FACTORIZATION OF POLYNOMIALS

A.K. Lenstra

Doctoraalscriptie o.l.v. P. van Emde Boas

November 1980
1. INTRODUCTION

A method for factoring univariate polynomials over algebraic number fields is presented. An outline of the factorization algorithm is as follows. First we obtain the irreducible factorization of the given polynomial over one or more finite fields, using some well-known algorithms for the factorization over finite fields (chapters 2, 3 and 4). Next we apply the Hensel-lemma to lift this factorization (chapter 5). Finally we use this lifted factorization to determine the factorization over the algebraic number field. Details of this last step are given in the complete factorization-schemes for \( \mathbb{Z}[X] \) (chapter 6) and \( \mathbb{Q}(\alpha)[X] \) (chapter 7). In chapter 6 we also briefly discuss the problem of gcd-computations in \( \mathbb{Z}[X] \).

The algorithms described have been implemented in Algol 68 on a Cyber CDC 173 computer.
**Lemma 2.4.** The polynomial $X^{q^k} - X$ factors over $\mathbb{F}_q$ into the product of all monic irreducible polynomials of degrees dividing $k$.

Lemma 2.4 enables us to determine the partial factorization (remember that $f$ is square-free):

$$g_i = \gcd \left( X^{q^i} - X, f/\Pi_{j=1}^{i-1} g_j \right), \ i = 1, \ldots, n.$$

In practice we can apply this method by computing $X^{q^i} \mod f$ and

$$X^{q^i} \mod f = (X^{q^{i-1}} \mod f)^q \mod f, \ i = 2, \ldots, \lfloor n/2 \rfloor.$$  

In the case that $f$ is irreducible it will take $O([n/2] \log q)$ polynomial multiplications modulo $f$ to compute $X^{q^i} - X \mod f, \ i = 1, \ldots, \lfloor n/2 \rfloor$, in this way. We can improve on the number of polynomial multiplications modulo $f$ as follows.

Let the $n \times n$ matrix $Q$ have as its $i$-th row the coefficients of $X^{q^i} \mod f$ for $i = 0, \ldots, n-1$. In the sequel we identify polynomials of degree $< n$ with the rowvectors formed by their coefficients.

**Theorem 2.1.** For any $v = \sum_{i=0}^{n-1} v_i X^i \in \mathbb{F}_q[x]$ we have $v \cdot Q = v^q \mod f$.

**Proof.** Let $Q = (q_{ij})_{i,j=0}^{n-1}$.

$$v(X)^q \mod f = v(X^q) \mod f = (\sum_{i=0}^{n-1} v_i X^{q^i}) \mod f$$

$$= \sum_{i=0}^{n-1} v_i \sum_{k=0}^{n-1} q_{ik} X^{q^k} \mod f$$

$$= \sum_{k=0}^{n-1} (\sum_{i=0}^{n-1} v_i q_{ik}) X^k \mod f$$

$$= v \cdot Q \ . \ \square$$

From Theorem 2.1 we conclude $(X^{q^{i-1}} \mod f) \cdot Q = X^{q^i} \mod f$.

It takes $O(\log q + n-2)$ polynomial multiplications modulo $f$ to compute the matrix $Q$. Using $Q$ we then compute the $\lfloor n/2 \rfloor$ polynomials $X^{q^i} - X \mod f, \ i = 1, \ldots, \lfloor n/2 \rfloor$, in $O(n^3)$ finite field computations. For large $q$ this method is therefore preferable.
EXAMPLE 2.1. Let \( f = x^6 + 3x^5 + 2x^4 + 3x^3 - 3x^2 + 3x + 2 \in \mathbf{F}_7[x] \). From \( f' \neq 0 \) and \( \gcd(f, f') = 1 \), it follows that \( f \) is square-free. We compute \( Q \).

\[
\begin{align*}
X^7 \mod f &= 3x^4 - 2x^3 + 2x^2 - 1, \\
X^{14} \mod f &= x^7 \cdot X^7 \mod f = -x^5 + 3x^4 - 2x^3 - x^2 - x + 2, \\
X^{21} \mod f &= x^{14} \cdot X^7 \mod f = -x^5 + 3x^4 + 2x^2 + 3x + 2, \\
X^{28} \mod f &= x^{21} \cdot X^7 \mod f = 2x^5 - 2x^4 + 2x^3 - x^2 + x + 2 \text{ and} \\
X^{35} \mod f &= x^{28} \cdot X^7 \mod f = x^4 - 3x^3 - x^2 + 2x - 2.
\end{align*}
\]

We obtain

\[
Q = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 2 & -2 & 3 & 0 \\
2 & -1 & -1 & -2 & 3 & -1 \\
2 & 3 & 2 & 3 & 0 & -1 \\
2 & 1 & -1 & 2 & -2 & 2 \\
-2 & 2 & -1 & -3 & 1 & 0
\end{pmatrix}
\]

\[
g_1 = \gcd(x^7-x, f) = \gcd(3x^4 - 2x^3 + 2x^2 - x - 1, f) = x + 1,
\]
with cofactor \( f/g_1 = x^5 + 2x^4 + 3x^2 + x + 2 \).

In order to compute \( g_2 \) we need \( x^{49} \mod f \), and this we compute by one vector \times matrix multiplication:

\[
X^{49} \mod f = (X^7 \mod f) \cdot Q = (-1, 0, 2, -2, 3, 0) \cdot Q = (-2, 2, -2, 3, 0, -1).
\]

Now, \( g_2 = \gcd(-x^5 + 3x^3 - 2x^2 + x - 2, x^5 + 2x^4 + 3x^2 + x + 2) = x^2 + 2x - 2 \),
with cofactor \( f/(g_1 \cdot g_2) = x^3 + 2x - 1 \).

Since the irreducible factors of \( f \) have different degrees, we now have obtained the complete factorization:

\[
f = (x+1) \cdot (x^2 + 2x - 2) \cdot (x^3 + 2x - 1).
\]

An improved version of this partial factorization algorithm can be found in Berlekamp [2].
3. ROOT-FINDING IN $\mathbb{F}_q^*[X]$

Let $f$ be a monic, square-free polynomial of degree $n > 1$ in $\mathbb{F}_q[X]$, which splits over $\mathbb{F}_q$ into $n$ linear factors. In this chapter we present three methods to determine the $n$ distinct roots of $f$, i.e. $s_1, \ldots, s_n \in \mathbb{F}_q$ such that $f = \prod_{i=1}^{n} (X-s_i)$.

The first, rather trivial, method is to test for every element $s \in \mathbb{F}_q$ whether or not $s$ is a root of $f$. For small $q$ this method is certainly very efficient, but it is clear that we need another approach for large $q$.

We denote by $\text{Tr}(X)$ the trace-polynomial $\sum_{i=0}^{\ell-1} X^p^i$, where $q = p^\ell$. Observe that it gives rise to a linear map $\mathbb{F}_q \rightarrow \mathbb{F}_p$.

