89          return
90
91
92  # The class RichExtensionField which inherits from the class
    RichFiniteField, is mainly an extension of another RichFiniteField-
    instance such that the elements of the base field are embedded in the
    extension field.
93  class RichExtensionField(RichFiniteField):
94      def __init__(self, RFF, nn, charroot, charvar):
95          super().__init__(RFF.q^nn, charroot, charvar)
96          self._basefield = RFF
97          self._RFFE = RelativeFiniteFieldExtension(self.F,RFF.F)
98          self._base_embedding = (self._RFFE).embedding()
99          self._extension_degree = nn
100         return
101
102     def __str__(self):
103         return "RichExtension"+(super().__str__())[10:]+"\n"+sep+"extension
     of degree : "+sep+str(self._extension_degree)+"\n"+sep+"over : "+sep
    +((str(self._basefield)).split("\n"))[0]
104
105     # public: function returns basefield as RichFiniteField-instance
106     def get_basefield(self):
107         return self._basefield
108
109     # public: function returns the extension degree over basefield
110     def extension_degree(self):
111         return self._extension_degree
112
113     # public: function embeds an element el from the base field in
    extension field self
114     def to_extf(self, el):
115         return self._base_embedding(el)
116
117     # public: function casts an element of F_{q^n} to F_{q} (redirection)
118     def to_basef(self, el):
119         return (self._RFFE).cast_into_relative_field(el)
120
```

```
121     # public: function represents an element of F_{q^n} in the basis of F_{
        q^n} over F_q (redirection)
122     def in_basis_over_basef(self,el):
123         return (self._RFFE).relative_field_representation(el)
124 
125     # public: function converts a RichPolynomial-instance over the
        basefield to a RichPolynomial-instance over the extension field
126     def pol_to_extf(self,pol):
127         return RichPolynomialClass.RichPolynomial([self.to_extf(el) for el
        in pol.list], self)
128 
129     # public: function converts a RichPolynomial-instance over the
        extension field to a RichPolynomial instance over the basefield
130     def pol_to_basef(self, pol):
131         return RichPolynomialClass.RichPolynomial([self.to_basef(el) for el
         in pol.list], self._basefield)
132 
133     # public: function computes the minimal polynomial over the base field
        of an element of extension field
134     def minimal_polynomial_over_basefield(self, el):
135         mp  = self.x - el
136         for i in range(1, self._extension_degree):
137             eltoqi = el^(((self._basefield).q)^i)
138             if eltoqi == el:
139                 break
140             else:
141                 mp = mp * (self.x-eltoqi)
142         mp = [self.to_basef(e) for e in mp.list()]
143         mp.reverse()
144         return RichPolynomialClass.RichPolynomial(mp,self._basefield)
145 
146     # public: function computes the characteristic polynomial over the base
         field of an element of the extension field
147     def characteristic_polynomial_over_basefield(self, el):
148         mp = self.minimal_polynomial_over_basefield(el)
149         return RichPolynomialClass.RichPolynomial((mp.get_polynomial())^(
        int(self._extension_degree/mp.deg)), self._basefield)
```

The following file B.2 specifies the custom Python class `RichPolynomial`. The class stores a polynomial f over a `RichFiniteField` as a polynomial and as a list in parallel. It can compute and store the properties degree, weight, irreducibility, order, primitivity, k-normality, and its factorization, the roots of f in the finite field and the roots of f in the splitting field. Additionally, the functions `is_composition(self,m)` and `extract(self,m)` can be used to extract the polynomial g from f if f is a polynomial of the form $g(X^m)$ for a positive integer m. The function `construction_mbetap(self,p)` is an implementation of Construction 1 for a prime p.

Source Code B.2: RichPolynomialClass.sage

```
1  # ********************************************************************
2  #       Copyright (C) 2023 Anna-Maurin Graner
3  #
4  # This program is free software: you can redistribute it and/or modify
5  # it under the terms of the GNU General Public License as published by
6  # the Free Software Foundation, either version 3 of the License, or
7  # (at your option) any later version.
8  #                  https://www.gnu.org/licenses/
9  # ********************************************************************
10 
11 # This file contains the following Python Classes for computations with
     polynomials over finite fields and their extension fields
12 # - RichPolynomial
13 
```

```
14 # The class RichPolynomial is useful for working with a polynomial over a
       given RichFiniteField-instance as a list and as a polynomial in
       parallel.
15 # Furthermore, it is enRICHed with many useful functions returning
       properties of the polynomial such as its weight, its order, its k-
       normality, its primitivity.
16
17 # This class contains many functions some of which are redirections to
       existing SageMath-functions, many others are new implementations.
18 # All functions are split into private functions doing the computations and
        storing the result in a class attribute and a public function
       returning this attribute. This has the big advantage that all
       computations are done exactly once and only if needed.
19
20 from sage.arith.functions import LCM_list
21 import RichFiniteFieldClass
22
23 sep = "\t"
24
25 class RichPolynomial:
26     def __init__(self,pol,RFF):
27         self._RFF = RFF
28
29         if isinstance(pol, list):
30             self.list = [RFF.F(pol[i]) for i in range(len(pol))]
31             self.deg = len(self.list)-1
32         else:
33             self._polynomial = pol
34             self.deg = pol.degree()
35             self.list = pol.list()
36             (self.list).reverse()
37         return
38
39     def __str__(self):
40         return str(self.list)+" = "+str(self.get_polynomial())
41
42     def __mul__(self,pol):
43         return RichPolynomial(self.get_polynomial()*pol.get_polynomial(),
    self._RFF)
44
45     def __add__(self,pol):
46         return RichPolynomial(self.get_polynomial()+pol.get_polynomial(),
    self._RFF)
47
48     def __eq__(self,pol):
49         return self.list == pol.list
50
51     def __neq__(self,pol):
52         return not self.list == pol.list
53
54     # If self is a binomial with nonzero constant term, this functions
    returns a string representing this polynomial of the form X^t-a
55     def str_binomial(self):
56         if self.get_weight() >2 or (self.list)[-1] == (self._RFF).zero:
57             raise ValueError("The polynomial "+str(self)+" is no binomial!"
    )
58         return str((self._RFF).charvar)+"^"+str(self.deg)+" - ("+str(-(self
    .list)[-1])+")"
59
60     # This function returns a string detailing all information that can be
    obtained from a RichPolynomial-instance
61     def get_full_info(self):
62         s = str(self.list)+" = "+str(self.get_polynomial())+"\n"+sep+"
```

```
    irreducible: "+sep+str(self.is_irreducible())+"\n"+sep+"weight: "+sep+
    str(self.get_weight())
63          if self.is_irreducible():
64              s= s+"\n"+sep+"order: "+sep+str(self.get_order())+" = "+str((
    self.get_order()).factor())+"\n"+sep+"primitive: "+sep+str(self.get_
    primitive())+"\n"+sep+"k-normality: "+sep+str(self.get_knormal())
65          return s
66
67      # This function returns the finite field over which self is defined
68      def get_RFF(self):
69          return self._RFF
70
71      #public: returns the polynomial in F[X]
72      def get_polynomial(self):
73          if hasattr(self,"_polynomial"):
74              return self._polynomial
75          else:
76              self._add_polynomial()
77              return self._polynomial
78
79      #private: adds polynomial in F[X] to attributes
80      def _add_polynomial(self):
81          self._polynomial = sum([self.list[i]*(self._RFF).x^(self.deg-i) for
     i in range(self.deg+1)])
82          return
83
84      # public: get splitting field of polynomial as RichExtensionField-
    instance:
85      def get_splitting_field(self):
86          if not hasattr(self, "_RSplF"):
87              self._add_splitting_field()
88          return self._RSplF
89
90      # private: if polynomial is irreducible, this function adds the
    splitting field as a RichExtensionField-instance
91      def _add_splitting_field(self):
92          if not hasattr(self, "_RSplF"):
93              if self.is_irreducible():
94                  self._RSplF = RichFiniteFieldClass.RichExtensionField(self.
    _RFF, self.deg, "z", "X")
95              else:
96                  max_deg = LCM_list([fac[0].deg for fac in self.get_factors_
    asRP()])
97                  self._RSplF = RichFiniteFieldClass.RichExtensionField(self.
    _RFF,max_deg, "z", "X")
98          return
99
100     #public: returns self as RichPolynomial-instance over the splitting
    field
101     def get_RP_in_splitting_field(self):
102         if not hasattr(self, "_in_RSplF"):
103             self._add_pol_in_RSplF()
104         return self._in_RSplF
105
106     #private: adds self as RichPolynomial-instance over the splitting field
107     def _add_pol_in_RSplF(self):
108         self._in_RSplF = RichPolynomial([(self.get_splitting_field()).to_
    extf(e) for e in self.list], self.get_splitting_field())
109         return
110
111     #public: returns whether polynomial is irreducible
112     def is_irreducible(self):
113         if not hasattr(self,"_irreducible"):
```

```
114             self._add_irreducible()
115         return self._irreducible
116 
117     #private: determines whether polynomial is irreducible
118     def _add_irreducible(self):
119         self._irreducible = (self.get_polynomial()).is_irreducible()
120         return
121 
122     # public: returns the factorization of the polynomial over the finite
    field (as SageMath Factorization object)
123     def get_factorization(self):
124         if not hasattr(self, "_factorization"):
125             self._add_factorization()
126         return self._factorization
127 
128     # private: adds the factorization to the attributes of the polynomial
129     def _add_factorization(self):
130         self._factorization = (self.get_polynomial()).factor()
131         return
132 
133     #public: returns the list of factors as RichPolynomial-instances
134     def get_factors_asRP(self):
135         if not hasattr(self, "_factors"):
136             self._add_factors()
137         return self._factors
138 
139     # private: adds a list of factors as RichPolynomial-instances
140     def _add_factors(self):
141         self._factors = [(RichPolynomial( g[0], self._RFF), g[1]) for g in
    self.get_factorization()]
142         return
143 
144     # Returns one root of the polynomial in _RFF
145     def root(self):
146         if len(self.roots())>0:
147             return ((self.roots())[0])[0]
148         else:
149             raise ValueError("The polynomial "+str(self)+" does not have a
    root in the finite field of size "+str((self._RFF).q)+".")
150         return
151 
152     # Returns all roots of the polynomial in _RFF
153     def roots(self):
154         if not hasattr(self, "_roots"):
155             self._add_roots()
156         return self._roots
157 
158     # private:
159     def _add_roots(self):
160         self._roots = (self.get_polynomial()).roots()
161         return
162 
163     # public: Returns a list of the roots of all irreducible factors of the
     polynomial over the respective extension fields
164     def get_all_roots(self):
165         if not hasattr(self, "_all_roots"):
166             self._add_all_roots()
167         return self._all_roots
168 
169     # private: computes the roots of all irreducible factors over the
    respective extension field
170     # Returns a list (sorted in the same order as the list self._factors)
    of the form
```

```
171     # [(factor, q^k, multiplicity, [root1, root2, root3]), ... ]
172     def _add_all_roots(self):
173         self._all_roots = [(fac[0].list, (fac[0].get_splitting_field()).q,
        fac[1], (fac[0].get_RP_in_splitting_field()).roots()) for fac in self.
        get_factors_asRP()]
174         return
175
176     #private: adds the variable order to polynomial
177     def _add_order(self):
178         if not self.is_irreducible():
179             print("The polynomial "+str(self)+" is not irreducible.
        Therefore has no order.")
180             self._order = 0
181         else:
182             self._order = ((self.get_RP_in_splitting_field()).root()).
        multiplicative_order()
183         return
184
185     #public: Returns the order of the polynomial
186     def get_order(self):
187         if not hasattr(self,"_order"):
188             self._add_order()
189         return self._order
190
191     #private: Computes and sets the variable primitive to True or False
192     def _add_primitive(self):
193         self._primitive = (self.get_order()==((self._RFF).q)^self.deg-1)
194         return
195
196     #public: Return True/False whether polynomial is primitive
197     def get_primitive(self):
198         if not hasattr(self, "_primitive"):
199             self._add_primitive()
200         return self._primitive
201
202     # public: Returns the weight of the polynomial
203     def get_weight(self):
204         if not hasattr(self,"_weight"):
205             self._add_weight()
206         return self._weight
207
208     #private: compute weight of polynomial -> package multiset needed!
209     def _add_weight(self):
210         #Version without multiset dependence since this package is missing
        from the SageMath version of Python
211         self._weight = sum([1 for a in self.list if not a==0])
212         return
213
214     # public: Returns the positive integer k such that the polynomial is k-
        normal
215     def get_knormal(self):
216         if hasattr(self,"_knormal"):
217             return self._knormal
218         else:
219             self._add_knormal()
220             return self._knormal
221
222     #private: computes the k-normality of the polynomial and stores it in
        self._knormal
223     def _add_knormal(self):
224         alpha = (self.get_RP_in_splitting_field()).root()
225         f=((self.get_splitting_field()).x)^(self.deg)-1
226         g=sum([alpha^((self._RFF).q^i)*((self.get_splitting_field()).x)^(
```

```
    self.deg-1-i) for i in range(self.deg)])
227         self._knormal = (f.gcd(g)).degree()
228         return
229 
230     # Checks if the polynomial is a composition of the form g(X^m)
231     def is_composition(self, m):
232         for i in [j for j in range(len(self.list)) if j%m !=0]:
233             if self.list[i] !=0:
234                 return False
235         return True
236 
237     # Extracts the polynomial g from the composition f = g(X^m)
238     def extract(self, m):
239         if self.is_composition(m):
240             return RichPolynomial([self.list[i] for i in range(len(self.
    list)) if i%m == 0], self._RFF)
241         else:
242             raise ValueError("The polynomial "+str(self)+" is no
    composition of the form g(X^"+str(m)+").")
243         return
244 
245     # This function constructs m_{beta^p} from self, where beta is a root
    of self and where p is a prime dividing (q-1)
246     # It is an implementation of Corollary 8 from the paper <<Constructing
    irreducible polynomials recursively with a reverse composition method>>
     by Anna-Maurin Graner and Gohar M. Kyureghyan
247     def construction_mbetap(self, p):
248         # checks if p divides q-1:
249         if p not in (self._RFF).get_primefactors():
250             raise ValueError(str(p)+" is not a prime factor of "+str((self.
    _RFF).q)+"-1.")
251 
252         if self.is_composition(p):
253             # if f = g(X^p), then m_{beta^p}(X^p)=f
254             m_betap_Xp = self
255         else:
256             # if f is no composition with X^p, then m_{beta^p}(X^p) is
    computed with formula:
257             m_beta = self.get_polynomial()
258             zeta_p = (self._RFF).get_unityroot(p)
259 
260             m_betap_Xp = (-1)^(self.deg*(p+1)) * prod([m_beta(zeta_p^j*(
    self._RFF).x) for j in range(p)])
261             m_betap_Xp = RichPolynomial(m_betap_Xp, self._RFF)
262 
263         return m_betap_Xp.extract(p)
```

The following script shows briefly how to use the two classes `RichFiniteField` and `RichPolynomial`:

Source Code B.3: RichTutorial.sage

```
1 from RichFiniteFieldClass import *
2 from RichPolynomialClass import *
3 
4 q = 4
5 
6 # Create a RichFiniteField instance of size q, with generator displayed as
    "a" - F_q = F_p(a)- and a polynomial ring in the variable displayed as
    "X":
7 RFF = RichFiniteField(q, "a", "X")
8 a = RFF.gen
9 
10 # Get all information on this RichFiniteField
```

```
print(RFF)

# The SageMath finite field GF(q) hidden inside of the RichFiniteField can be accessed with:
print(RFF.F)

# The RichFiniteField already contains a Polynomial Ring in the variable "X"
# Its generator can be accessed with
x = RFF.x

# Create a RichPolynomial f over RFF:
# f = X^2+a
f = RichPolynomial([1,1,"a"], RFF)

# Get all information on f:
print(f.get_full_info())

# or every property separately:
print(f.get_weight())
print(f.get_order())
print(f.get_knormal())
print(f.get_primitive())
print(f.is_irreducible())

# Factorization:
print(f.get_factorization())
print(f.get_factors_asRP())

# Get roots of f over F_q:
print(f.roots())

# Create another RichPolynomial g over RFF:
g = RichPolynomial(x^3+a^2*x, RFF)
print(g.get_full_info())
print(g.roots())

# Basic arithmetic operations are implemented for RichPolynomials:
print(f+g)
print(f*g)
print(f==g)

# Lets create an extension of RFF:
k = 2
REF = RichExtensionField(RFF, k, "b", "Y")
print(REF)

# Cast f to the extension field and consider it as a polynomial over F_{q^k}
f_inREF = REF.pol_to_extf(f)
print(f_inREF)

# Get roots over F_{16}:
print(f_inREF.roots())

# It is not necessary to build the splitting field of f in general, it is already implemented in f:
print(f.get_all_roots())

# If you wish to access the splitting field:
print(f.get_splitting_field())
```

B.2 Useful functions

The following SageMath file contains a collection of functions which are used by the programs to come.

Source Code B.4: UsefulFunctions.sage

```
# ****************************************************************************
#       Copyright (C) 2023 Anna-Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ****************************************************************************

import itertools

def pause():
    input("\n(Press ENTER to continue and CTR+C to stop)")
    return

#_____________________________________________________
# Functions in number theory:

# function splits the positive integer n into two positive integers n' and m such that gcd(n',q)=1 and rad(m)|q
# where q=p^l is a prime power
def make_gcd_one(q,n):
    p=((q.factor())[0])[0]
    l=n.valuation(p)
    return (int(n/p^l), p^l)

# function checks whether a given integer is a prime power
def is_prime_power(q):
    if q==0:
        return False
    else:
        if len(factor(q))==1:
            return True
        else:
            return False
    return

# function returns the radical of a positive integer n:
def rad(n):
    return  prod([fac[0] for fac in factor(n)])

# function returns all k-subsets of the set s:
def k_subsets(s,k):
    return set(itertools.combinations(s,k))

# This function returns the cyclotomic coset of i mod d over Fq
def cycl_coset(q,d,i):
    return [i*q^j % d for j in range((Mod(q,d/(gcd(d,i))).multiplicative_order()))]

# This function returns a representative system for the q-cyclotomic classes modulo d
def cycl_coset_repsystem(q,d):
    if d==1:
        coset_reps = {0}
    elif d>1:
```

```
        coset_reps = set()
        tbd = list(range(0,d))
        while len(tbd)>0:
            i = tbd[0]
            coset_reps.add(i)
            tbd.remove(i)
            cyclel = i*q % d

            while not cyclel == i:
                tbd.remove(cyclel)
                cyclel = cyclel*q % d
    return coset_reps

#_____________________________________________________
# Functions in finite fields:

# inverse function on a finite field
def inverse(a,F):
    return a^(F.cardinality()-2)

#_____________________________________________________
# Functions for polynomials:

# f(X^n) for f given as a list:
def composition_Xn(f,x, n):
    k=len(f)
    return sum([f[i]*x^(n*(k-1-i)) for i in range(k)])

# function computes the q-spin of a given polynomial pol in the variable x
# coefficient degree is already given as cd
def spin(q,x,pol,cd):
    poll = pol.list()
    spin = pol
    for j in range(1,cd):
        spin = spin * sum([poll[i]^(q^j)*x^i for i in range(len(poll))])
    return spin

# function computes the q-spin of a binomial X^t-b given as (x,t,b)
# coefficient degree is already given as cd
def spin_binomial(q,x,t,b,cd):
    spin =0
    for i in range(cd+1):
        bproduct = sum([prod([b^(q^j) for j in J]) for J in k_subsets(range
    (cd),cd-i)])
        spin = spin+x^(t*i)*(-1)^(cd-i)*bproduct
    return spin

# function casts polynomial in X^d to the basefield of REF
def pol_inXd_to_basefield(pol, REF, x, d):
    pol = pol.list()
    pol = [pol[l] for l in range(len(pol)) if l%d ==0]
    pol = sum(REF.to_basef(pol[l])*x^(d*l) for l in range(len(pol)))
    return pol


# function computes the q-spin of the linear factor X-b given as x,b
# with the explicit formula based on k-subsets
# coefficient degree is already given as cd
def spin_linear_factor_explicit(q,x,b,cd):
    spin =0
    for i in range(cd+1):
        bproduct = sum([prod([b^(q^j) for j in J]) for J in k_subsets(range
    (cd),cd-i)])
```

```
116         spin = spin+x^(i)*(-1)^(cd-i)*bproduct
117     return spin
118
119 # functions prints the factorization as Magma does
120 def print_magma_style(factrztn):
121     print(str_magma_style(factrztn))
122     return
123
124 # function returns a string of a Factorization object in Magma-style
125 def str_magma_style(factrztn):
126     return "\t"+"\n\t".join(["<"+str(fact[0])+","+str(fact[1])+">" for fact
     in factrztn])
127
128 # function returns a string of a polynomial for use with LaTeX:
129 def latex_pol(f,x):
130     f=f.list()
131     s=""
132     for i in range(len(f)):
133         if not f[i]==0:
134             if not f[i]==1:
135                 sn = str(f[i])
136             else:
137                 sn =""
138
139             if not i in [0,1]:
140                 sn=sn+x+"^{"+str(i)+"}"
141             if i==0:
142                 if f[i]==1:
143                     sn = str(f[i])
144             if i ==1:
145                 sn = sn+x
146
147             if len(s)>0:
148                 s=sn+"+"+s
149             else:
150                 s=sn
151     return s
```

B.3 Generating irreducible polynomials to begin with

Some of the programs in sections B.4 and B.5 need an irreducible polynomial over $\mathbb{F}_q$ to begin with. For a comprehensive list of the irreducible polynomials of degree ≤ 11 over $\mathbb{F}_2$, of degree ≤ 7 over $\mathbb{F}_3$, of degree ≤ 5 over $\mathbb{F}_5$ and of degree ≤ 4 over $\mathbb{F}_7$ we refer to [LN94, pp. 384-394].

For finding polynomials of higher degree over a larger field, we present the following program `IrreducibleGenerator.sage`. It computes monic irreducible polynomials of a given degree k over a finite field $\mathbb{F}_q$. For every computed polynomial the program gives its weight, its order, its k-normality and information on whether the polynomial is primitive or not.

Look for `USER INPUT` to choose the finite field size q and the degree k. The variable `maxnumber` can be set to `False` if you wish to generate all monic irreducible polynomials of degree k over $\mathbb{F}_q$. If you set `maxnumber=100` then at most 100 monic irreducible polynomials of degree k over $\mathbb{F}_q$ are computed. The results are written to a `.csv`-file called `irreducible_pols_over_Fq_deg_k.csv` in the folder specified in the variable `filepath`. A `.csv`-file can be opened with LibreOffice Calc, Microsoft Excel or any text editor. When opening this `.csv`-file make sure that in LibreOffice Calc you specify the separator to be `\t`. You can also change the variable `sep` to any other separator.

The program calls the function `minimal_polynomial_over_basefield` from the class

`RichExtensionField` and actually computes the minimal polynomials over $\mathbb{F}_q$ for all elements in $\mathbb{F}_{q^k}$. Thus, the computations are done in the extension field $\mathbb{F}_{q^k}$. For large field sizes q and large positive integers k this might become problematic, but for small examples this program is quite handy.

Source Code B.5: IrreducibleGenerator.sage

```
# ********************************************************************
#       Copyright (C) 2023 Anna-Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ********************************************************************

# This program computes all monic irreducible polyomials over F_q of given degree k
# for every polynomial its weight, its order, its k-normality and whether it is primitive are given

# OUTPUT: a .csv-file detailing the irreducible polynomials

import datetime
from RichFiniteFieldClass import *
sep = "\t"

def main():
    # USER INPUT: Finite field size q and degree k
    q=64
    k=5 # degree of the monic irreducible polynomials, which are generated

    maxnumber = 100  # maximum number of polynomials to be generated
    # Set maxnumber=False if you wish to generate all monic irreducible polynomials of degree k over Fq

    # USER INPUT: Filepath
    filepath = "/put/your/filepath/here/"

    print("This program computes all monic irreducible polyomial over F_{"+str(q)+"} of degree "+str(k)+".\n\n")
    RFF = RichFiniteField(q, "a", "X")
    print(RFF)

    #Initialization of file
    dt = str(datetime.datetime.now())
    filename = "irreducible_pols_over_F"+str(q)+"_deg_"+str(k)+".csv"
    file = open(filepath+filename,"w")
    s= sep+"Polynomial"+sep+"Weight"+sep+"Order"+sep+"Primitive"+sep+"k-normal\n"
    file.write(dt+"\n\nIrreducible polynomials \nover a\n"+str(RFF)+"\nwith modulus: "+sep+str(RFF.get_modulus())+"\nof degree: "+str(k)+"\n\n"+s)
    print(s)

    i=1
    REF = RichExtensionField(RFF,k,"b","Y")
    for gamma in REF.F:
        pol = REF.minimal_polynomial_over_basefield(gamma)
        if pol.deg==k:
            s=str(i)+sep+str(pol.list)+sep+str(pol.get_weight())+sep+str(pol.get_order())+sep+str(pol.get_primitive())+sep+str(pol.get_knormal())
            file.write("\n"+s)
```

```
            print(s)
            i=i+1
            if not maxnumber==False and i>maxnumber:
                break

    print("\nThe irreducible polynomials and their properties have been
    written to the file <<"+filename+">> \nwhich can be found in the folder
     \n<<"+filepath+">>.\n")

    return

if __name__=="__main__":
    main()
```

In order to verify some of the statements in Chapter 3 we need a monic irreducible polynomial f of form $g(X^m)$ for a monic irreducible polynomial $g \in \mathbb{F}_q[X]$ and a positive integer m to begin with. The following SageMath script `CompositionGenerator.sage` generates all monic irreducible compositions of the form $f = g(X^m)$ of degree `mindeg` $\leq \deg(f) \leq$ `maxdeg` for a fixed positive integer m over a fixed finite field of size q (look for `USER INPUT` to change the parameters).

Source Code B.6: CompositionGenerator.sage

```
# ****************************************************************************
#       Copyright (C) 2023 Anna-Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ****************************************************************************

#Script to look for irreducible polynomials satisfying f=g(X^m) for g in F_
    q[X]

from RichFiniteFieldClass import *
from RichPolynomialClass import *

#_____________________________________________

# USER INPUT:
q=4  # Field size q

m=3  # f=g(X^m)

mindeg = 1 # minimum degree to be considered

maxdeg=6 # maximum degree to be considered

#_____________________________________________

F = RichFiniteField(q, "a", "X")

print("Looking for irreducible compositions f=g(X^m) over F_"+str(q)+"\nq-1
     = "+str(q-1)+" = "+str((q-1).factor())+"\nm = "+str(m)+"\nminimum
    degree: "+str(mindeg)+", maximum degree: "+str(maxdeg)+"\n")

found = 0

f_vecs = [[F.one]]

for i in range(1,maxdeg+1):
```

```
    if i%m==0:
        f_vecs_new = []
        for a in F.F:
            f_vecs_new = f_vecs_new +[f_vec+[a] for f_vec in f_vecs]
        f_vecs = f_vecs_new

        if i >= mindeg:
            for f_vec in f_vecs:
                if f_vec[-1] == F.zero:
                    pass
                else:
                    fv = f_vec.copy()
                    fv.reverse()
                    pol = RichPolynomial([(fv[0]^(-1))*fv[j] for j in range
    (len(fv))], F)
                    if pol.is_irreducible():
                        print(pol)
                        print("\torder = "+str((pol.get_order()).factor()))
                        print("\tq^"+str(pol.deg)+"-1 = "+str((q^(pol.deg)
    -1).factor()))
                        found = found+1
    else:
        f_vecs = [f_vec+[F.zero] for f_vec in f_vecs]
    #print(f_vecs)

print("\nNumber of polynomials found: "+str(found))
```

B.4 Source Code for Chapter 2

The following script computes Examples 2.24, 2.29, 2.31, 2.34, 2.36, 2.42, 2.46, 2.64. Look for `USER INPUT` and select the examples you wish to see. Put them into the list `examples`.

Source Code B.7: Ch2-Example.sage

```
# ****************************************************************************
#       Copyright (C) 2023 Anna-Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ****************************************************************************

# This program computes the examples given in Chapter 2

from RichPolynomialClass import *
from RichFiniteFieldClass import *
from UsefulFunctions import *
from fXnAlgorithm import *

# USER INPUT:
# Select which examples you wish to see:
examples = [24,29,31,34,36,42,46,64]
# Examples available:
e1=24
e2=29
e3=31
e4=34
e5=36
e6=42
e7=46
```