**Lemma 3.1.** For each $\gamma \in \mathbb{F}_q$ we have $X^{q^d} - X = \gamma^{-1} \prod_{s \in \mathbb{Z}/p\mathbb{Z}} (\text{Tr}(\gamma X) - s)$.  

**Lemma 3.2.** Let $f$ and $v_i$, $i = 1, \ldots, k$, be monic polynomials in $\mathbb{F}_q[X]$ such that $\gcd(v_i, v_j) = 1$, $i \neq j$, and $f \mid \prod_{i=1}^{k} v_i$. Then $f = \prod_{i=1}^{k} \gcd(f, v_i)$.

Let $\beta \in \mathbb{F}_q$ so that $\{\beta^0, \beta^1, \ldots, \beta^{\ell-1}\}$ forms a basis for $\mathbb{F}_q$ over $\mathbb{Z}/p\mathbb{Z}$.

Since $f$ splits over $\mathbb{F}_q$ into $n$ different linear factors, we have $X^{q^d} - X \equiv 0 \pmod f$ (Lemma 2.4).

Applying Lemmas 3.1 and 3.2 we find that

$$f = \prod_{s \in \mathbb{Z}/p\mathbb{Z}} \gcd(f, \text{Tr}(\beta^j X) - s), \ 0 \leq j < \ell.$$  

If there exists $j'$, $0 \leq j' < \ell$, such that $\text{Tr}(\beta^{j'} X)$ is not congruent to a scalar modulo $f$, then we can find a non-trivial factorization of $f$ just by computing $\gcd(f, \text{Tr}(\beta^{j'} X) - s)$ for every $s \in \mathbb{Z}/p\mathbb{Z}$. The roots of $f$ can be found by applying this method recursively to the non-linear, non-trivial factors of $f$. Remark that we can not use the same $j'$ in the recursion.
We now prove that such a \( j' \) exists. Suppose on the contrary that for all \( j, 0 \leq j < \ell \), \( \text{Tr}(\beta^j x) \) is congruent to a scalar modulo \( f \),

\[
\exists \ t_j \in \mathbb{F}_q \text{ such that } \text{Tr}(\beta^j x) \equiv t_j \mod f, \ 0 \leq j < \ell.
\]

Let \( s_1, \ldots, s_n \) be the roots of \( f \) in \( \mathbb{F}_q \), then

\[
\text{Tr}(\beta^j s_1) = \ldots = \text{Tr}(\beta^j s_n) = t_j, \ 0 \leq j < \ell,
\]

and using Lemma 2.1 we have

\[
\text{Tr}(\beta^{j}(s_i-s_k)) = 0, \ 0 \leq j < \ell, \ 1 \leq i < k \leq n.
\]

We find that for any \( p_0', \ldots, p_{k-1} \in \mathbb{Z}/p\mathbb{Z} \)

\[
0 = \sum_{j=0}^{\ell-1} p_j \text{Tr}(\beta^{j}(s_i-s_k)) = \text{Tr}(\sum_{j=0}^{\ell-1} p_j \beta^{j}(s_i-s_k)) \quad \text{(Lemma 2.1)}.
\]

We conclude that \( \text{Tr}(s) = 0, \forall s \in \mathbb{F}_q \), because \( \{\beta^0, \ldots, \beta^{\ell-1}\} \) forms a basis for \( \mathbb{F}_q \) over \( \mathbb{Z}/p\mathbb{Z} \) and \( s_1 \neq s_k \). From Lemma 3.1 it is clear that the trace-polynomial has only \( p^{\ell-1} \) roots, and therefore we have a contradiction.

**Example 3.1.** Let \( f = x^3 - 3x^2 + 3x + 5 \in \mathbb{F}_{121}[x] \). The polynomial \( f \) is square-free and splits over \( \mathbb{F}_{121} \) into three linear factors. In order to represent the elements of \( \mathbb{F}_{121} \) we choose the irreducible polynomial \( G(T) = T^2 + 5 \in \mathbb{F}_{11}[T] \), and \( \beta \) a zero of \( G \). We find

\[
\text{Tr}(x) = x + x^{11} \equiv 4x^2 + 4x + 5 \mod f.
\]

Since \( \text{Tr}(x) \) is not congruent to a scalar we will find a non-trivial factorization of \( f \) by computing

\[
\gcd(f, 4x^2 + 4x + 5 - s), \ s \in \mathbb{Z}/11\mathbb{Z}.
\]
It turns out that only $s = -3$ and $s = -1$ give a non-trivial factor:

$$\gcd(f, 4x^2 + 4x - 3) = x - 4 \text{ and } \gcd(f, 4x^2 + 4x - 5) = x^2 + x - 4 = g.$$ 

We now apply the method recursively to $g$. Clearly $\text{Tr}(X)$ is congruent to a scalar modulo $g$, and therefore $\text{Tr}(\beta X)$ cannot be a scalar modulo $g$, because $0 \leq j < 2$. Indeed

$$\text{Tr}(\beta X) = \beta x + \beta^{11} x^{11} \equiv 2\beta x + \beta \mod g,$$

and we find

$$\gcd(g, 2\beta x + \beta + 5) = x + 5\beta - 5 \text{ and } \gcd(g, 2\beta x + \beta - 5) = x - 5\beta - 5.$$ 

Conclusion: $f = (x - 4) \cdot (x + 5\beta - 5) \cdot (x - 5\beta - 5)$ over $\mathbb{F}_{121}$.

This method is not very efficient for finite fields with a large characteristic, because it takes $O(p)$ gcd-computations for each trace-polynomial, and therefore, in the worst case, $O(p^2)$ gcd-computations. For this reason we describe a third root-finding algorithm. Regardless of the size of $q$, this method will find the roots of a given polynomial quite fast, but we cannot give an upperbound on the number of computation steps, because it is a probabilistic algorithm.

We will restrict ourselves to the case that the characteristic $p$ is odd. Then we know that for every $s \in \mathbb{F}_q$ either $s^r = 1$ or $s^r = -1$, where $r = (q-1)/2$. We say that non-zero elements $s_1$ and $s_2$ in $\mathbb{F}_q$ are of different types if $s_1^r \neq s_2^r$. 
LEMMA 3.3. Let $s_1$ and $s_2$ be two non-zero, unequal elements in $F_q$, then

$$\# \{s \in F_q \mid 0 \neq (s_1 + s)^r \neq (s_2 + s)^r \neq 0 \} = r.$$ 

**Proof.** If $s_1 + s \neq 0$ then either $(s_1 + s)^r = 1$ or $(s_1 + s)^r = -1$, $i = 1, 2$. Therefore

$$0 \neq (s_1 + s)^r \neq (s_2 + s)^r \neq 0 \iff \left( \frac{s_1 + s}{s_2 + s} \right)^r = -1.$$ 

The equation $x^r + 1 = 0$ has exactly $r$ distinct roots in $F_q$, and every root $t$ gives a unique $s \in F_q$ because $t \neq 1$: $s = \frac{s_1 - ts_2}{t - 1}$.