```
e8=64

def example(i,q):
    print("\n
    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\
    nSageMath computations for Example 2."+str(i)+":\n")
    F = RichFiniteField(q, "a", "X")
    a = F.gen
    x = F.x
    print(F)
    return (F,a,x)

if e1 in examples:
    (F,a,x)=example(e1,7)

    alpha = F.F(2)
    print("a = "+str(alpha)+", ord(a) = "+str(alpha.multiplicative_order())
    )

    for p in [2,3,5,19]: #,13,17,19
        print("_____________________\np="+str(p))
        f = RichPolynomial(x^p-alpha, F)
        roots = f.get_all_roots()
        print(f.get_factorization())
        #print(roots)

        for rs in roots:
            print("\n"+str(rs[0])+":\n\t"+"\n\t".join([str(r[0])+"\torder:
    "+str((r[0]).multiplicative_order()) for r in rs[3]])+"\n\n\tin "+str(
    RichFiniteField(rs[1],"b","Y")))
        pause()

if e2 in examples:
    (F,a,x)=example(e2,7)

    alpha = F.F(2)
    f = RichPolynomial(x^3-alpha,F)
    print(f.get_full_info())
    p=3

    print("_____________________\np="+str(p))
    f = RichPolynomial(x^(3*p)-alpha, F)
    roots = f.get_all_roots()
    print(str(f.get_polynomial())+ " = "+str(f.get_factorization()))

    for rs in roots:
        print("\n"+str(rs[0])+":\n\t"+"\n\t".join([str(r[0])+"\torder: "+
    str((r[0]).multiplicative_order()) for r in rs[3]])+"\n\n\tin "+str(
    RichFiniteField(rs[1],"b","Y")))

    pause()

if e3 in examples:
    (F,a,x)=example(e3,7)
    alpha = F.F(2)
    f = RichPolynomial(x^3-alpha,F)
    print(f.get_full_info())

    for p in [2,19]:
        print("_____________________\np="+str(p))
        f = RichPolynomial(x^(3*p)-alpha, F)
        roots = f.get_all_roots()
        print(str(f.get_polynomial())+ " = "+str(f.get_factorization()))
```

```
85
86          for rs in roots:
87              print("\n"+str(rs[0])+":\n\t"+"\n\t".join([str(r[0])+"\torder:
    "+str((r[0]).multiplicative_order()) for r in rs[3]])+"\n\n\tin "+str(
    RichFiniteField(rs[1],"b","Y")))
88
89          pause()
90
91  if e4 in examples:
92      q=37
93      (F,a,x)=example(e4,q)
94
95      alpha = F.F(11)
96      print("\na="+str(alpha)+", ord(a)="+str(alpha.multiplicative_order()))
97      n=2^2*3^3
98      print("n="+str(n)+"="+str(factor(n)))
99
100     print("\nElements in F_"+str(q)+":")
101     for b in F.F:
102         if not b == F.F(0):
103             print("b="+str(b)+", ord(b)="+str(b.multiplicative_order())+",
    b^3="+str(b^3)+", b^2="+str(b^2))
104     pause()
105
106     f=x-alpha
107     ps = [3,3,3,2,2]
108     for i in range(len(ps)):
109         p=ps[i]
110         f=f(x^p)
111         print("\np="+str(p)+", "+str(f)+"="+str(f.factor()))
112
113     pause()
114
115 if e5 in examples:
116     q=37
117     (F,a,x)=example(e5,q)
118
119     alpha = F.F(6)
120     print("\na="+str(alpha)+", ord(a)="+str(alpha.multiplicative_order()))
121     n=3^4
122     print("n="+str(n)+"="+str(factor(n)))
123
124     print("\nElements in F_"+str(q)+":")
125     for b in F.F:
126         if not b == F.F(0):
127             print(str(b)+", ord(b)="+str(b.multiplicative_order())+", b^3="
    +str(b^3))
128     pause()
129
130     f=x-alpha
131     ps = [3,3,3,3]
132     for i in range(len(ps)):
133         p=ps[i]
134         f=f(x^p)
135         print("\np="+str(p)+", "+str(f)+"="+str(f.factor()))
136     pause()
137
138     print("\nTheorem 2.32:")
139     for v in (3^2).divisors():
140         for j in range(3^2):
141             if gcd(j,v)==1:
142                 print("v="+str(v)+", j="+str(j)+",7^j*a^v ="+str((7^j*6^v)
    %37)+", order = "+str(Mod(7^j*6^v, 37).multiplicative_order()))
```

```
143
144     f = RichPolynomial(x^(3^4)-alpha, F)
145     roots = f.get_all_roots()
146     print("\n"+str(f.get_polynomial())+ " = "+str(f.get_factorization()))
147
148     pause()
149
150 if e6 in examples:
151     q=5
152     (F,a,x)=example(e6,q)
153
154     print("\nElements in F_"+str(q)+":")
155     for c in F.F:
156         if not c==F.F(0):
157             print("c="+str(c)+",ord(c)="+str(c.multiplicative_order()))
158
159     a = F.F(2)
160     e = a.multiplicative_order()
161     print("\na ="+str(a)+", ord(a)="+str(e))
162     n = 2^4*3*7^2
163     print("n="+str(n))
164     n1 = 2^4
165     n2=3*7^2
166     print("n_1="+str(n1)+", n_2="+str(n2))
167     d1 = 2
168     d2 = 21
169     print("d_1="+str(d1)+", d_2="+str(d2))
170     s1 = Mod(q,d1*e).multiplicative_order()
171     print("s_1="+str(s1))
172     s = Mod(q,rad(n)).multiplicative_order()
173     if n%4==0 and q%4 ==3:
174         s = 2*s
175     print("s="+str(s))
176     pause()
177
178     Fs = RichExtensionField(F,s,"b","Y")
179     b = Fs.gen
180     y = Fs.x
181     print(Fs)
182     ainFs = Fs.to_extf(a)
183
184     print("\nRelevant elements in F_"+str(q^s)+": ")
185     for c in Fs.F:
186         if not c==Fs.F(0):
187             #print("c="+str(c)+" ord(c)="+str(c.multiplicative_order()))
188             if c^2 == ainFs:
189                 print("c="+str(c)+", c^2 ="+str(c^2))
190             if c.multiplicative_order()==d2:
191                 print("c="+str(c)+",ord(c)="+str(c.multiplicative_order()))
192
193     binfac = b^5 + 3*b^4 + b^2 + 4*b + 3
194     r=3
195     z21 =3*b^5 + 3*b^3 + 2
196     z2 = F.F(-1)
197     print("\nb="+str(binfac)+", \n\tb^r = b^3 ="+str(binfac^3)+", \n\t(b^r)
    ^(d1*d2)="+str(binfac^(3*2*n2)))
198     print("zeta_21="+str(z21))
199     print("zeta_2="+str(z2))
200     pause()
201
202     ts = (Mod(5,21).multiplicative_order()).divisors()
203     ts.sort()
204     tis = dict()
```

```
    print("\nt_i:")
    for i in range(21):
        for t in ts:
            if i%int(d2/gcd(d2,5^t-1)) ==0:
                tis[i]=t
                print("i="+str(i)+", t_i= "+str(t))
                break
    pause()

    print("\nCyclotomicCosets:\n\nCR_5(21)="+str(cycl_coset_repsystem(5,21)
    ))
    for i in range(21):
        print("C_(5,21)("+str(i)+")="+str(cycl_coset(5,21,i)))

    print("\nRelation on js:")
    for j in range(2):
        print("j="+str(j)+",\t(-1)^jb = "+str(z2^j*binfac)+",\n\t((-1)^jb)
    ^5 = "+str((z2^j*binfac)^5))

    pause()

    print("\nRelation on (j,v,i)s:")
    for v in divisors(7):
        for i in range(21):
            if gcd(i,v)==1:
                coeff = (z21)^i*(z2^j*binfac)^(3*v)
                eqclass = [(m%2,v,(i*q^m)%d2) for m in range(lcm(s1,tis[i])
    )]
                spininFs = RichPolynomial(prod([(y^(int(n1/d1)*vv)-z21^ii*(
    z2^jj*binfac)^(r*vv)) for (jj,vv,ii) in eqclass]),Fs)
                spininFq = Fs.pol_to_basef(spininFs)
                print("[(0,"+str(v)+","+str(i)+")]={"+str(eqclass)+"}\n\t"+
    str(spininFq.get_polynomial()))
    pause()

    fac = factor_fXn(RichPolynomial([1,F.F(-a)],F),n,0)
    fac = list(fac)
    print("\nX^"+str(n)+"-"+str(a)+" =")
    for fact in fac:
        print("\t<"+str(fact[0])+","+str(fact[1])+">,")
    pause()

if e7 in examples:
    q=5
    (F,a,x)=example(e7,q)
    a = F.F(4)
    print("a ="+str(a))
    e = a.multiplicative_order()
    print("ord(a)="+str(e))
    n = 2^5*7^2*13
    print("n="+str(n))
    radn = prod([fac[0] for fac in factor(n)])
    print("rad(n)="+str(radn)+"="+str(factor(radn)))

    print("\nStep 1:")
    for m in (euler_phi(radn)).divisors():
        print("m="+str(m)+",q^m-1="+str(factor(q^m-1)))
        if (q^m-1)%radn ==0:
            print("w="+str(m))
            w=m
            break

    print("\nStep 2:")
```

```
263     if q^w%4==1:
264         s = w
265         print("s="+str(w))
266 
267     print("\nStep 3:")
268     for m in (q-1).divisors():
269         print("a^"+str(m)+"="+str(a^m))
270         if a^m == F.F(1):
271             print("ord(a)="+str(m))
272             e = m
273             break
274     pause()
275 
276     print("\nStep 4:")
277     n1 =1
278     n2 =1
279     for f in factor(n):
280         if e%f[0]==0:
281             n1 = n1*f[0]^(f[1])
282         else:
283             n2 = n2*f[0]^(f[1])
284     print("n1 ="+str(n1)+"="+str(factor(n1)))
285     print("n2 ="+str(n2)+"="+str(factor(n2)))
286 
287     print("\nStep 5:")
288     d1s = gcd(n1, int((q^s-1)/e))
289     d11 = gcd(n1,int((q-1)/e))
290     print("d1s ="+str(d1s)+"="+str(factor(d1s)))
291     print("d11 ="+str(d11)+"="+str(factor(d11)))
292 
293     print("\nStep 6:")
294     if n%4==0 and q%4==3:
295         s1 = int(d1s/(d11*2^(min([int(n/4).valuation(2),int((q+1)/2).
    valuation(2)]))))
296     else:
297         s1 = int(d1s/d11)
298     print("s1 = "+str(s1))
299 
300     print("\nStep7: ")
301     d2s = gcd(n2, q^s-1)
302     print("d2s="+str(d2s)+"="+str(factor(d2s)))
303 
304     print("\nStep 8: ")
305     s2 = Mod(5,d2s).multiplicative_order()
306     print("s2 ="+str(s2))
307     pause()
308 
309     print("\nStep 9:")
310     d2ts = dict()
311     T=[]
312     for t in s2.divisors():
313         d2t = gcd(n2,q^t-1)
314         print("t="+str(t)+", d2t=gcd("+str(factor(n2))+","+str(factor(q^t
    -1))+")="+str(factor(d2t)))
315         if not d2t in d2ts.values():
316             d2ts[t] = gcd(n2,q^t-1)
317             T.append(t)
318     print("T="+str(T))
319 
320     print("\nStep 10: ")
321     print("n2 mod (ord(a)*d1s)="+str(n2%(e*d1s)))
322     r=n2.inverse_mod(e*d1s)
323     print("r="+str(r))
```

```
324
325     print("\nStep 11: ")
326     Fs = RichExtensionField(F,s,"c","Y")
327     b = Fs.gen
328     y = Fs.x
329     print(Fs)
330     print("beta="+str(b)+", ord(beta)="+str(b.multiplicative_order()))
331     pause()
332
333     print("\nStep 12:")
334     zd1s= b^(int((q^s-1)/d1s))
335     zd2s = b^(int((q^s-1)/d2s))
336     print("zeta_{d1}="+str(zd1s))
337     print("zeta_{d2}="+str(zd2s))
338     zd1se=b^(int((q^s-1)/(d1s*e)))
339     print("zeta_{d1*ord(a)}="+str(zd1se))
340
341     print("\nStep 13:")
342     for l in range(d1s*e):
343         if gcd(l,d1s*e)==1:
344             print("l="+str(l))
345             gamma = zd1se^l
346             if gamma^d1s==a:
347                 beta = gamma
348                 print("b="+str(gamma)+", b^d1s="+str(beta^d1s))
349                 break
350
351     print("\nStep 14:")
352     bjs = dict()
353     js = dict()
354     for j in range(d1s):
355         bj = (zd1s^j*beta)^r
356         bjs[j] = bj
357         js [bj] = j
358         print("j="+str(j)+", bj="+str(bj))
359     pause()
360
361     print("\nStep 15:")
362     J = list(range(d1s))
363     R=[]
364     while(len(J)>0):
365         j=J[0]
366         R.append(j)
367         print("j="+str(j))
368         for u in range(s1):
369             bju = (bjs[j])^(q^u)
370             ju = js[bju]
371             J.remove(ju)
372             print("m="+str(u)+", b_"+str(j)+"^{q^"+str(u)+"}=b_"+str(ju)+"=
    "+str(bju))
373     print("R="+str(R))
374
375     print("\nStep 16:")
376     I = list(range(d2s))
377     C=[]
378     tis = dict()
379     cis = dict()
380     while(len(I)>0):
381         i = I[0]
382         C.append(i)
383         ti = min([t for t in T if i%int(d2s/(d2ts[t]))==0])
384         tis[i] = ti
385         ci = lcm(ti,s1)
```

```
386         cis[i] = ci
387         print("i="+str(i)+", t_i="+str(ti)+", c_i="+str(ci))
388         for m in range(ti):
389             im = (i*q^m)%d2s
390             I.remove(im)
391             print("\tm="+str(m)+", i*q^m mod d2s = "+str(im))
392         if not (i*q^ti)%d2s==i:
393             print("Something went wrong.")
394     pause()
395 
396     print("\nStep 17 (Computation of the irreducible factors):")
397     n1d1 = int(n1/d1s)
398     print("\nn1/d1s="+str(n1d1)+"\n")
399     for i in C:
400         for v in divisors((int(n2/d2s))):
401             if gcd(i,v)==1:
402                 for m in range(gcd(tis[i],s1)):
403                     for j in R:
404                         spininFs = spin_binomial(q,y,n1d1*v,zd2s^(i*q^m)*
    bjs[j],cis[i])
405                         spininFq = pol_inXd_to_basefield(spininFs, Fs, x,
    n1d1*v)
406                         print("("+str(j)+","+str(v)+","+str(i)+","+str(m)+"
    ): "+str(spininFq))
407     pause()
408 
409 if e8 in examples:
410     q=2
411     (F,a,x)=example(e8,q)
412 
413     f = RichPolynomial( [1,1,1,0,1,0,1], F)  #x^6 + x^5 + x^4 + x^2 + 1
414     print(f.get_full_info())
415 
416     n=2^3*7*3^3*11^2
417     print("\nn="+str(n)+"="+str(factor(n)))
418     pause()
419 
420     factrztn = factor_fXn(f,n,1)
421     print("\nFactorization of f(X^n):")
422     print_magma_style(factrztn)
423     #print(factrztn)
424 
425 print("\nThank you, these were all the examples you wished to see.")
```

B.4.1 Implementation of Algorithm 2

The SageMath file **fXnAlgorithm.sage** is a Python/SageMath implementation of Algorithm 2. It can be preparsed (see Section 1.3) and the resulting file **fXnAlgorithm.py** can be loaded as a Python module with the command

```
from fXnAlgorithm import *
```

We discuss the performance of our implementation in Section 2.8.

Source Code B.8: fXnAlgorithm.sage

```
1 # ****************************************************************************
2 #       Copyright (C) 2023 Anna–Maurin Graner
3 #
4 # This program is free software: you can redistribute it and/or modify
5 # it under the terms of the GNU General Public License as published by
6 # the Free Software Foundation, either version 3 of the License, or
7 # (at your option) any later version.
```

```
#                   https://www.gnu.org/licenses/
# ****************************************************************************

# This module contains the function factor_fXn which computes the
    factorization of f(X^n) for any positive integer n and any irreducible
    polynomial f over Fq
# with the new algorithm by Anna-Maurin Graner

from RichPolynomialClass import *
from RichFiniteFieldClass import *
from UsefulFunctions import *

# function computes the q-spin of a the linear factor x-b
# coefficient degree is already given as cd
def spin_linear_factor(q,x,b,cd):
    spin = x-b
    for u in range(1,cd):
        spin = spin * (x-b^(q^u))
    return spin

# Cast list in X^t to basefield of tower of two extension fields:
# mode:
#   3 - q_ks-> q_k -> q
#   2 - q_ks -> q_k=q
#   1 - q_k -> q
#   0 - none
def cast_from_qks_to_q(mode, pol, RF_qks, RF_qk, x, d):
    if mode==0:
        return pol(x^d)
    else:
        pol = pol.list()
        if mode == 3:
            return sum(RF_qk.to_basef(RF_qks.to_basef(pol[l]))*x^(l*d) for
    l in range(len(pol)))
        elif mode == 2:
            return sum(RF_qks.to_basef(pol[l])*x^(l*d) for l in range(len(
    pol)))
        elif mode == 1:
            return sum(RF_qk.to_basef(pol[l])*x^(l*d) for l in range(len(
    pol)))
    return

# Factorization of f(X^n) with the new algorithm by Anna-Maurin Graner
# Input: RichPolynomial f, positive integer n
# Output: Factorization of f(X^n)

def factor_fXn(f,n, printing):
    if not f.is_irreducible():
        raise ValueError("The given polynomial f is not irreducible.")
    else:
        RF_q = f.get_RFF()
        q=RF_q.q
        k = f.deg

        # Determine root of f and F_{q^k}
        if k==1:
            RF_qk = RF_q
            alpha = f.root()
        else:
            RF_qk = f.get_splitting_field()
            alpha = (f.get_RP_in_splitting_field()).root()

        q_k = RF_qk.q
```

```
66
67          # Determine nu_p(n) and tilde-alpha
68          p_to_l = 1
69          # Case gcd(q,n)>1: Using remark of Chapter 2 we consider b instead
    of alpha where b^(p^l)=alpha
70          if gcd(q_k, n)>1:
71              (n, p_to_l) = make_gcd_one(q_k, n)
72              alpha = alpha.nth_root(p_to_l)
73
74          # Determine ord(alpha)
75          e = alpha.multiplicative_order()
76
77          # Determine F_{q^s}, the extension field such that rad(n)|(q^s-1)
    and 4 !| n or q^s = 1 mod 4
78          w = Mod(q_k,rad(n)).multiplicative_order()
79          if (not n%4==0) or q_k^w%4 ==1:
80              s = w
81          else:
82              s = 2*w
83
84          if s>1:
85              RF_qks = RichExtensionField(RF_qk, s , "y", "T")
86              alpha = RF_qks.to_extf(alpha)
87          else:
88              RF_qks = RF_qk
89
90          # Determine n_1, n_2, d_1, d_2
91          q_ks = RF_qks.q
92          n_1 = 1
93          n_2 = 1
94          for fac in factor(n):
95              if e%fac[0] ==0:
96                  n_1 = n_1 * (fac[0])^(fac[1])
97              else:
98                  n_2 = n_2 * (fac[0])^(fac[1])
99
100         d_1s = gcd(n_1, int((q_ks-1)/e))
101         d_11 = gcd(n_1, int((q_k-1)/e))
102
103         # Determine s_1:
104         if (not d_1s%4==0) or q_k%4 ==1:
105             s_1 = int(d_1s/d_11)
106         else:
107             s_1= int(d_1s/(d_11*2^(min([(Integer(n/4)).valuation(2), (
    Integer((q_k+1)/2)).valuation(2)]))))
108
109         #Determine d_2^s:
110         d_2s = gcd(n_2, q_ks-1)
111         s_2 = Mod(q_k, d_2s).multiplicative_order()
112
113         # Determine T and d_2t s:
114         ts = s_2.divisors()
115         ts.sort()
116         d2sd2ts = dict()
117         for l in range(len(ts)):
118             d2sd2t = int(d_2s/gcd(d_2s, q_k^(ts[l])-1))
119             if d2sd2t not in d2sd2ts.values():
120                 d2sd2ts[ts[l]] =  d2sd2t
121         ts = list(d2sd2ts.keys())
122
123         # Determine r:
124         if e==1:
125             r=1
```

```
126         else:
127             r= n_2.inverse_mod(e*d_1s)
128 
129         # Determine gamma:
130         #gamma = (RF_qks.F).multiplicative_generator()
131         #print("gamma="+str(gamma))
132 
133         # Find primitive unity roots
134         zeta_d_1s = RF_qks.get_unityroot(d_1s)
135         zeta_d_2s = RF_qks.get_unityroot(d_2s)
136         zeta_d_1s_e = RF_qks.get_unityroot(d_1s*e)
137 
138         # Find beta:
139         for l in range(d_1s*e):
140             if gcd(l,d_1s*e)==1:
141                 if zeta_d_1s_e^(l*d_1s) == alpha:
142                     beta = zeta_d_1s_e^l
143 
144         # Compute beta_j:
145         bjs = dict()
146         js = dict()
147         for j in range(d_1s):
148             #bj = (zeta_d_1s^j*beta)^r
149             bj = (zeta_d_1s^j*beta)^r
150             bjs[j] = bj
151             js[bj] = j
152 
153         #
    ------------------------------------------------------------------------
154         # Determine a representative system of the equivalence classes of
    ~:
155         if s_1 ==1:
156             j_repsystem = set(range(d_1s))
157         else:
158             j_repsystem = set()
159             J = set(range(d_1s))
160             while len(J)>0:
161                 j = J.pop()
162                 j_repsystem.add(j)
163                 J = J.difference(set([js[bjs[j]^(q_k^l)] for l in range(s_
    1)]))
164 
165         # Cyclotomic representatives and t_is:
166         if s_2 ==1:
167             i_repsystem = set(range(d_2s))
168             tis = dict([(i,1) for i in i_repsystem])
169             cis = dict([(i,s_1) for i in i_repsystem])
170         else:
171             i_repsystem = set()
172             tis = dict()
173             cis = dict()
174             I = set(range(0,d_2s))
175             while len(I)>0:
176                 i= I.pop()
177                 i_repsystem.add(i)
178                 for l in range(len(ts)):
179                     if i%d2sd2ts[ts[l]]==0:
180                         tis[i] = ts[l]
181                         cis[i] = lcm(s_1, ts[l])
182                         break
183                 I= I.difference(set([(i*q^(k*l))%d_2s for l in range(tis[i
    ])]))
184 
```

```
185         n1d1 = int(n_1/d_1s)
186         irred_factors = []
187 
188         if s>1:
189             if k>1:
190                 casting = 3
191             else:
192                 casting =  2
193         else:
194             if k>1:
195                 casting =1
196             else:
197                 casting =0
198 
199         if printing:
200             print("n="+str(n)+"="+str(factor(n)))
201             if gcd(q_k, n)>1:
202                 print("n = "+str(n)+"*"+str(p_to_l))
203                 print("beta = "+str(alpha)+" satisfies beta^"+str(p_to_l)+"
    ="+str(alpha^p_to_l))
204             print("k="+str(k))
205             print("q^k = "+str(q_k)+"="+str(factor(q_k)))
206             print("F_q^k: "+str(RF_qk))
207             print("alpha="+str(alpha))
208             print("ord(alpha) = "+str(e))
209             print("rad(n)="+str(factor(rad(n))))
210             print("rad(n)|q^w-1 for w="+str(w))
211             print("s = "+str(s))
212             print("q^s-1 = "+str(RF_qk.q^s-1)+" = "+str(factor(RF_qk.q^s-1)
    ))
213             print("n_1 = "+str(n_1)+"="+str(factor(n_1)))
214             print("n_2 = "+str(n_2)+"="+str(factor(n_2)))
215             print("d_1s = "+str(d_1s)+"="+str(factor(d_1s)))
216             print("d_11 = "+str(d_11)+"="+str(factor(d_11)))
217             print("s_1="+str(s_1))
218             print("d_2s="+str(d_2s)+"="+str(factor(d_2s)))
219             print("s_2 = "+str(s_2))
220             print("d_2s/d_2t : "+str(d2sd2ts))
221             print("r="+str(r))
222             print(RF_qks)
223             print("zeta_{d_1s}="+str(zeta_d_1s))
224             print("zeta_{d_2s}="+str(zeta_d_2s))
225             print("zeta_{d_1s_e}="+str(zeta_d_1s_e))
226             print("beta="+str(beta))
227             print("bjs = "+str(bjs))
228             print("j_reps : "+str(j_repsystem))
229             print("i_reps : "+str(i_repsystem))
230             for i in i_repsystem:
231                 print("i="+str(i))
232                 print("t_i="+str(tis[i]))
233                 print("c_i="+str(cis[i]))
234                 print("C_q,d_2s(i)="+str(set([(i*q^(k*l))%d_2s for l in
    range(tis[i])])))
235             print("casting method: "+str(casting))
236 
237         # Product over j, v, i, m
238         for j in j_repsystem:
239             for v in divisors(int(n_2/d_2s)):
240                 for i in i_repsystem:
241                     if gcd(i,v)==1:
242                         for m in range(gcd(s_1,tis[i])):
243 
244                             fact = spin_linear_factor(q,RF_qks.x,zeta_d_2s
```

```
    ^(i*q_k^m)*bjs[j]^v,cis[i]*k)
245
246                             fact = cast_from_qks_to_q(casting, fact, RF_qks
    , RF_qk, RF_q.x, n1d1*v)
247
248                             if printing:
249                                 print("(j,v,i,m)=("+str(j)+","+str(v)+","+
    str(i)+","+str(m)+")\n\t"+str(fact))
250                                 #print("("+str(j)+","+str(v)+","+str(i)
    +","+str(m)+")&: "+latex_pol(fact,"X"))
251                             irred_factors.append((fact,p_to_l))
252
253     return Factorization(irred_factors)
```

With the following program `Ch2-TestNewAlgorithm.sage` our implementation of Algorithm 2 in `fXnAlgorithm` can be tested and compared to the existing SageMath function `factor` for univariate polynomials.

For tuples (q, n, f) the program computes the factorization of $f(X^n)$ over $\mathbb{F}_q$ with the new algorithm or the existing SageMath function. Here $\mathbb{F}_q = \mathbb{F}_p(c)$ is a finite field in c and the polynomial f is given by its coefficient vector i.e. $f = X - c^3$ is given as `[1,"-c^3"]`.

Look for `USER INPUT` to insert your tuples in the variable `qnfs` and to change the variables `filepath, fileID` and `sep` to specify `.csv`-file that the measurements are written to. Each computation is going to be stopped after `timeout` seconds. If you wish to compare the computation times of the new algorithm with the SageMath function factor, then set `comparison=True`. Set `print_factorization=True` if you wish to see the factorization of $f(X^n)$. Otherwise, if you are only interested in the computation times, you can set `print_factorization=False`.

If the variable `parallel_computations` is set to `True`, then the computations are done by multiple processes in parallel using the Python module `multiprocessing`. The computations are stopped after `timeout` seconds with the use of the module `signal`. The module `signal` might not agree with the PARI computations in C for the SageMath function `factor` if you set `comparison=True`. In the documentation of Python 3 for `signal`[1], it says:

A long-running calculation implemented purely in C (such as regular expression matching on a large body of text) may run uninterrupted for an arbitrary amount of time, regardless of any signals received. The Python signal handlers will be called when the calculation finishes.

Thus, if the program seems to run endlessly or your computer memory (RAM) is full, then set `parallel_computations=False`. Then the computations are executed consecutively and the timout is done with the module `multiprocessing`.

Source Code B.9: Ch2-TestNewAlgorithm.sage

```
1  # ********************************************************************
2  #       Copyright (C) 2023 Anna-Maurin Graner
3  #
4  # This program is free software: you can redistribute it and/or modify
5  # it under the terms of the GNU General Public License as published by
6  # the Free Software Foundation, either version 3 of the License, or
7  # (at your option) any later version.
8  #                  https://www.gnu.org/licenses/
9  # ********************************************************************
10
```

[1] `https://docs.python.org/3/library/signal.html` [March 10, 2024]

```
# With this program the implementation fXnAlgorithm of the new factorization algorithm for polynomials of the form f(X^n) by Anna-Maurin Graner can be tested and compared with the existing SageMath function factor

from datetime import datetime
import time
import multiprocessing
import signal
from RichPolynomialClass import *
from RichFiniteFieldClass import *
from fXnAlgorithm import *

def main():
    print("This program tests the implementation of the new factorization algorithm for polynomials of the form f(X^n) by Anna-Maurin Graner and compares it with the existing SageMath function factor (if you wish).")