The idea of the probabilistic root-finding algorithm is as follows. Let $s_1$ and $s_2$ be two unknown, unequal, non-zero roots of $f$. We can try and separate the factors $(X - s_1)$ and $(X - s_2)$ of $f$ by computing $\gcd(f(X), X^r - 1)$. This succeeds if and only if $s_1$ and $s_2$ are of different types. In the other case we select at random an element $s \in F_q$ and replace $f(X)$ by $f(X - s)$. According to Lemma 3.3 the roots $s_1 + s$ and $s_2 + s$ of $f(X - s)$ have a probability of at least $\frac{1}{2}$ to be of different types. If the shift $s$ is "lucky", that means $s_1 + s$ and $s_2 + s$ are of different types, we can separate $s_1$ and $s_2$ by computing $\gcd(f(X - s), X^r - 1)$, otherwise we select another $s \in F_q$ and try again.

**Example 3.2.** Take $f$ and $\beta$ as in Example 3.1.

$\gcd(f, X^{60} - 1) = X - 4$, go on with $g = f(X - 4) = X^2 + X - 4$.

Select at random an element $s \in F_{121}$, for example $s = \beta + 1$, and replace $g(X)$ by $g(X - s)$. Unfortunately $\gcd(g(X - s), X^{60} - 1) = 1$, so this $s$ is unlucky. We try again, and we select $s = -2\beta - 2$ in $F_{121}$ at random. We get $\gcd(g(X + 2\beta + 2), X^{60} - 1) = X - 4\beta - 3$, so this is a lucky choice for $s$. Now $X - 4\beta - 3$ is a factor of $g(X + 2\beta + 2)$ and therefore $X - 4\beta - 3 - 2\beta - 2 = X + 5\beta - 5$ is a factor of $g(X)$ with cofactor $X - 5\beta - 5$. 


4. FACTORIZATION OF POLYNOMIALS IN $\mathbb{F}_q[X]$

Let $f$ be a monic, square-free polynomial of degree $n$ in $\mathbb{F}_q[X]$. In this chapter we describe two methods to factorize $f$ completely over a finite field. The first method is the well-known Berlekamp's factorization algorithm for a small finite field $\mathbb{F}_q$. The second method is an adaptation of Berlekamp's factorization algorithm for large finite fields using the root-finding algorithms from chapter 3.

The matrix $Q$, introduced in chapter 2, appears to be very useful here, and in particular the kernel of the matrix $Q-I$ will be of vital importance, where $I$ is the $n \times n$ identity matrix.

**THEOREM 4.1.** Let $v \in \mathbb{F}_q[X], \deg(v) < n$, then

$$v^q \equiv v \mod f \iff v \cdot Q = v.$$

**PROOF.** Use Theorem 2.1. []

Suppose we have a polynomial $v \in \mathbb{F}_q[X], 0 < \deg(v) < n$, such that $v \cdot Q = v$,
i.e. $v$ is an element of the kernel of $Q-I$. Theorem 4.1 gives

$$f \mid v^q - v$$

and with Lemma 2.4

$$f \mid \prod_{s \in \mathbb{F}_q} (v-s).$$

We now apply Lemma 3.2 and we find

$$f = \prod_{s \in \mathbb{F}_q} \gcd(f, v-s).$$

This equation gives a non-trivial factorization of $f$, because $0 < \deg(v) < n$, and we see that such a $v \in \mathbb{F}_q[X]$ can not exist if $f$ is irreducible over $\mathbb{F}_q$. 
But, if \( f \) is irreducible over \( \mathbb{F}_q \), can we find a non-trivial \( v \) in the kernel of \( Q-I \) to factorize \( f \) in this way, and if so, how do we know whether or not we have found the complete factorization of \( f \) over \( \mathbb{F}_q \)? The next two theorems answer these questions.

**Theorem 4.2.** The rank of the kernel of \( Q-I \) equals the number of irreducible factors of \( f \) over \( \mathbb{F}_q \).

**Proof.** Let \( f = \prod_{i=1}^{r} g_i \) be the complete factorization of \( f \) over \( \mathbb{F}_q \), with \( g_i \) pairwise relatively prime, \( i = 1, \ldots, r \), because \( f \) is square-free.

Using the Chinese Remainder Theorem, we find that

\[
\mathbb{F}_q[x]/(f) \cong \prod_{i=1}^{r} (\mathbb{F}_q[x]/(g_i)),
\]

where the isomorphism is defined by

\[v \mapsto (v \mod g_i)_{i=1}^{r}, \quad v \in \mathbb{F}_q[x]/(f).
\]

Let

\[t \mapsto (t_i)_{i=1}^{r}, \quad t \in \mathbb{F}_q[x]/(f),
\]

then

\[t_i^q = t \circ t_i = t_i, \quad i = 1, \ldots, r.
\]

Since \( \mathbb{F}_q[x]/(g_i) \) is a field, we have

\[t_i^q = t_i \circ t_i = t_i, \quad i = 1, \ldots, r,
\]

for \( i = 1, \ldots, r \), and therefore

\[t \in \text{Ker}(Q-I) \circ (t_i)_{i=1}^{r} \in (\mathbb{F}_q)^r.
\]

We conclude that the rank of the kernel of \( Q-I \) equals \( r \), the number of irreducible factors of \( f \) over \( \mathbb{F}_q \). \( \square \)
THEOREM 4.3. Let \( f \) and \( g_j, i = 1, \ldots, r, \) be as in Theorem 4.2. Let \( \{v_1, \ldots, v_r\} \) form a basis of the kernel of \( Q-I. \) For every \( j \neq j', 1 \leq j < j' \leq r, \) there exist \( k, 1 \leq k \leq r, \) and \( s \in \mathbb{F}_q \) such that \( g_j | \gcd(f, v_k-s) \) and \( g_{j'} | \gcd(f, v_k-s). \)

PROOF. It is clear from the proof of Theorem 4.2 that we can find an element \( t \) in the kernel of \( Q-I \) such that

\[ t_j \neq t_{j'}, \quad \text{where} \quad t = (t_i)_{i=1}^r. \]

Therefore there exists a \( k, 1 \leq k \leq r, \) such that

\[ v_k \mod g_j \neq v_k \mod g_{j'}, \]

Let \( v_k \mod g_j = s \in \mathbb{F}_q, \) then

\[ g_j | v_k - s \quad \text{and} \quad g_{j'} | v_k - s. \]

From these theorems it will be clear how we can factorize \( f \) completely over \( \mathbb{F}_q. \)

ALGORITHM 4.1. (Berlekamp's factorization algorithm)

For \( f \in \mathbb{F}_q[X], \) this algorithm computes the irreducible factors of \( f \) over \( \mathbb{F}_q. \)

(1) Determine \( \{v_1, \ldots, v_r\}, \) a basis of the kernel of \( Q-I, \) by diagonalizing \( Q-I. \) Take \( v_1 = 1 \) and \( 0 < \deg(v_1) < n, \) \( i = 2, \ldots, r. \)

(2) If \( r = 1 \) then \( f \) is irreducible.