    # USER INPUT:
    filepath = "/put/your/filepath/here/"
    fileID = "your_file_id"
    sep = "\t"

    timeout = 30 # timeout of the computation after timeout seconds
    comparison = True # comparison with the SageMath algorithm (PARI)?
    print_factorization = True # Shall the factorization be printed?

    parallel_computations = False  # Do you wish to do the computations simultaneously or not?
    # IMPORTANT: If the computation seems to run endlessly or your computer is out of RAM, then set parallel_computations=False. Otherwise, set parallel_computations=True

    # Polynomials given as follows: (q,n,f), where  f is the coefficient vector of an irreducible polynomial over F_q = F_p(c)
    # f = X^2+c <-> f=[1,0,"c"]

    qnfs = [(16,11^3,[1,"-c^3"]), (2,19^2,[1,1,1,0,1,0,1])]
    # Table 2.8:
    #[(2, 2^3*7*3^3*11^2, [1,1,1,0,1,0,1]), (2,19^2,[1,1,1,0,1,0,1]),(2,19^3,[1,1,1,0,1,0,1]), (2,19^4,[1,1,1,0,1,0,1]),(2,19^5,[1,1,1,0,1,0,1]), (2,19^6,[1,1,1,0,1,0,1])]
    # Table 2.11:
    #[(4, 2^3*7*3^3*11^2, [1,0,0,1,0,0,"c"]),(4,19^2,[1,0,0,1,0,0,"c"]),(4,19^3,[1,0,0,1,0,0,"c"]), (4,19^4,[1,0,0,1,0,0,"c"]),(4,19^5,[1,0,0,1,0,0,"c"]),(4,19^6,[1,0,0,1,0,0,"c"])]
    # Table 2.12:
    #[(16,11^3,[1,"-c^3"]),(16,1321,[1,"-c^3"]),(16,4513,[1,"-c^3"]),(16,4177, [1,"-c^3"]),(16,3^4*5^4,[1,"-c^3"]),(16,3^5*5^5,[1,"-c^3"]),(16,3^6*5^6,[1,"-c^3"]),(16,3^7*5^7,[1,"-c^3"])]

    # END of USER INPUT. Do not change anything after here.

    print("\ntimeout after: \t\t\t"+str(timeout)+" seconds\nparallel computations: \t\t"+str(parallel_computations)+"\ncomparison with SageMath: \t"+str(True)+"\nprinting factorization: \t"+str(print_factorization)+"\n\nPlease be patient - measurements are in progress.\n")

    if comparison:
        print("Info: Due to the comparison with the SageMath function, the polynomial f(X^n) needs to be computed explicitly and this can take a long time. Don't worry if you see an error thrown by the SageMath
```

```
    function (PARI). Unfortunately, we cannot suppress all error messages
    but the program will terminate properly.\n")
53
54      measurements(qnfs, timeout, comparison, sep, filepath, fileID, parallel
        _computations,print_factorization)
55
56      return
57
58  def computation_time(new, n, f):
59      if not new:
60          f = composition_Xn(f.list,(f.get_RFF()).x,n)
61      wst1 = time.time()
62      cst1 = time.process_time()
63      if new:
64          factorization = factor_fXn(f,n,0)
65      else:
66          try:
67              factorization = f.factor()
68          except NotImplementedError:
69              factorization = False
70      wet1 = (time.time()-wst1)  # in seconds
71      cet1 = time.process_time()-cst1
72      if factorization==False:
73          wet1="stackovf"
74          cet1="stackovf"
75      return (cet1, wet1, factorization)
76
77  def computation_time_to_Queue(new, n, f, que):
78      que.put(computation_time(new, n, f))
79      return
80
81  def computation_time_max_mprocessing(new, n, f, seconds):
82      q = multiprocessing.Queue()
83      p = multiprocessing.Process(target=computation_time_to_Queue, name="
        computation_time_max_fXn_mprocessing", args=(new, n, f, q,))
84      p.start()
85      p.join(timeout=seconds)
86      if not q.empty():
87          result = q.get()
88      else:
89          result = ("infty","infty",False)
90      p.terminate()
91      return result
92
93  class TimeoutException(Exception):   # Custom exception class
94      pass
95
96  def timeout_handler(signum, frame):   # Custom signal handler
97      raise TimeoutException
98
99  def computation_time_max_signal(new, n, f, seconds):
100     # Change the behavior of SIGALRM
101     signal.signal(signal.SIGALRM, timeout_handler)
102
103     signal.alarm(seconds)
104     try:
105         result = computation_time(new, n, f)
106     except TimeoutException:
107         result = ("infty", "infty", False)
108     else:
109         signal.alarm(0)
110     return result
111
```

```
112 def measure_tupl(q,n,f,seconds, comparison, multiprocess, printing):
113     RFF = RichFiniteField(q,"c","X")
114     c = RFF.gen
115     f = RichPolynomial(f, RFF)
116 
117     n_tilde = make_gcd_one(q,n)[0]
118     w=Mod(q^f.deg,rad(n_tilde)).multiplicative_order()
119     if not n_tilde%4==0 or q^(f.deg*w)==1:
120         s=w
121     else:
122         s=2*w
123 
124     string = "\n_____________________________________\nq="+str(q)+", n="+str(
    factor(n))+", f="+str(f.get_polynomial())+",  w="+str(w)+", s="+str(s)+
    "\n"
125 
126     if multiprocess:
127         (ctn, wtn, fac) = computation_time_max_signal(True, n, f, seconds)
128     else:
129         (ctn, wtn, fac) = computation_time_max_mprocessing(True, n, f,
    seconds)
130 
131     if fac==False:
132             print(string+"The new AMG algorithm did not yield a result
    after "+str(seconds)+" seconds.")
133     else:
134         if printing:
135             string2= string+"f(X^n)=\n"+str_magma_style(fac)+"\n\n"
136         else:
137             string2=string
138         print(string2+"New AMG Algorithm: \n\tCPU time: \t"+str(ctn)+"
    seconds\n\tWall time: \t"+str(wtn)+" seconds")
139 
140     if comparison:
141         if multiprocess:
142             (cto, wto, faco) = computation_time_max_signal(False, n, f,
    seconds)
143             string2 = string
144         else:
145             (cto, wto, faco) = computation_time_max_mprocessing(False, n, f
    , seconds)
146             string2 = ""
147 
148         if cto == "stackovf":
149             print(string2+"The SageMath function factor() exited due to a \
    n<<PariError: the PARI stack overflows>>.")
150         elif faco==False:
151             print(string2+"The SageMath function factor() did not yield a
    result after "+str(seconds)+" seconds.")
152 
153         if not fac==False and not faco==False:
154             if not fac==faco:
155                 raise SystemError("\nThe factorizations of the new and the
    SageMath Algorithm are not equal. Something must have gone wrong!")
156             else:
157                 rno = int(round(ctn/cto,0))
158                 ron = int(round(cto/ctn,0))
159         elif faco==False and not fac==False:
160             rno=0
161             ron="infty"
162         elif fac==False and faco==False:
163             rno=False
164             ron = False
```

```
        elif fac==False and not faco==False:
            rno="infty"
            ron =0
            fac=faco
            if printing:
                string2 = string2+"\nf(X^n)=\n"+str_magma_style(fac)+"\n"

        if not faco==False:
            print(string2+"SageMath Algorithm: \n\tCPU time: \t"+str(cto)+"
     seconds\n\tWall time: \t"+str(wto)+" seconds\nratio NA:SM = "+str(rno)
    +":1\nratio SM:NA = "+str(ron)+":1\n")

    if not fac==False:
        max_deg = ((fac[-1])[0]).degree()
    else:
        max_deg = False

    result_str = "\n"+str(q)+sep+str(n)+sep+str(factor(n))+sep+str(f.list)+
    sep+str(f.get_order())+sep+str(w)+sep+str(s)+sep+str(max_deg)+sep+str(
    ctn)

    if comparison:
        result_str = result_str+sep+str(cto)+sep+str(rno)+" : 1"+sep+str(
    ron)+" : 1"
    if printing:
        result_str=result_str+sep+str(fac)

    return result_str

def measurements(qnfs, seconds, comparison, sep, filepath, filename,
    multiprocess, printing):
    filehead = "NewfXnAlg_measurements_"
    file = open(filepath+filehead+filename+".csv", "w")
    file.write("Computation time measurements \nfor the new f(X^n)
    factorization algorithm \nby Anna-Maurin Graner\n\n"+str(datetime.now()
    )+"\n\nq"+sep+"n"+sep+"factor(n)"+sep+"f"+sep+"ord(f)"+sep+"w"+sep+"s"+
    sep+"max deg"+sep+"New Alg (CPU time)")
    if comparison:
        file.write(sep+"SageMath Alg (CPU time)"+sep+"NA:SM"+sep+"SM:NA")
    if printing:
        file.write(sep+"Factorization f(X^n)")

    if multiprocess:
        pool = multiprocessing.Pool()

        for st in pool.starmap(measure_tupl,[(q,n,f,seconds,comparison,
    multiprocess,printing) for (q,n,f) in qnfs]):
            file.write(st)

        pool.close()
    else:
        for (q,n,f) in qnfs:
            file.write(measure_tupl(q,n,f,seconds,comparison,multiprocess,
    printing))

    file.close()
    print("\nThe results of the measurements can be found in "+filepath+
    filehead+filename+".csv.")
    return

if __name__ == "__main__":
    main()
```

The following Magma script has been used to measure the computations in Table 2.8,

Table 2.11 and Table 2.12. In the Magma command prompt it can be executed with the command:

```
> load "./Ch2-MagmaMeasurements.txt";
```

Note that the `Alarm` function quits Magma. Thus, if one computation takes longer than 600 seconds, you need to restart Magma and load the script for the next positive integer n.

Source Code B.10: Ch2-MagmaMeasurements.txt

```
// MAGMA computations for the tables in Chapter 2:

// Table 2.4:
F2:=GF(2);
P2<Y>:=PolynomialRing(F2);

h:= Y^6+Y^5+Y^4+Y^2+1;
print h;
//2^3*7*3^3*11^2,19^2,19^3,19^4,19^5,19^6
for n in [19^2] do
    g:=Evaluate(h,Y^n);
    print "\nn=",Factorization(n);
    Alarm(600);
    time Factorization(g);
end for;


// Table 2.7:
F4<c>:=GF(4);
P4<X>:=PolynomialRing(F4);
print F4;
print "modulus: ",DefiningPolynomial(F4);
f:= X^6+X^3+c;

//2^3*7*3^3*11^2,19^2,19^3,19^4,19^5,19^6
for n in [19^2] do
    g:=Evaluate(f,X^n);
    print "\nn=",Factorization(n);
    Alarm(600);
    time Factorization(g);
end for;

// Table 2.8:
F16<c>:= GF(16);
P16<X>:=PolynomialRing(F16);
print F16;
print "modulus: ",DefiningPolynomial(F16);

//11^3,1321,4513,4177,3^3*5^3,3^4*5^4,3^5*5^5,3^6*5^6,3^7*5^7,
f:= X-c^3;
for n in [11^3] do
    g:=Evaluate(f,X^n);
    print "\nn=",Factorization(n);
    Alarm(1500); //Alarm(1500);
    time Factorization(g);
end for;
```

B.5 Source Code for Chapter 3

The following script computes Examples 3.8, 3.10, 3.12, 3.14, 3.17, 3.20, 3.22, 3.24, 3.28 and 3.29. Look for `USER INPUT`. Some of the numerical results are written to the folder

specified in `filepath`. Please enter a filepath of your choice. Then select the examples you wish to see and change the variable `examples` accordingly.

Source Code B.11: Ch3-Example.sage

```
# ****************************************************************************
#       Copyright (C) 2023 Anna-Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ****************************************************************************

from RichFiniteFieldClass import *
from RichPolynomialClass import *
from ConstructionClasses import *
from UsefulFunctions import *

#_____________________________________________
# USER INPUT:

# Please provide a filepath of your choice:
filepath =  "/put/your/filepath/here"
# Select which examples you would like to compute:
examples = [8, 10, 12, 14, 17, 20, 22, 24, 28, 29]
#_____________________________________________

e1=8
e2=10
e3=12
e4=14
e5=17
e6=20
e7=22
e8=24
e9=28
e10=29

def example(i):
    print("\n
    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\
    nSageMath computations for Example 3."+str(i)+":\n")
    return

#_____________________________________
#Initialization

q=4
F = RichFiniteField(q, "a", "X")
a = F.gen
x = F.x
print(F)

if e1 in examples:
    example(e1)

    f = RichPolynomial([1,a,a], F)
    # Counterexample for roots spread out: [1, a + 1, a + 1, a, a], q=4, n
    =2*5, n=17

    print("\nf = "+f.get_full_info())

```

```
    SF = f.get_splitting_field()

    print("\nSplitting field of f: SF = "+str(SF))

    print("\nAll roots in splitting field: "+str((f.get_all_roots()[0])[2])
    )

    beta = (((f.get_all_roots()[0])[3])[0])[0]

    print("\nbeta = "+str(beta)+" = "+str(SF.in_basis_over_basef(beta)))

    n = 2*5
    hatn = make_gcd_one(q,n)[0]

    print("\nn = "+str(n))

    print("\nbeta^(10) = "+str(beta^n)+" = "+str(SF.in_basis_over_basef(
    beta^n)))

    print("\nm_{beta^(10)} = "+str(SF.minimal_polynomial_over_basefield(
    beta^n)))

    print("\nbeta^5 = "+str(beta^(hatn))+" = "+str(SF.in_basis_over_basef(
    beta^hatn)))

    print("\nm_{beta^5} = "+str(SF.minimal_polynomial_over_basefield(beta^
    hatn)))

    pause()


if e2 in examples:
    example(e2)

    f = RichPolynomial([1,a,a], F)

    print("\nf = "+str(f))

    beta = (((f.get_all_roots()[0])[3])[0])[0]

    print("\nbeta = "+str(beta))

    n=5

    print("\nn = "+str(n))

    print("\nbeta^5 = "+str(beta^(n))+" = "+str(SF.in_basis_over_basef(beta
    ^hatn)))

    chi_betan = SF.characteristic_polynomial_over_basefield(beta^n)

    print("\nchi_{beta^5} \n\t= "+str(chi_betan)+" = "+str(chi_betan.get_
    factorization()))

    print("\nRoots of chi_{beta^5} in Fqtok: \n\t"+str((SF.pol_to_extf(chi_
    betan)).roots()))

    print("\nRoots of chi_{beta^5} in Fq: \n\t"+str(chi_betan.roots()))

    chi_betan_Xn = RichPolynomial((chi_betan.get_polynomial())(F.x^n), F)