(3) Compute \( \gcd(f, v_2-s) \) for all \( s \in \mathbb{F}_q. \) Since \( f = \prod_{s \in \mathbb{F}_q} \gcd(f, v_2-s), \) and because \( 0 < \deg(v_2) < n, \) this gives a non-trivial factorization of \( f. \)

If we find \( r \) factors of \( f \) in this way we are done. In the other case we compute \( \gcd(g, v_k-s) \) for all \( g \in \mathbb{F}_q, \) for all factors \( g \) of \( f \) discovered so far and \( k = 3, \ldots, r. \) Theorem 4.3 guarantees that we find all factors of \( f \) in this way.
EXAMPLE 4.1. Let \( f = x^6 + 3x^5 + 2x^4 + 3x^3 - 3x^2 + 3x + 2 \in \mathbb{F}_7[x] \), the same polynomial as in Example 2.1. We apply Berlekamp's factorizational algorithm directly to \( f \).

By diagonalizing \( Q-I \) we find \( \{1, x^4 + 2x^2 - x, x^5 - x^2 + 3x\} \), a basis of the kernel of \( Q-I \), so \( r = 3 \). Compute \( \gcd(f, v_2 - s) \) for all \( s \in \mathbb{F}_7 \), with \( v_2 = x^4 + 2x^2 - x \):

\[
\begin{align*}
gcd(f, v_2) &= x^3 + 2x - 1, \\
gcd(f, v_2 - 1) &= 1, \\
gcd(f, v_2 - 2) &= x^2 + 2x - 2, \\
gcd(f, v_2 - 3) &= 1, \\
gcd(f, v_2 + 3) &= x + 1, \\
gcd(f, v_2 + 2) &= 1 \text{ and} \\
gcd(f, v_2 + 1) &= 1.
\end{align*}
\]

We have found 3 factors of \( f \), so we are done, and the complete factorization of \( f \) is: \( f = (x+1) \cdot (x^2 + 2x - 2) \cdot (x^3 + 2x - 1) \).

The number of computation steps required for Berlekamp's factorizational algorithm is \( O(n^3) \) computations in \( \mathbb{F}_q \) for step (1), and \( O(q) \) gcd-computations in \( \mathbb{F}_q[X] \) for every \( v_i \) in the basis of the kernel of \( Q-I \) in step (3). If \( q \) is small this is an efficient algorithm, but if \( q \) is large it becomes rather impractical.

In step (3) we compute \( \gcd(f, v_2 - s) \) for all \( s \in \mathbb{F}_q \), and at most \( r \) of these gcd's will not be trivial, so at least \( q-r \) of the gcd-computations will be completely useless.

We deal with this problem using the root-finding algorithms from chapter 3.

For \( v \in \mathbb{F}_q[X] \) in the kernel of \( Q-I \) we define \( S_v \in \mathbb{F}_q \) as follows:

\[
S_v = \{ s \in \mathbb{F}_q \mid \gcd(f, v-s) \neq 1 \}.
\]

It is clear that an efficient algorithm to determine \( S_v \) for a given \( v \) will give us an adaptation of Berlekamp's factorizational algorithm to large finite fields.
We know that

\[ f = \prod_{s \in S_v} \gcd(f, v - s), \]

and therefore

\[ f \mid \prod_{s \in S_v} (v - s). \]

So

\[ f \mid H(v) \]

where

\[ H = \prod_{s \in S_v} (X - s). \]

Conversely, let \( f \mid G(v) \) with \( G \in \mathbb{F}_q[X] \). Substituting a common zero of \( f \) and \( v - s \) for \( X \), we then find that \( G(s) = 0 \), for all \( s \in S_v \), so \( H \) divides \( G \).

This proves that \( H \) is the polynomial of minimal degree in \( \mathbb{F}_q[X] \) for which \( f \mid H(v) \). This enables us to compute \( H \) by looking for the first linear relation between the powers \( v^0, v^1, v^2, \ldots \) considered modulo \( f \). Since \( H \) has degree \( \#S_v \) and \( \#S_v \leq r \) we do not have to look beyond the \( r \)-th power.

Once we have \( H \) we can determine \( S_v \) by the methods of chapter 3.

**EXAMPLE 4.2.** We determine \( S_v \) where \( v_2 = x^4 + 2x^2 - x \) from Example 4.1.

Since \( r = 3 \) the powers \( v_2^0, v_2^1, v_2^2 \) mod \( f \) and \( v_2^3 \) mod \( f \) must be linearly dependent.

\[ v_2^0 = 1, \]
\[ v_2^1 = x^4 + 2x^2 - x, \]
\[ v_2^2 \mod f = 3x^5 + x^4 - x^2 + x - 1 \] and
\[ v_2^3 \mod f = -3x^5 - 2x^4 - x^2 + 1. \]

Indeed \( v_2^3 + v_2^2 + v_2 \equiv 0 \mod f \), so \( H(v) = v^3 + v^2 + v \). The roots of \( H \) in \( \mathbb{F}_7 \) are 0, 2 and -3, so \( S_v = \{0, 2, -3\} \) in agreement with Example 4.1.
5. LIFTING A FACTORIZATION

First we have to introduce the notion of the so-called truncated Witt-ring $W_k(F_q)$. To represent the elements of $F_q$, $q = p^l$, we need a monic polynomial $G \in \mathbb{Z}[T]$ of degree $l$, that is irreducible modulo $p$. We then have

$$F_q = \{ \sum_{i=0}^{l-1} a_i \beta^i \mid a_i \in \mathbb{Z}/p \mathbb{Z} \},$$

where $\beta$ is a zero of $(G \mod p)$. We define $W_k(F_q)$ as follows:

$$W_k(F_q) = \{ \sum_{i=0}^{l-1} a_i \beta^i \mid a_i \in \mathbb{Z}/p^k \mathbb{Z} \},$$

where $\beta$ now denotes a zero of $(G \mod p^k)$. So $W_k(F_q)$ is a ring with $q^k$ elements. Remark that $W_1(F_q) = F_q$ and $W_k(\mathbb{Z}/p \mathbb{Z}) = \mathbb{Z}/p^k \mathbb{Z}$.

Let $f$ be a polynomial of degree $n$ in $(\mathbb{Z}[q])[X]$ and $g$ and $h$ polynomials in $F_q[X]$ such that $f = gh$ over $F_q$ and $\gcd(g, h) = 1$. Here $a$ is a zero of an irreducible, monic polynomial over $\mathbb{Z}$, the so-called minimum polynomial. In the factorization algorithms from chapters 6 and 7 we shall need a method to extend the factorization $f = gh$ over $F_q$ to a factorization over $W_k(F_q)$ for an arbitrary $k \geq 1$, i.e. $\tilde{g}, \tilde{h} \in (W_k(F_q))[X]$ such that $f = \tilde{g}\tilde{h}$ over $W_k(F_q)$, and such that $\tilde{g} \equiv g$ and $\tilde{h} \equiv h$ over $F_q$.