    SF_chiXn = chi_betan_Xn.get_splitting_field()

```

```
112     print("\nchi_{beta^5}(X^5) \n\t= "+str(chi_betan_Xn)) #+" \n\t= "+str(
    chi_betan_Xn.get_factorization()))
113
114     print("\nSplitting field of chi_{beta^5}(X^5):\n\tChiSF = "+str(SF_
    chiXn))
115
116     print("\nRoots of chi_{beta^5}(X^5) in ChiSF:  \n\t"+str((SF_chiXn.pol_
    to_extf(chi_betan_Xn)).roots()))
117
118     print("\nFactorization of chi_{beta^5}(X^5) ChiSF: \n\t"+str((SF_chiXn.
    pol_to_extf(chi_betan_Xn)).get_factorization()))
119
120     roots_in_Fq=chi_betan_Xn.roots()
121
122     print("\nRoots of chi_{beta^5}(X^5) in Fq: \n\t"+str(chi_betan_Xn.roots
    ()))
123
124     if len(roots_in_Fq)>0:
125         print("\nRoots in Fq cast into ChiSF: \n\t"+str([(root[0], SF_chiXn
    .to_extf(root[0])) for root in roots_in_Fq]))
126     pause()
127
128 if e3 in examples:
129     example(e3)
130     f = RichPolynomial([1,a,a], F)
131     print("\nf = "+str(f))
132
133     beta = (((f.get_all_roots()[0])[3])[0])[0]
134     print("\nbeta = "+str(beta))
135
136     n=5
137     print("\nn = "+str(n))
138
139     # Extension field where zeta_n lies:
140     ZEF = RichExtensionField(F, Mod(q,n).multiplicative_order(), "c", "Z")
141     z = ZEF.x
142
143     print("\nExtension F(zeta_5):\n\t"+str(ZEF))
144     f_inZEF = ZEF.pol_to_extf(f)
145     print("\nf cast into F(zeta_5): f = "+str(f)+" = "+str(f_inZEF))
146     f_inZEF = f_inZEF.get_polynomial()
147
148     zeta = ZEF.get_unityroot(n)
149     print("\nPrimitive 5-th root of unity in Fq(zeta_5): "+str(zeta))
150
151     print("\nComputation of formula (3.1):")
152     irred_factorization = True
153     chi_betan_Xn_comp = (f_inZEF)
154     for i in range(1,n):
155         next_pol = f_inZEF(zeta^i*z)
156         print("\nj="+str(i)+": \n\tin Fq(zeta): \t"+str(next_pol)+" \n\t\t=
     "+str(next_pol.factor()))
157         try:
158             next_pol_inFq = ZEF.pol_to_basef(RichPolynomial(next_pol, ZEF))
159             print("\tin Fq: \t"+str(next_pol_inFq)+" = "+str(next_pol_inFq.
    get_factorization())+"\n\tirreducible = "+str(next_pol_inFq.is_
    irreducible()))
160             if not next_pol_inFq.is_irreducible():
161                 irred_factorization=False
162         except ValueError:
163             irred_factorization = False
164             print("\tThis polynomial is no polynomial over Fq.")
165             pass
```

```
166
167         chi_betan_Xn_comp = chi_betan_Xn_comp*next_pol
168
169     print("\n(3.1) yields irreducible factors over Fq: "+str(irred_
    factorization))
170
171     print("\nchi_{beta^5}(X^5) in F(zeta_5) computed with formula (3.1): \n
    \t"+str(chi_betan_Xn_comp)+"\n\t\t= "+str(chi_betan_Xn_comp.factor()))
172
173     chi_betan_Xn_basef = ZEF.pol_to_basef(RichPolynomial(chi_betan_Xn_comp,
     ZEF))
174
175     factors_in_basef = chi_betan_Xn_basef.get_factorization()
176
177     print("\nchi_{beta^5}(X^5) in F_4[X]: \n\t"+str(chi_betan_Xn_basef)+" \
    n\t= "+str(chi_betan_Xn_basef.get_factorization()))
178     pause()
179
180
181 if e4 in examples or e6 in examples:
182     example(e4)
183
184     f = RichPolynomial([1, 0, 0, 1, 0, 0, a + 1], F)
185     print("\nf = "+str(f.get_full_info()))
186
187     SF = f.get_splitting_field()
188     print("\nSplitting field of f:\n\tSF = "+str(SF))
189
190     beta = (((f.get_all_roots()[0])[3])[0])[0]
191     print("\nbeta = "+str(beta))
192
193     m = 3
194     print("\nm = "+str(m))
195
196     g = f.extract(m)
197     print("\ng = "+str(g))
198     g = g.get_polynomial()
199
200     betam = beta^m
201     print("\nbeta^m = "+str(betam)+"\nord(beta^m) = "+str(betam.
    multiplicative_order()))
202
203     n = 9
204     print("\nn = "+str(n))
205
206     betan = beta^n
207     print("beta^n = (beta^m)^(n/m) = "+str(betan)+"\nord(beta^n) = "+str(
    betan.multiplicative_order()))
208
209
210     # Extension field where zeta_n lies:
211     ZEF = RichExtensionField(F, Mod(q,n).multiplicative_order(), "c", "Z")
212     print("\nExtension F(zeta_9):\n\t"+str(ZEF))
213
214     zeta = F.get_unityroot(3)
215     print("\nzeta_3 = "+str(zeta))
216
217     pause()
218
219     print("\nComputation of formula (3.2) in F_4:")
220     irred_factorization = True
221     chi_betan_Xn_comp = 1
222
```

```
223     for i in range(1,n+1):
224         print()
225         next_pol = zeta^(-i*2)*g(zeta^i*x^3)
226         print("\nj="+str(i)+": \tzeta_3^("+str(-2*i)+")*g(zeta_3^"+str(i)+"
     X^3): \n\t"+str(next_pol)+" = "+str(next_pol.factor()))
227         if not next_pol.is_irreducible():
228             print("\tnot irreducible over F_"+str(q)+".")
229             irred_factorization=False
230 
231         chi_betan_Xn_comp = chi_betan_Xn_comp*next_pol
232 
233     print("\n(3.2) yields monic irreducible factors over Fq: "+str(irred_
    factorization))
234 
235     print("\nchi_{beta^9}(X^9) in F_q: \n\t"+str(chi_betan_Xn_comp)+"\n\t\t
    = "+str(chi_betan_Xn_comp.factor()))
236 
237     pause()
238 
239 if e5 in examples or e6 in examples:
240 
241     example(e5)
242 
243     f = RichPolynomial([1, 0, 0, 0, 0, a + 1, 0, 0, 0, 0, a + 1, 0, 0, 0,
    0, a, 0, 0, 0, 0, a, 0, 0, 0, 0, a + 1, 0, 0, 0, 0, a + 1], F)
244     print("\nf = "+str(f.get_full_info()))
245 
246     m=5
247     print("\nm = "+str(m))
248 
249     g = f.extract(5)
250     print("\ng = "+str(g))
251 
252     SF = f.get_splitting_field()
253     print("\nSplitting field of f:\n\tSF = "+str(SF))
254 
255     beta = (((f.get_all_roots()[0])[3])[0])[0]
256     print("\nbeta = "+str(beta))
257 
258     n=15
259     print("\nn = "+str(n))
260 
261     zeta = F.get_unityroot(3)
262     print("\nzeta_3 = "+str(zeta))
263 
264     pause()
265 
266     chin_Xn = 1
267     for j in range(1,n+1):
268         next_pol = (g.get_polynomial())(zeta^j*F.x^5)
269         print("\nj="+str(j)+": \t "+str(next_pol)+"\n\tirreducible: "+str(
    next_pol.is_irreducible()))
270         chin_Xn = chin_Xn * next_pol
271 
272     print("\nchi_{beta^(15)}(X^(15)) \n\t= "+str(chin_Xn)+"\n\t= "+str(chin
    _Xn.factor()))
273     pause()
274 
275 if e7 in examples:
276     example(e7)
277 
278     f = RichPolynomial([1, 0, 0, 1, 0, 0, a], F)
279     print("\nf = "+str(f.get_full_info()))
```

```
280
281     n = 3
282     print("\nn = "+str(n))
283
284     k = f.deg
285     print("\nk = "+str(k))
286
287     d = gcd(n,k)
288     print("\nd = gcd(n,k) = "+str(d))
289
290     ms = [m for m in d.divisors() if f.is_composition(m)]
291     print("\nDivisors of d s.t. f=g(X^m): "+str(ms))
292
293     t = max(ms)
294     print("\nt = "+str(t))
295
296     g = f.extract(t)
297     print("\ng = "+str(g))
298
299     zeta = F.get_unityroot(n)
300     print("\nzeta_3 = "+str(zeta))
301
302     print("\nComputation of Theorem 3.20:")
303
304     m_betan_Xn = 1
305     for j in range(1, int(n/t)+1):
306         next_pol = zeta^(-j*k)*(f.get_polynomial())(zeta^j*x)
307         print("\nj = "+str(j)+": "+str(next_pol))
308         m_betan_Xn= m_betan_Xn*next_pol
309
310     print("\nm_{beta^3}(X^3) = "+str(m_betan_Xn))
311
312     print("\nm_{beta^3} = "+str((RichPolynomial(m_betan_Xn, F)).extract(n))
    )
313
314     pause()
315
316     f = RichPolynomial([1, 1, 0, 0, 0, 1, a], F)
317     print("\nf = "+str(f.get_full_info()))
318
319     n = 3
320     print("\nn = "+str(n))
321
322     k = f.deg
323     print("\nk = "+str(k))
324
325     d = gcd(n,k)
326     print("\nd = gcd(n,k) = "+str(d))
327
328     ms = [m for m in d.divisors() if f.is_composition(m)]
329     print("\nDivisors of d s.t. f=g(X^m): "+str(ms))
330
331     t = max(ms)
332     print("\nt = "+str(t))
333
334     g = f.extract(t)
335     print("\ng = "+str(g))
336
337     zeta = F.get_unityroot(n)
338     print("\nzeta_3 = "+str(zeta))
339
340     print("\nComputation of Theorem 3.20:")
341
```

```
342     m_betan_Xn = 1
343     for j in range(1, int(n/t)+1):
344         next_pol = zeta^(-j*k)*(f.get_polynomial())(zeta^j*x)
345         print("\nj = "+str(j)+": "+str(next_pol))
346         m_betan_Xn= m_betan_Xn*next_pol
347 
348     print("\nm_{beta^3}(X^3) = "+str(m_betan_Xn))
349 
350     print("\nm_{beta^3} = "+str((RichPolynomial(m_betan_Xn, F)).extract(n))
    )
351 
352 #_____________________________________#
353 if e8 in examples:
354     example(e8)
355 
356     f = RichPolynomial([1, 0, 0, 1, 0, 0, a], F)
357     print("\nf = "+str(f.get_full_info()))
358 
359     n = 27
360     print("\nn = "+str(n))
361 
362     m_beta3 = f.construction_mbetap(3)
363     print("m_{beta^3} = "+str(m_beta3))
364 
365     m_beta9 = m_beta3.construction_mbetap(3)
366     print("m_{beta^9} = "+str(m_beta9))
367 
368     m_beta27 = m_beta9.construction_mbetap(3)
369     print("m_{beta^(27)} = "+str(m_beta27))
370 
371     pause()
372 
373 #_____________________________________#
374 if e9 in examples:
375     example(e9)
376 
377     f = [1,1,0,0,0,1,a]
378 
379     constr = Construction2(RichPolynomial(f,F), filepath)
380     constr.start_construction()
381     pause()
382 
383     F2 = RichFiniteField(16, "b", "X")
384     b = F2.gen
385 
386 
387     f = [1, 0, 0, b^2 + 1, 0, 0, b + 1]
388 
389     constr2 = Construction2(RichPolynomial(f,F2),filepath)
390     constr2.start_construction()
391     pause()
392 
393 #_____________________________________#
394 if e10 in examples:
395     example(e10)
396     F3 = RichFiniteField(8,"c","X")
397     c = F3.gen
398     f = [1,c,1,c,c^2+c,c^2]
399     constr3 = Construction2(RichPolynomial(f,F3),filepath)
400     constr3.start_construction()
401     pause()
402 
403     constr4 = Construction1(RichPolynomial(f,F3),filepath)
```

```
404     constr4.start_construction()
```

B.5.1 Implementation of Construction 1 and Construction 2

The following module B.12 `ConstructionClasses.sage` can be preparsed and loaded as a Python module (see Section 1.3). It contains the base class `Construction` with the kids `Construction1` and `Construction2`.

The class `Construction1` is built for the computation of all monic irreducible polynomials, which can be obtained from one monic irreducible polynomial $f \in \mathbb{F}_q[X]$ with Construction 1, as discussed in Section 3.4. The polynomials together with their weight, their order and their k-normality are written to a `.csv`-file. An overview of the parameters of the constructed polynomials can be found in a second file with the ending `_ov.csv`.

The class `Construction2` is an implementation of Construction 2. It gives the p-adic valuation of $\mathrm{ord}(f)$ for every prime factor p of $q-1$ and a list of positive integers such that $\mathrm{ord}(f)$ is among them. For some polynomials f the exact order $\mathrm{ord}(f)$ is determined. All results are written to a `.csv`-file in the folder specified by the variable `filepath`.

The polynomial f needs to be given as a list and the finite field $\mathbb{F}_q$ as a `RichFiniteField`. Then the constructions can be used as follows:

```
from RichFiniteFieldClass import *
from ConstructionClasses import *

filepath = "/home/user/Documents/"

F = RichFiniteField(8,"c","X")
c = F.gen
f = RichPolynomial([1,c,1,c,c^2+c,c^2],F)

constr1 = Construction1(f,filepath)
constr1.start_construction()

constr2 = Construction2(f,filepath)
constr2.start_construction()
```

Source Code B.12: ConstructionClasses.sage

```
# ****************************************************************************
#       Copyright (C) 2023 Anna-Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ****************************************************************************

# This is an implementation of the constructions of irreducible polynomials
    presented in the paper
# <<Constructing irreducible polynomials recursively with a reverse
    composition method>> by Anna-Maurin Graner and Gohar M. Kyureghyan

ver = "2024-03-11"

from collections import deque
from RichPolynomialClass import *
from UsefulFunctions import *
from datetime import datetime

sep = "\t" #separator used for the csv-files
```

```
filename = "NewConstruction"

#_____________________________________________________________________

# CLASSES:

# Class Construction is the parent class of Construction 1 and Construction 2
# contains the finite field as RichFiniteField-instance, the initial polynomial as RichPolynomial-instance and a file that the results are written to
class Construction:
    def __init__(self,f, no, filepath):
        self.finite_field = f.get_RFF()
        self.init_pol = f
        self.filename = filename+"_"+str(no)+"_F"+str((self.finite_field).q)+"_"+str((self.init_pol).list)
        self.filepath = filepath
        self.no = no
        if not (self.init_pol).is_irreducible():
            raise ValueError("The polynomial "+str(f)+" is not irreducible.")