We present two methods to perform this so-called lifting of a factorization.

We denote by $\text{lc}(f)$ the leading coefficient of $f$.

**Algorithm 5.1.** (Extended Euclidean Algorithm in $F_q[X]$)

Given $g, h \in F_q[X]$, this algorithm computes unique (up to units) $a, b, d \in F_q[X]$ such that $ag + bh = d = \gcd(g, h)$ over $F_q$, $\deg(a) < \deg(h) - \deg(d)$ and $\deg(b) < \deg(g) - \deg(d)$. A description of this algorithm can be found in Knuth [12].

**Algorithm 5.2.**

Given $a, b, c, g, h \in (W_k(F_q))[X]$, such that $ag + bh = 1$ over $W_k(F_q)$.

The algorithm computes $\tilde{a}, \tilde{b} \in (W_k(F_q))[X]$ such that $\tilde{a}g + \tilde{b}h = c$ over $W_k(F_q)$ and $\deg(\tilde{a}) < \deg(h)$ as follows.
Compute $\tilde{a}, s \in (W_k(F_q))[x]$ such that $ac = sh + \tilde{a}$ (with $\deg(\tilde{a}) < \deg(h)$), and take $\tilde{b} = bc + sg$. It is trivial to verify that $\tilde{a}$ and $\tilde{b}$ satisfy $\tilde{a}g + \tilde{b}h = c$.

**Lemma 5.1.** (Hensel)

Let $f = (\mathbb{Z}[x])_q$ such that $lc(f) \neq 0$ in $F_q$, $g_1, h_1 \in F_q[x]$ such that $f = g_1 h_1$ over $F_q$ and $gcd(g_1, h_1) = 1$. For every $k \geq 1$ there exist $g_k, h_k \in (W_k(F_q))[x]$ such that $f = g_k h_k$ over $W_k(F_q)$, $g_k = g_1$ and $h_k = h_1$ over $F_q$.

**Proof.** Using Algorithm 5.1 we can find $a, b \in F_q[x]$ such that $ag_1 + bh_1 = 1$.

Now suppose there exist for some $j \geq 1$ polynomials $g_j, h_j \in (W_j(F_q))[x]$ such that $f = g_j h_j$ over $W_j(F_q)$, $g_j = g_1$ and $h_j = h_1$ over $F_q$.

Let $c_j \in F_q[x]$ such that $f - g_j h_j = p^j c_j$ over $W_{j+1}(F_q)$. Compute $a_j, b_j \in F_q[x]$ such that $a_j g_{j+1} + b_j h_{j+1} = c_j$ over $F_q$, using Algorithm 5.2.

Define $g_{j+1} = g_j + p^j b_j$ and $h_{j+1} = h_j + p^j a_j$ (notice: $lc(h_{j+1}) = lc(h_j)$).

Now clearly $g_{j+1}, h_{j+1} \in (W_{j+1}(F_q))[x]$, $g_{j+1} = g_1$ and $h_{j+1} = h_1$ over $F_q$ and

$$g_{j+1} h_{j+1} = (g_j + p^j b_j) \cdot (h_j + p^j a_j)$$

$$= g_j h_j + p^j (a_j g_1 + b_j h_1)$$

$$= g_j h_j + p^j c_j$$

$$f \text{ over } W_{j+1}(F_q). []$$

It is clear that the proof of this lemma provides us with a method to lift a factorization step by step, i.e. from $F_q = W_1(F_q)$ to $W_2(F_q), W_3(F_q), \ldots$ This is therefore called the linear lift algorithm. In practice it frequently occurs that there are more than two factors to be lifted. It is of course possible to apply the above method to each factor and cofactor, but if $f$ is monic it is possible to modify the algorithm so that an arbitrary number of relatively prime factors can be lifted simultaneously, without use of the cofactors [27].
**EXAMPLE 5.1.** Let $F(T) = T^2 + T + 1$ be a minimum polynomial, a zero of $F$, and 
$f = T^3 + (11a+1)T^2 - 25(a+1)T + 30a + 5 \in (\mathbb{Z}[a])[T]$. The minimum polynomial $F$ is irreducible modulo 2, so we can factorize $f$ in $F_4[X]$ without difficulties:

$$f = (T+1) \cdot (T^2 + aT + 1) \text{ over } F_4.$$ 

As in the Hensel-lemma we take $g_1 = T + 1$ and $h_1 = T^2 + aT + 1$. We lift the factorization of $f$ two steps, i.e. from $F_4 = W_1(F_4)$ to $W_3(F_4)$. Using Algorithm 5.1 we find $a = (a+1)X + a$ and $b = a+1$, $a, b \in F_4[X]$, satisfying $ag_1 + bh_1 = 1$.

$W_1(F_4) \rightarrow W_2(F_4)$: 
$f = g_1h_1 = 10aX^2 - 26(a+1)X + 30a + 5 \in (F_4)[X]$ 

$$= 2aX^2 + 2(a+1)X + 2a \text{ over } W_2(F_4)$$

$$= 2(aX^2 + (a+1)X + a).$$

Hence $c_1 = aX^2 + (a+1)X + a \in F_4[X]$. Apply Algorithm 5.2 to find $a_1, b_1 \in F_4[X]$ such that $a_1g_1 + b_1h_1 = c_1$ over $F_4$:

$a_1 = (X+a+1)h_1 + 0$, so $s = X+a+1$ and $a_1 = 0$ and therefore $b_1 = b_1 + sg_1 = a$.

We find $g_2 = g_1 + 2b_1 = X+2a+1 \in (W_2(F_4))[X]$ and $h_2 = h_1 + 2a_1 = X^2 + aX + 1 \in (W_2(F_4))[X]$.

$W_2(F_4) \rightarrow W_3(F_4)$: 
$f = g_2h_2 = 4(a+1) \in (W_3(F_4))[X]$, so $c_2 = a+1 \in F_4[X]$.

Using Algorithm 5.2 we find $a_2 = aX+1$ and $b_2 = a$ such that $a_2g_1 + b_2h_1 = c_2$ over $F_4$.

We find $g_3 = g_2 + 2b_2 = X+2a+1 \in (W_3(F_4))[X]$ and $h_3 = h_2 + 2a_2 = X^2 - 3aX - 3 \in (W_3(F_4))[X]$.

Now we have determined the factorization of $f$ over $W_3(F_4)$:

$$f = (X-2a+1) \cdot (X^2 - 3aX - 3) \text{ over } W_3(F_4).$$

We now present the second lifting method.
LEMMA 5.2. (Zassenhaus)

text goes here.

PROOF. Let $c \in (W_k(F_q))[x]$ such that $f - gh = p^k \cdot c$ over $W_{2k}(F_q)$. Compute $	ilde{h}, \tilde{g} \in (W_k(F_q))[x]$ such that $\tilde{h}g + \tilde{g}h = c$ over $W_k(F_q)$, using Algorithm 5.2. Define $\tilde{g} = g + k^{-1} \cdot \tilde{g}$ and $\tilde{h} = h + k^{-1} \cdot \tilde{h}$. It is trivial to verify that $\tilde{g}$ and $\tilde{h}$ satisfy the conditions above.

Let $r \in (W_k(F_q))[x]$ such that $\tilde{ag} + \tilde{bh} = 1 + p^k \cdot r$ over $W_{2k}(F_q)$. Compute $\tilde{a}, \tilde{b} \in (W_k(F_q))[x]$ such that $\tilde{ag} + \tilde{bh} = r$ over $W_k(F_q)$. Define $\tilde{a} = a - k^{-1} \cdot \tilde{a}$ and $\tilde{b} = b - k^{-1} \cdot \tilde{b}$, then $\tilde{a}, \tilde{b} \in (W_{2k}(F_q))[x]$ and

$$\tilde{ag} + \tilde{bh} = (a - k^{-1} \cdot \tilde{a}) \tilde{g} + (b - k^{-1} \cdot \tilde{b}) \tilde{h} = \tilde{ag} + \tilde{bh} - k^{-1} \cdot (\tilde{ag} + \tilde{bh}) = 1 \text{ over } W_{2k}(F_q).$$

This proof gives us a method to lift a factorization substantially faster than the linear lift algorithm, and it is called the quadratic lift algorithm because it extends a factorization in one step from $W_k(F_q)$ to $W_{2k}(F_q)$. Remark that it uses only $O(\log k)$ computation steps where the linear method uses $O(k)$ steps.

EXAMPLE 5.2. Take $F$ and $f$ as in Example 5.1, so $f = (x+1) \cdot (x^2 + \alpha x + 1)$ over $F_4$. Take $g_0 = x + 1$ and $h_0 = x^2 + \alpha x + 1$. From Example 5.1 we know $a_0 = (\alpha + 1) x + \alpha$ and $b_0 = \alpha + 1$. We lift the factorization of $f$ in two steps from $F_4 = W_1(F_4)$ to $W_4(F_4)$. 


\[ W_1(F_4) \rightarrow W_2(F_4) : \] See Example 5.1, \( c_0 = \alpha x^2 + (a+1)x + \alpha, h_0 = 0, \bar{g}_0 = \alpha, \]
\[ g_1 = g_0 + 2g_0 = x^2 + 2a + 1 \in (W_2(F_4))[x] \text{ and} \]
\[ h_1 = h_0 + 2h_0 = x^2 + ax + 1 \in (W_2(F_4))[x]. \]
\[ \alpha_0 g_1 + \beta_0 h_1 = 1 + 2((a+1)x^2 + (a+1)x - 1) \text{ over } W_2(F_4) \text{ so} \]
\[ r_0 = (a+1)x^2 + (a+1)x - 1 \in (W_1(F_4))[x]. \]
Applying Algorithm 5.2 we find \( \alpha_0 = a, \beta_0 = a+1, \) such that \( \alpha_0 g_0 + \beta_0 h_0 = r_0 \text{ over } W_1(F_4). \]

We find \( a_1 = a_0 - 2a_0 = (a+1)x - \alpha \in (W_2(F_4))[x] \text{ and} \]
\[ b_1 = b_0 - 2b_0 = -(a+1) \in (W_2(F_4))[x]. \]

\[ W_2(F_4) \rightarrow W_4(F_4) : \] \( f - g_1 h_1 = 4(2ax^2 + 2(a+1)x - a) \) over \( W_4(F_4), \) so \( c_1 = 2ax^2 + 2(a+1)x - a + 1 \in (W_2(F_4))[x]. \)
Using Algorithm 5.2 we find \( h_1 = ax + 1 \) and \( g_1 = \alpha \) such that \( h_1 g_1 = c_1 \text{ over } W_2(F_4). \)

We find \( g_2 = g_1 + 4g_1 = x + 6a + 1 \in (W_4(F_4))[x] \) and \( h_2 = h_1 + 4h_1 = x^2 + 5ax + 5 \in (W_4(F_4))[x]. \)

Now we have determined the factorization of \( f \) over \( W_4(F_4), \) and we remark that this factorization also holds over \( \mathbb{Z}[\alpha]: \)
\[ f = (x + 6\alpha + 1) \cdot (x^2 + 5aX + 5) \text{ over } \mathbb{Z}[\alpha]. \]
Let $f$ be a polynomial of degree $n$ in $\mathbb{Z}[X]$. Using the algorithms from the previous chapters, we shall give a method to construct the irreducible factors of $f$ over $\mathbb{Z}$. We denote by $\text{lc}(f)$ the leading coefficient of $f$, and by $\text{pp}(f)$ the primitive part of $f$.

Before we can give the factorization algorithm, we have to say something about $\text{gcd}$-computations in $\mathbb{Z}[X]$. Suppose we want to compute the $\text{gcd}$ $g$ of two polynomials $f_1$ and $f_2$ in $\mathbb{Z}[X]$. It is a well-known fact that $g$ is also a polynomial in $\mathbb{Z}[X]$. However, if we apply Euclid's algorithm to $f_1$ and $f_2$, we have to compute $f_{i+1} = f_{i-1} \mod f_i$, $i = 2, 3, \ldots$, until $f_i = 0$, and in general the coefficients of the polynomials $f_{i+1}$ will not be in $\mathbb{Z}$, but in $\mathbb{Q}$. This is in many cases unacceptable, and therefore we have to find a way to remain in $\mathbb{Z}[X]$. One way to deal with this problem is to use a pseudo-division in Euclid's algorithm, i.e.

$$f_{i+1} = (\text{lc}(f_i)^{\text{deg}(f_{i-1}) - \text{deg}(f_i) + 1} \cdot f_{i-1}) \mod f_i.$$ 

Now $f_{i+1}$ will certainly be in $\mathbb{Z}[X]$, but in practice this co-called EPRS-algorithm (Euclidean Polynomial Remainder Sequence) is completely useless because of the exponential coefficient growth. There are several improvements on the EPRS-algorithm which keep the coefficient growth linear, or almost always linear, like PrimitivePRS (PPRS), SubresultantPRS (SPRS) and ImprovedSPRS (ISPRS), see [3][4][5][7][8][9][10][12][16], but we will leave the PRS-algorithms and we take another approach.

**Theorem 6.1.** Let $f_1, f_2, g \in \mathbb{Z}[X]$, $g = \text{gcd}(f_1, f_2)$. For all primes $p$, such that $p|\text{lc}(f_1)\cdot\text{lc}(f_2)$ we have $\deg(g_p) \geq \deg(g)$, where $g_p = \text{gcd}(f_1, f_2)$ over $\mathbb{Z}/p\mathbb{Z}$, and there are only finitely many $p$ such that $\deg(g_p) > \deg(g)$. 
PROOF. The first assertion is trivial, the second assertion follows when we look at the leading coefficients of the polynomials \( f_{i+1} \) in the EPRS-algorithm. A larger degree for the gcd \( g_p \) can only occur if \( p \) divides one of these leading coefficients, and this can only happen for finitely many primes. \( \square \)

Combining this theorem with the well-known Chinese Remainder Theorem, we get the so-called modular gcd-algorithm (MODGCD).