        self.file = open(self.filepath+self.filename+".csv","w")
        (self.file).write(str(self))
        print(str(self))
        return

    def __str__(self):
        return "Construction "+str(self.no)+" from the paper\n<<Constructing irreducible polynomials recursively with a reverse composition method>> \nby Anna-Maurin Graner and Gohar M. Kyureghyan\n\nversion"+sep+ver+"\n\ndate"+sep+str(datetime.now())+"\n\nInitial polynomial: "+str(self.init_pol)+"\n\n"+str(self.finite_field)

    def end_construction(self):
        (self.file).close()
        print("\nAll results have been written to the file\n<<"+self.filename+">> \nin the folder: "+self.filepath+".\n_____________________________________________")
        return

#_____________________________________________________________________

class Construction1(Construction):

    def __init__(self, f, filepath):
        super().__init__(f,1, filepath)

        self.ov_file =  open(self.filepath+self.filename+"_ov.csv","w")
        s =  "\nOrbits:\nn"+sep+"order"+sep+"length"
        (self.ov_file).write(str(self)+"\n"+s)
        print(s)

        self.no_of_pols = 0
        self.no_of_orbit_pols = dict() #stores a list with two entries:
        # [the number of irreducible polynomials of given order (as mentioned in Daykin(1965)), the number of constructed polynomials of this order]

        self.parameters = dict() # Sorted by degree the number of weights and normal polynomials is collected.
        self.orbit_reps = dict()  # key is a representant for every orbit
```

```
    and the value is a set of all ks that share the same orbit
71          self.new_orbit_exponents = dict()
72
73          #Compute ord_rp(p) for all p in primefactors where e=p^{nu_p(e)}*rp
74          self.max_exponents = dict()
75          for p in (self.finite_field).get_primefactors():
76              nu_p = ((self.init_pol).get_order()).valuation(p)
77              self.max_exponents[p]=nu_p+(Mod(p,(self.init_pol).get_order()/(
    p^nu_p)).multiplicative_order())-1
78
79          self.nu_pmax = ((self.init_pol).get_order()).valuation(((self.
    finite_field).get_primefactors())[-1]) #nu for maximal primefactor of q
    -1 of order of initial polynomial
80          self.tail_pols = [dict() for i in range(self.nu_pmax)]  # store
    tail_pols sorted by nu_p(ord(f)) = 5^{nu_pmax-i}
81          return
82
83      def str_parameters(self, separator):
84          ds = [d for d in self.parameters]
85          ds.sort()
86
87          return "\nParameters of the "+str(self.no_of_pols)+" constructed
    polynomials:\nDegree"+separator+"Weight"+separator+"Total"+separator+
    separator.join([str(i)+"-normal" for i in range((self.init_pol).deg)])+
    "\n"+"\n".join([str(d)+"\n"+("\n".join([separator+str(w)+separator+str
    (((self.parameters[d])[w])["all"])+separator+separator.join([str(i) for
     i in ((self.parameters[d])[w])["k-normal"]]) for w in self.parameters[
    d]]))   for d in ds])
88
89      def update_no_of_pols(self, i):
90          self.no_of_pols = self.no_of_pols+i
91          return
92
93      def update_parameters(self, pol):
94          d = pol.deg
95          w = pol.get_weight()
96          knormal = pol.get_knormal() #Returns an integer in [0,...,d-1]
97          if d in self.parameters:
98              if w in self.parameters[d]:
99                  ((self.parameters[d])[w])["all"] = ((self.parameters[d])[w
    ])["all"]+1
100                 (((self.parameters[d])[w])["k-normal"])[knormal] = (((self.
    parameters[d])[w])["k-normal"])[knormal]+1
101             else:
102                 (self.parameters[d])[w]= {"all":1,"k-normal":[1 if i==
    knormal else 0 for i in range(d)]}
103         else:
104             self.parameters[d]={w:{"all": 1, "k-normal": [1 if i==knormal
    else 0 for i in range(d)]}}
105
106         return str(pol)+sep+str(w)+sep+str(knormal)
107
108     def start_construction(self):
109
110         #List of constructed polynomials that still need to be considered
    recursively
111         #Stored as tuple (k, k_list, position, m_beta^k) where k_list
    contains the exponents of the primefactors of q-1 in integer k
112         pols = deque([(1, [0 for p in (self.finite_field).get_primefactors
    ()], 0, self.init_pol)])
113         self._recursive_construction(1 ,self.init_pol)
114
115         while(len(pols)>0):
```

```
116             #print([str((pol[0]).factor())+","+str(pol[1])+","+str(pol[2])
    +","+str((pol[3].list)) for pol in pols])
117             # Polynomial is popped (removed) from left end of deque
118             (k, k_list, pos, pol) = pols.popleft()
119             for i in range(pos, len((self.finite_field).get_primefactors())
    -1):
120                 if k_list[i]< self.max_exponents[(self.finite_field).get_
    primefactors()[i]]:
121                     new_k = k*((self.finite_field).get_primefactors()[i])
122                     new_k_list = k_list.copy()
123                     new_k_list[i] = new_k_list[i]+1
124                     new_pol = pol.construction_mbetap((self.finite_field).
    get_primefactors()[i])
125 
126                     # New polynomials are appended at the right end of
    deque
127                     pols.append((new_k, new_k_list, i, new_pol))
128 
129                     # Construction of tail and orbit for the new polynomial
     -> all computations for this polynomial are done, only needed for
    construction of next polynomials
130                     self._recursive_construction(new_k ,new_pol)
131             #end while
132 
133         self.end_construction()
134         return
135 
136     def end_construction(self):
137         (self.file).close()
138 
139         #Write overview file:
140         s = "\nComparison of constructed number with total number of
    irreducible polynomials with orbit-order:\nOrder"+sep+"Total"+sep+"
    Constructed\n"+"\n".join([str(key.factor())+sep+str((self.no_of_orbit_
    pols[key])[0])+sep+str((self.no_of_orbit_pols[key])[1]) for key in self
    .no_of_orbit_pols])
141         print(s)
142         (self.ov_file).write("\n"+s)
143 
144         print(self.str_parameters("\t"))
145 
146         print("\nThe constructed polynomials can be found in the file \n"+
    str((self.file).name))
147 
148         (self.ov_file).write("\n\n"+self.str_parameters(sep))
149 
150         (self.ov_file).close()
151         print("\nAn overview of the results can be found in the file \n"+
    str((self.ov_file).name))
152 
153         return
154 
155     # Generate the exponents of the primefactors and construct the minimal
    polynomial of beta^k recursively
156     # k -   is the current k (just for information!)
157     # pos -         is the position of the exponent which needs to be
    raised for next k
158     # pol -       is m_beta^k for current k
159 
160     def _recursive_construction(self, k, pol):
161         p = ((self.finite_field).get_primefactors())[-1]
162 
163         (self.file).write("\n\nn = "+str(k.factor())+"*"+str(((self.finite_
```

```
    field).get_primefactors())[-1])+"^i")
164
165         # Tail:
166         # Construct tail if nu_pmax > 0
167         if self.nu_pmax > 0:
168
169             (self.file).write("\ni"+sep+"m_beta^n"+sep+"weight"+sep+"k-
    normal\nTail:")
170
171             for i in range(self.nu_pmax):
172                 if tuple(pol.list) in self.tail_pols[i]:
173                     (self.file).write("\n"+str(i)+sep+"same polynomial as n
     in {"+",".join(str(factor(othern)) for othern in (self.tail_pols[i])[
    tuple(pol.list)])+"}.")
174
175
176                 # This is Construction 2 hidden - polynomials for one
    prime factor power - if same as one power before, then set this as new
    max
177                 if i==0 and len(list(k.factor()))==1:
178                     for prev_k in (self.tail_pols[i])[tuple(pol.list)]:
179                         if k % prev_k ==0:
180                             pf = (list(k.factor()))[0]
181                             self.max_exponents[pf[0]] = pf[1]-1
182                             s = "Construction 2 for "+str(pf[0])+"
    yields max exponent "+str(pf[1]-1)+"."
183                             print("\n"+s+"\n")
184                             (self.file).write("\n\n"+sep+s)
185
186                 ((self.tail_pols[i])[tuple(pol.list)]).append(k)
187                 self.update_no_of_pols(i)
188                 return #BREAK: stop construction of successors since
    all successors are equal to previous polynomials
189
190             else:
191                 (self.tail_pols[i])[tuple(pol.list)] = [k]
192                 (self.file).write("\n"+str(i)+sep+self.update_
    parameters(pol))
193
194                 # next construction step
195                 pol = pol.construction_mbetap(p)
196
197         # After tail:
198         self.update_no_of_pols(self.nu_pmax)
199     #orbit:
200     (self.file).write("\nOrbit:")
201
202     #Check if exponents are smaller than new_orbit_exponents:
203     for pf in dict(k.factor()):
204         if pf in self.new_orbit_exponents:
205             if (dict(k.factor()))[pf]>self.new_orbit_exponents[pf]:
206                 (self.file).write("\n"+sep+"n larger than new orbit
    exponents - orbit already constructed.")
207                 return
208
209     pols = []  # first polynomial of orbit has been constructed in last
     step of tail-loop or is m_{beta^n}
210
211     # Construct orbit until the first polynomial appears for the second
     time
212     while(len(pols)<1 or pol.list != (pols[0]).list):
213         # collect constructed polynomials in list pols
214         pols.append(pol)
```

```
215
216             # Check if this polynomial has been constructed in another
    orbit
217
218             if tuple(pol.list) in self.orbit_reps:
219                 # Determine maximal exponent for one-prime-factor-orbits
220                 if len(list(k.factor()))==1:
221                     for prev_k in self.orbit_reps[tuple(pol.list)]:
222                         if k % prev_k ==0:
223                             pf = (list(k.factor()))[0]
224                             self.new_orbit_exponents[pf[0]]=pf[1]-1
225                             print(str(k.factor())+" same orbit as "+str(
    prev_k.factor())+" -> New orbit-max-exponent for "+str(pf[0])+": "+str(
    pf[1]-1)+".")
226                             break
227                 (self.file).write("\n"+sep+"orbit for representant "+str(
    pol.list)+" - same as n in "+str([k.factor() for k in self.orbit_reps[
    tuple(pol.list)]]))
228
229                 # Then add k to the list of orbits which consist of the
    same polynomials
230                 self.orbit_reps[tuple(pol.list)].append(k)
231
232                 return
233
234             # next construction step
235             pol = pol.construction_mbetap(((self.finite_field).get_
    primefactors())[-1])
236
237         #orbit has not been constructed before:
238         self.orbit_reps[tuple(pol.list)] = [k]  #first (and last)
    polynomial is stored in orbit-reps as representant of this orbit
239
240         i=self.nu_pmax
241         for f in pols:
242             (self.file).write("\n"+str(i)+sep+self.update_parameters(f))
243             i=i+1
244
245         # Update number of irreducible polynomials of order of this orbit-
    representant
246         r= pol.get_order()
247         if r in self.no_of_orbit_pols:
248             (self.no_of_orbit_pols[r])[1]= (self.no_of_orbit_pols[r])[1]+
    len(pols)
249         else:
250             self.no_of_orbit_pols[r] = [euler_phi(r)/pol.deg , len(pols)]
251
252         s =(str(k.factor())+"*"+str(((self.finite_field).get_primefactors()
    )[-1])+"^i", str(r.factor()), str(len(pols)))
253         print(s[0]+"\t\t"+s[1]+"\t\t"+s[2])
254         (self.ov_file).write("\n"+s[0]+sep+s[1]+sep+s[2])
255         self.update_no_of_pols(len(pols))
256
257         return
258
259 #_____________________________________________________________________
260
261 class Construction2(Construction):
262
263     def __init__(self, f, filepath):
264         super().__init__(f, 2, filepath)
265
266         self.nu_maxes = dict(factor((self.finite_field).q^(self.init_pol).
```

```
    deg-1)) #p-adic valuations for primefactors of q-1 in q^(deg(init_pol))
    -1
267         self.valuations = dict() # p-adic valuations for primefactors of q
    -1 of order(init_pol)
268         self.orbit_lengths = dict()
269         self.candidates = divisors((self.finite_field).q^(self.init_pol).
    deg-1) #list of possible orders
270         return
271
272     def __str__(self):
273         qn1 = (self.finite_field).q^(self.init_pol).deg - 1
274         return (super().__str__())+"\n\nOrder is a divisor of "+str((self.
    finite_field).q)+"^("+str((self.init_pol).deg)+")-1 = "+str(qn1)+" = "+
    str(qn1.factor())
275
276     # This function is the exact implementation of Construction 2  for a
    prime integer p dividing q-1
277     # it returns the parameters nu_p(ord(f)) and the length of the orbit
278     def nu_p_of_order(self, p):
279         first_pols = [(self.init_pol).list] #polynomials f_0,...,f_nu_max
    are stored here
280         i=0
281         pol = self.init_pol
282
283         while True:
284             i=i+1
285             pol = pol.construction_mbetap(p)
286             try:  #If polynomial is in list of first polynomials, then
    orbit ends
287                 nu = first_pols.index(pol.list)
288                 # Return tuple with information (nu_p(ord(init_pol)), orbit
    _length)
289                 return (nu, i-nu)
290             except ValueError:
291                 if i<=self.nu_maxes[p]:
292                     first_pols.append(pol.list)
293         return 1
294
295     def start_construction(self):
296         s = "\np\tnu_p\torbit-length"
297         (self.file).write("\n"+s)
298         print(s)
299
300         # For every prime p dividing q-1 Construction 2 is computed with
    the function self.nu_p_of_order(p)
301         for p in (self.finite_field).get_primefactors():
302             (self.valuations[p], self.orbit_lengths[p]) = self.nu_p_of_
    order(p)
303             s = str(p)+"\t"+str(self.valuations[p])+"\t"+str(self.orbit_
    lengths[p])
304             (self.file).write("\n"+s)
305             print(s)
306
307         # Remove candidates that do not satisfy the conditions of Corollary
     3.25:
308         # remove all possible orders that do not have the correct p-adic
    valuations:
309         for p in (self.finite_field).get_primefactors():
310             self.candidates = [candidate for candidate in self.candidates
    if candidate.valuation(p)==self.valuations[p]]
311
312         if len(self.candidates)>1:
313             for p in (self.finite_field).get_primefactors():
```