**ALGORITHM 6.1. (MODGCD)**

Let \( f_1, f_2 \in \mathbb{Z}[x] \), \( f_1 \) and \( f_2 \) primitive, this algorithm computes \( \gcd(f_1, f_2) \) in \( \mathbb{Z}[x] \).

1. Choose a large prime \( p \), such that \( p \nmid \text{lcm}(f_1) \cdot \text{lcm}(f_2) \),
   \[
g := \gcd(f_1 \mod p, f_2 \mod p) \in (\mathbb{Z}/p\mathbb{Z})[x], \quad \text{product} := p.
   \]

2. Test whether or not \( g \) will lead to the gcd of \( f_1 \) and \( f_2 \) over \( \mathbb{Z} \):
   take \( h = \text{lcm}(f_1) \cdot g \mod \text{product} \),
   if \( h \mid \text{lcm}(f_1) \cdot f_1 \) and \( pp(h) \mid f_2 \) then \( pp(h) = \gcd(f_1, f_2) \)
   else go to step (3).

3. Choose another large prime \( p' \), \( p' \nmid \text{lcm}(f_1) \cdot \text{lcm}(f_2) \),
   \[
g_{p'} := \gcd(f_1 \mod p', f_2 \mod p') \in (\mathbb{Z}/p'\mathbb{Z})[x].
   \]

   There are three possibilities:
   - \( \deg(g_{p'}) < \deg(g) \): \( g := g_{p'} \), product := \( p' \), go to step (2),
   - \( \deg(g_{p'}) = \deg(g) \): combine \( g \) and \( g_{p'} \) with the Chinese Remainder Theorem to the unique polynomial \( h \), such that \( h \equiv g \mod \text{product} \), \( h \equiv g_{p'} \mod p' \), and all coefficients of \( h \) are in absolute value \( \leq \frac{\text{product} \cdot p'}{2} \).
     \[
g := h, \quad \text{product} := \text{product} \cdot p', \quad \text{go to step (2)},
   \]
   - \( \deg(g_{p'}) > \deg(g) \): repeat step (3).
Comment: The gcd of $f_1$ and $f_2$ modulo some prime, or modulo the product of several primes, is monic. However, it is possible that the gcd of $f_1$ and $f_2$ over $\mathbb{Z}$ is not monic, so we must be able to construct a possibly non-monic gcd over $\mathbb{Z}$. This is what we are doing in step (2).

In view of Theorem 6.1 it will be clear that the modular gcd-algorithm will terminate: after a finite (but unknown) number of steps $|\text{product}/2|$ will be larger than the absolute value of any coefficient of the gcd of $f_1$ and $f_2$ over $\mathbb{Z}$.

It is also possible to use the lift-algorithms from chapter 5 to find the gcd of two polynomials over $\mathbb{Z}$. This is the so-called EZGCD-algorithm (Extended Zassenhaus GCD, see [17][29]). We will not give a description of this algorithm.

Now we are ready for the factorization of $f$ over $\mathbb{Z}$. First we present the factorization-scheme, next we give a step by step explanation of the algorithm.

ALGORITHM 6.2. (Factorization of polynomials in $\mathbb{Z}[X]$)

Let $f \in \mathbb{Z}[X]$, this algorithm computes the irreducible factors of $f$ over $\mathbb{Z}$.

1. Make $f$ primitive and square-free.

2. Choose a prime $p$, and factorize $\tilde{f} = f \mod p$ over $\mathbb{Z}/p\mathbb{Z}$:

   \[ \tilde{f} = \text{lcm}(\tilde{g}_i) \prod_{i=1}^{r} g_i \mod p \quad \text{over} \quad \mathbb{Z}/p\mathbb{Z}, \quad g_i \in (\mathbb{Z}/p\mathbb{Z})[X], \quad i = 1, \ldots, r. \]

3. Choose $k \geq 1$, and lift the factorization from step (2) from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}/p^k\mathbb{Z}$:

   \[ f = \text{lcm}(f) \prod_{i=1}^{r} h_i \mod p^k \quad \text{over} \quad \mathbb{Z}/p^k\mathbb{Z}, \quad h_i \in (\mathbb{Z}/p^k\mathbb{Z})[X], \quad i = 1, \ldots, r. \]

4. Use the factorization over $\mathbb{Z}/p^k\mathbb{Z}$ to find the true factors of $f$ over $\mathbb{Z}$.

Comments:

1. By dividing out the gcd of the coefficients of $f$, we can make $f$ primitive.

   We can determine the square-free decomposition of $f$ using Lemma 2.2, which is of course still valid for polynomials in $\mathbb{Z}[X]$, and the
MODGCD-algorithm. The square-free decomposition algorithm is similar to the algorithm in chapter 2 and even simpler, because we don't have a characteristic $p$ here.

(2) In step (3) we require that the factors $g_i$ of $f_{\mod p}$ are relatively prime, in order to be able to apply one of the lift-algorithms from chapter 5. Therefore $f_{\mod p}$ must remain square-free, and this condition is met if $p \mid \text{disc}(f)$ ([21]). It is clear that a prime $p$ satisfying both conditions can be found.

It strongly depends on $\deg(f) = n$ and $p$ which method we will use to factor $f_{\mod p}$. In general we can give the following advice.

- $n$ and $p$ small: apply Berlekamp's factorization algorithm.
- $n$ large, $p$ small: first obtain a partial factorization of $f_{\mod p}$ (chapter 2), and apply Berlekamp's factorization algorithm if necessary to each of the partial factors. In practice it rarely occurs that we have to compute the complete basis of the kernel of $Q-I$ (which would take $O(n^3)$ finite field computations), we only determine a new polynomial in the kernel of $Q-I$ when we need it.
- $p$ large: independent of the value of $n$ it is advisable to apply the partial factorization method first, followed by the adaptation of Berlekamp's factorization algorithm for each factor. Remark that we will have to use the probabilistic root-finding algorithm.

In practice we can always find a small prime satisfying the conditions, and therefore the last possibility practically never occurs.

(3) The only problem we have here is to fix a value for $k$. We need a bound $b$, such that $b$ is larger than the absolute value of any coefficient of any factor $g$ of $f$ with $\deg(g) \leq \lfloor n/2 \rfloor$. Mignotte [15] gives us such a bound:

$$b = \left( \left\lfloor \frac{n}{2} \right\rfloor \right)^{\frac{2}{n}} \sqrt{\sum_{i=0}^{n} f_i^2},$$

where $f = \sum_{i=0}^{n} f_i x^i$. 