```
314                 #print("_____\np="+str(p))
315                 i=0
316                 while i<len(self.candidates):
317                     candidate = self.candidates[i]
318                     #print("candidate: "+str(factor(candidate)))
319 
320                     # remove all possible orders that do not satisfy that
    the orbit length can be an intersection of <q^j> and <p> in Z/ord_p(ord
    (f)/p^nu_p(ord(f)))Z
321                     r = candidate/(p^self.valuations[p])
322                     #print("r = "+str(r))
323                     #print("ord_r(p) = "+str(Mod(p,r).multiplicative_order
    ()))
324                     ord_qj = (Mod(p,r).multiplicative_order())/self.orbit_
    lengths[p]
325                     #print("d = "+str(ord_qj))
326                     if ord_qj == 1:
327                         # if ord_r(p)==s (orbit length), then j=0 satisfies
     the equation because p^s mod r = 1 and q^0 mod r = 1.
328                         i = i+1
329                     else:
330                         if ord_qj not in divisors((self.init_pol).deg):
331                             (self.candidates).remove(candidate)
332                             #print("removed because d not divisor of deg(f)
    ")
333                         else:
334                             p_ol = p^(self.orbit_lengths[p]) %r
335                             #print("p^s mod r = "+str(p_ol))
336                             keep_candidate = False
337                             #find js such that ord(q^j)=ord_qj and <q^j>
    and <p> have intersection at orbit_length
338                             # j=0 only relevant for p^s mod r = 1  <=>  s =
     ord_r(p)
339                             for j in range(1,(self.init_pol).deg):
340                                 if (Mod((self.finite_field).q^j,r)).
    multiplicative_order()==ord_qj and  p_ol == (self.finite_field).q^j % r
    :
341                                     #print("j="+str(j)+" satisfies equation
    ")
342                                     keep_candidate = True
343                                     break
344 
345                             if not keep_candidate:
346                                 (self.candidates).remove(candidate)
347                                 #print("removed because no j satisfied the
    equation")
348                             else:
349                                 i=i+1
350 
351         s="\nord("+str((self.init_pol).list)+")"
352         if len(self.candidates)==1:
353             s += " = "+str(self.candidates[0])
354         else:
355             s += " is in \n{"+",".join([str(candidate.factor()) for
    candidate in self.candidates])+"}."
356         print(s)
357 
358         (self.file).write(s)
359         self.end_construction()
360         return
```

B.5.2 Source code for the computations in [KK22]

The numerical results presented in [KK22, Example 2] have been obtained with the following program, B.13 `Ch3-ConstructionKK.sage`.

For a monic irreducible polynomial $f \in \mathbb{F}_p[X]$ over a prime field $\mathbb{F}_p$ the program computes ConstructionKK if f is not a composition of the form $g(X^2)$ for a monic irreducible polynomial $g \in \mathbb{F}_p[X]$. It writes the constructed polynomials and their weight table to a `.csv`-file in the folder specified by the variable `filepath`. If $f = g(X^2)$ for a monic irreducible polynomial $g \in \mathbb{F}_p[X]$, then the program computes ConstructionKK for $f(X + a)$ for all $a \in \mathbb{F}_p^*$ instead.

The user can enter the field size p and the polynomial $f \in \mathbb{F}_p[X]$ interactively. The polynomial f needs to be specified by its coefficient vector, i.e. $f = X^4 + X^3 + X^2 + 1 \in \mathbb{F}_5[X]$ would be given by `[1,1,1,0,1]`.

Source Code B.13: Ch3-ConstructionKK.sage

```
# ****************************************************************************
#       Copyright (C) 2023 Anna–Maurin Graner
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#                  https://www.gnu.org/licenses/
# ****************************************************************************

# This program computes Construction 1 from "A recurrent construction of irreducible polynomials of fixed degree over finite fields" by Gohar M. Kyureghyan and Melsik K. Kyureghyan

filepath = "/put/your/filepath/here/"

#=============================================
# Functions:

# Input: a coefficient list f_vec of entries in a finite field Fq, a generator var of a polynomial ring over Fq
# Output: a polynomial in the polynoial ring over Fq with coefficients from f_vec
def polynomial(f_vec, var):
    f=0
    for i in range(len(f_vec)):
        f+=f_vec[i]*var^(i)
    return f

# This program generates a string representing a polynomial only given as a coefficient vector - without computing a real polynomial
# Input: a coefficient list f_vec of a polynomial, a character/string var supposed to be a generator of a polynomial ring
# Output: a string of the form c_0+c_1*var+c_2*var^2+...
def print_pol_extended(f_vec, var):
    k=0 #number of terms
    s=""
    for i in range(len(f_vec)):
        j=len(f_vec)-1-i

        if f_vec[j]!=0:
            if k>0:
                s+=" + "

            if j==0:
```

```
                s+= str(f_vec[j])
            else:
                if f_vec[j]!=1:
                    s+=str(f_vec[j])+"*"
                if j==1:
                    s+=var
                else:
                    s+=var+"^"+str(j)
            k+=1
    return s

# This function returns a string representing the polynomial given as coefficient vector f_vec in the variable X
def print_pol(f_vec):
    return print_pol_extended(f_vec ,"X")

# This function returns the weight of a polynomial given as coefficient vector f_vec
def weight(f_vec):
    wt=0
    for c in f_vec:
        if c!=0:
            wt+=1
    return wt

# This function computes the order of a in (Z/nZ)*
# Attention: Only applicable for a such that gcd(a,n)=1
def order_mod(a,n):
    order = 1
    while(a^(order)%n != 1):
        order +=1
    return order

#=============================================================
# Main program
def main():
    sep="\t"

    #-------------------------------
    #Interactive user input of  filename-ID and field size q:
    print("This program computes Construction 1 from <<A recurrent construction of irreducible polynomials of fixed degree over finite fields>> by Gohar M. Kyureghyan and Melsik K. Kyureghyan")

    # If you wish to add a personal ID at the end of your filenames, then unquote the following line instead of the one after that:
    #id = input("Please enter an id for your computation in quotation marks (e.g. "test"). It will be added to the filename: ")
    id=""

    P=Primes()
    q=0
    while(q %2 ==0 or q not in P):
        q=int(input("\nNote: This program can only handle prime fields. \nPlease enter an odd prime as field size q: "))

    #-------------------------------
    #Initialization of the finite field F and its polynomial ring PRF

    F = GF(q,"a")
    PRF = PolynomialRing(F,"x")
    x = PRF.gen()

```

```
95  #-------------------------------
96  # Interactive user input of the initial polynomial as a coefficient
    vector
97  # e.g. [c_0,c_1,..., 1] for the polynomial c_0 * X^0 + c_1 * X + ...+ 1
     * X^n
98  # The polynomial needs to be irreducible over Fq
99
100 irreducible = False
101 n=0
102 while (irreducible == False):
103     # Convert input (which is a string in Python3) to list of integers
104     start_f_vec  = list(map(int, (((input("\nPlease enter the
    coefficient vector of an irreducible polynomial over Fq. \n(eg.
    [1,0,0,0,1,0,1] for X^6+X^2+1)\nf = "))[1:-1]).strip()).split(",")))
105
106     start_f_vec.reverse()
107
108     if (polynomial(start_f_vec, x).is_irreducible() == False):
109         print("\nERROR: The given polynomial f is not irreducible.")
110     else:
111         irreducible = True
112
113 print("\nThe polynomial you gave was: f = "+print_pol(start_f_vec))
114
115 #---------------------------
116
117 n=len(start_f_vec)-1
118
119 satisfies_conditions = True
120
121 if (n%2==0):
122     odd_index = False
123     for i in range(len(start_f_vec)):
124         if(i%2 == 1 and start_f_vec[i]!=0):
125             odd_index = True
126             break
127
128     if(odd_index==False):
129         print("\nFOR YOUR INFORMATION: \nThe polynomial does not have
    an odd index i with c_i!=0. \nTherefore, the program computes
    Construction 1 for f(X+a) for all a!=0 in GF("+str(q)+").")
130         satisfies_conditions = False
131
132 if satisfies_conditions == True:
133     print("\nThe program computes Construction 1 for the given
    polynomial.")
134
135 filename = "nr_ConstructionKK_"+str(q)+"_"+str(start_f_vec)
136 filename_prefix = filepath+filename
137 file_init = sep +"Numerical results obtained with Construction 1 from \
    n"+sep+"<<A recurrent construction of irreducible polynomials of fixed
    degree over finite fields>> \n"+sep+"by Gohar M. Kyureghyan and Melsik
    K. Kyureghyan\n"+sep+"implemented by A.-M. Graner\n\n"+sep+"F=GF("+str(
    q)+")\n\n"
138
139 #-----------------------------------------------
140 # Initialization of the extension field F_{q^n} - needed for the
    computation of the order of the constructed polynomials
141
142 E = GF(q^n, "b")
143 PRE = PolynomialRing(E, "y")
144 y=PRE.gen()
145
```

```
146     #-----------------------------------
147
148     # This variable stores the weight of all constructed polynomials
149     weight_tables = []
150
151     # Computation of Construction 1 if condition (2) is satisfied
152     # Computation of Construction 1 for all f(X+a), a in Fq\{0}, if
        condition (2) is not satisfied
153     # Here a = el
154     for el in F:
155         compute = True
156         if satisfies_conditions == True and el !=0:
157             compute = False
158         if satisfies_conditions == False and el ==0:
159             compute = False
160
161
162         if compute:
163             f_vec = polynomial(start_f_vec, x+el).list()
164
165             if polynomial(f_vec, x).is_irreducible()==True:
166                 #-----------------------------
167                 #Initialization of files
168                 s=filename_prefix+"_X+"+str(el)+id
169
170                 # For every a in Fq a separate .csv-file containing all
        constructed polynomials is written  (otherwise the file would be too
        large to handle)
171                 csv_file = open(s+".csv","w")
172                 csv_file.write(file_init+sep+"Initial polynomial:\n"+sep+
        print_pol_extended(start_f_vec, "(X+"+str(el)+")")+"\n"+sep+" = "+print
        _pol(f_vec)+" in F[X]\n\n"+sep+str(q)+"^"+str(n)+"-1 = "+str(q^n-1)+"\n
        \n"+sep+"Polynomial"+sep+"Weight"+sep+"Order\n")
173
174                 #-----------------------------
175                 #Initialize the while-loop
176
177                 weights = [0 for i in range(n+1)] #Counts number of
        appearances for weights
178
179                 num_odd_order = 0  #Counts the number of polynomials with
        odd order
180                 order_mod_2 = 0 # Will be number of irreducible polynomials
         with odd order
181                 first_orbit_pol = False
182
183                 f_order = 0
184                 f = polynomial(f_vec, x)
185
186                 while (first_orbit_pol==False or not f==first_orbit_pol):
187
188                     #Check if f is irreducible
189                     if (f.is_irreducible()==False):
190                         return "The polynomial "+print_pol(f.list())+" is
        not irreducible in F!"
191
192                     #Compute order and/or factorization
193                     g=polynomial(f_vec, y) # embed the polynomial f in PRE
194
195                     s_csv = ""
196
197                     # Compute the order of the current polynomial while no
        polynomial of odd order has been found (tail polynomials)
```

```
198                     if f_order%2==0:
199                         root = g.roots()
200                         root = root[0]
201                         root = root[0]
202                         f_order = root.multiplicative_order()
203                         s_csv = sep + str(f_order)+" = "+str(q^n-1)+" / "+
    str((q^n-1)/f_order)+ " = "+str(f_order.factor())
204
205                         if f_order%2 ==1:
206                             order_mod_2 = order_mod(2,f_order)
207                             first_orbit_pol = f
208                             print("First orbit pol = "+str(first_orbit_pol)
    )
209                             csv_file.write("\n")
210
211                     if(f_order %2==1):
212                         num_odd_order +=1
213
214                     # Update weight table
215                     f_wt = weight(f_vec)
216                     weights[f_wt-1]+=1
217
218                     csv_file.write("\n"+str(num_odd_order)+sep+print_pol(f.
    list())+sep + str(f_wt) + s_csv)
219
220                     #Construct new polynomial:
221                     for i in range(len(f_vec)):
222                         if(i%2==1):
223                             f_vec[i]=-f_vec[i]
224
225                     # new_f_vec is the coefficient vector of the polynomial
     C(X^2):
226                     new_f_vec = (f* (-1)^n * polynomial(f_vec, x)).list()
227                     f_vec = []
228
229                     # Extraction of C from C(X^2):
230                     for i in range(len(new_f_vec)):
231                         if (i%2==0):
232                             f_vec.append(new_f_vec[i])
233
234                     f=polynomial(f_vec, x)
235
236                 # END of WHILE-Loop of Contruction 1 for f(X+a)
237
238                 #-----------------------------------------
239                 # Print and write the weight table:
240                 s_csv = "\n\n\n"+sep+"Weight table of constructed orbit
    polynomials of order "+str(f_order)+": "+"\n\n"+sep+(sep).join(str(i+1)
     for i in range(n+1))+"\n"
241
242                 for i in range(len(weights)):
243                     s_csv += sep+str(weights[i])
244
245                 csv_file.write(s_csv)
246                 csv_file.close()
247
248                 # If initial polynomial was irreducible, print weight table
     of constructed orbit polynomials
249                 # Else add weight table of f(X+a) to list and proceed with
    next a
250                 if satisfies_conditions:
251                     print(s_csv)
252                 else:
```

```
253                     weight_tables.append([str(el), f_order, str(order_mod_
    2), weights])
254
255     # END of FOR-Loop through all a in Fq
256
257     if satisfies_conditions==False:
258         #-----------------------------------
259         #Write overview file for all a in Fq
260
261         weights_file=open(filename_prefix+"_ov"+id+".csv", "w")
262         s=file_init+sep+"Initial polynomial f = "+str(print_pol(start_f_vec
    ))+"\n\n"+sep+str(q)+"^"+str(n)+"-1 = "+str(q^n-1)+"\n\n"+sep+"Weight
    tables of constructed orbit polynomials from f(X+a) for all a in F:\n\
    na in F"+sep+"Orbit order"+sep+"#Orbit polynomials"+sep+"Weight"+ sep +
     (sep).join(str(i+1) for i in range(n+1))+"\n"
263
264         print("_____\n\nWeight tables of
    constructed orbit polynomials from f(X+a) for all a in F:\n\na\tOrbit
    order\t#Orbit polynomials\tWeight:\t"+"\t".join(str(i+1) for i in range
    (n+1))+"\n")
265
266         for tuple in weight_tables:
267             print(tuple[0]+"\t"+str(tuple[1])+"\t\t"+tuple[2]+"\t\t\t\t"+"\
    t".join(str(m) for m in tuple[3]))
268             s+="\n"+tuple[0]+sep+str(tuple[1])+" = "+str(q^n-1)+" / "+str((
    q^n-1)/tuple[1])+sep+tuple[2]+sep+sep+(sep).join(str(m) for m in tuple
    [3])
269
270         weights_file.write(s)
271         weights_file.close()
272
273     print("\nMore detailed results can be found in the file(s) <<"+filename
    +"_....csv>> in the folder "+filepath+".")
274
275     return
276
277 if __name__ == "__main__":
278     main()
```