In view of step (4) we conclude that $k$ must be minimal so that
\[ p^k \geq 2|\text{lcm}(f)|b. \]
Depending on the value of $k$ we decide which of the lift-algorithms we
use (remark that we are allowed to apply them because $p$ is chosen such
that $f \mod p$ is square-free).

(4) We now have to combine the factors modulo $p^k$ to find the true factors
of $f$ over $\mathbb{Z}$. If $f$ is not monic we will test a combination of factors in
the same way as in Algorithm 6.1 step(2). I.e. if we want to know whether
or not $h_1, \ldots, h_s$ together form a true factor of $f$, we do this as
follows:
\[ g = (\text{lcm}(f) \cdot \prod_{j=1}^{s} h_j \mod p^k), \]
if $g | \text{lcm}(f) \cdot f$ then $p^\ell(g)$ is a true factor of $f$ over $\mathbb{Z}$.

If, for instance, $f$ is irreducible, and $f$ splits over $\mathbb{Z}/p^k\mathbb{Z}$ in $r$
irreducible factors, then we have to try every combination with degree
$s[n/2]$ of these $r$ factors before we know that $f$ is irreducible. This
will take a number of division-tests that is exponential in $r$, so this
step can become the bottleneck of the whole factorization algorithm.

It is often possible to reduce the number of division-tests:
- first try the constant coefficient.
- factorize $f$ modulo a number of primes. Use a prime with the smallest
  number of factors in steps (3) and (4), and combine the information
  about the degrees of the factors modulo the different primes to reduce
  the possibilities for the degrees of the factors over $\mathbb{Z}$. For example,
  if $f$ factors modulo $p_1$ in 3 factors of degree 2, and $f$ factors
  modulo $p_2$ in 2 factors of degree 3, then $f$ is irreducible over $\mathbb{Z}$.
- lift from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}/p^{k+1}\mathbb{Z}$ and don't try combinations which do not
  satisfy the given bound.
EXAMPLE 6.1. Let \( f = 9x^5 + 9x^4 + 15x^3 + 6x^2 + 7x + 4 \in \mathbb{Z}[x] \).

1. The polynomial \( f \) is primitive and square-free.

2. The prime \( p = 2 \) satisfies \( p \mid \text{lcm}(f) = 9 \), and \( f \text{ mod } 2 = x^5 + x^4 + x^3 + x \) is square-free in \( \mathbb{Z}/2\mathbb{Z} \).

Applying Berlekamp's factorization algorithm, we find
\[
\tilde{f} = x \cdot (x+1) \cdot (x^3 + x + 1) \text{ mod } 2.
\]

3. Choose \( k \) minimal, such that \( 2^k \geq 2 \cdot 9 \cdot \binom{2}{1} \cdot \sqrt{81 + 81 + 225 + 36 + 49 + 16} \approx 795 \), so \( k = 10 \).

The lift-algorithms give:
\[
f = 9 \cdot (x-187) \cdot (x+188) \cdot (x^3 - 341x - 341) \text{ mod } 1024.
\]

4. We try all combinations of factors with total degree \( \leq 2 \):

- \( x-187: 9 \cdot 187 = 1683 \equiv -365 \text{ mod } 1024 \text{ and } 365 \not| 9 \cdot 4 = 36 \rightarrow \text{no factor}, \)
- \( x+188: 9 \cdot 188 = 1692 \equiv -356 \text{ mod } 1024 \text{ and } 356 \not| 9 \cdot 4 = 36 \rightarrow \text{no factor}, \)
- \( (x-187) \cdot (x+188): 9 \cdot 187 \cdot 188 = 316404 \equiv -12 \text{ mod } 1024 \text{ and } 12 \not| 9 \cdot 4 = 36, \)

so this combination may give a true factor.

\[
P(9x^2 + 9x + 12) = 9x^2 + 9x + 12 \text{ mod } 1024 \text{ and } 9x^2 + 9x + 12 | 9 \cdot f, \text{ so}
\]

\[
\text{pp}(9x^2 + 9x + 12) = 3x^2 + 3x + 4 \text{ is a factor of } f \text{ over } \mathbb{Z}, \text{ with cofactor}
\]

\[
3x^3 + x + 1.
\]

The choice \( p = 7 \), for example, does not give erroneous factors:

\[
f \text{ mod } 7 = 2 \cdot (x^2 + x - 1) \cdot (x^3 + 5x + 5) \text{ mod } 7.
\]
7. FACTORIZATION OF POLYNOMIALS IN \((\mathbb{Q}(a))[X]\)

Let \(f\) be a monic, square-free polynomial of degree \(n\) in \((\mathbb{Q}(a))[X]\), where \(a\) is a zero of an irreducible, monic polynomial \(F\) of degree \(m\) in \(\mathbb{Z}[X]\). We present an algorithm to factorize \(f\) into irreducible factors over \(\mathbb{Q}(a)\).

In this chapter we restrict ourselves to square-free polynomials. If the given polynomial is not square-free, we can determine the square-free decomposition using Lemma 2.2, which is also valid for polynomials in \((\mathbb{Q}(a))[X]\). But then we need an algorithm for gcd-computations in \((\mathbb{Q}(a))[X]\). The Euclidean algorithm can of course be used in \((\mathbb{Q}(a))[X]\), but a serious drawback is that in general we will get rather large denominators. With some modifications the modular gcd-algorithm from chapter 6 can be adapted to \((\mathbb{Q}(a))[X]\). We will not give a description of this algorithm.

Factorization over \(\mathbb{Q}(a)\) introduces the problem of choosing a denominator. The first thing we remark is that we can find \(d \in \mathbb{Z}\) such that \(f \in \left(\frac{1}{d}\mathbb{Z}[a]\right)[X]\). However, even in the case that we can take \(d = 1\) (i.e. \(f \in (\mathbb{Z}[a])[X]\)) it is possible that the irreducible factors of \(f\) over \(\mathbb{Q}(a)\) have a non-trivial denominator, i.e. new denominators can occur when factoring over \(\mathbb{Q}(a)\). With the help of the following two lemmas we can determine \(D \in \mathbb{Z}\) such that \(f\) and all monic factors of \(f\) over \(\mathbb{Q}(a)\) lie in \(\left(\frac{1}{D}\mathbb{Z}[a]\right)[X]\).

**Lemma 7.1.** Let \(f \in \left(\frac{1}{d}\mathbb{R}\right)[X]\), \(f\) monic, and \(g\) and \(h\) monic polynomials in \((\mathbb{Q}(a))[X]\) such that \(f = gh\), then \(g, h \in \left(\frac{1}{d}\mathbb{R}\right)[X]\), where \(\mathbb{R}\) is the ring of integers of \(\mathbb{Q}(a)\).

**Lemma 7.2.** Let \(b = \max\{d \in \mathbb{N} \text{ such that } d^2|\text{discr}(F)\}\), then \(\text{defect}(a)|b\), and therefore \(R \subset \left(\frac{1}{D}\mathbb{Z}[a]\right),\) where \(\text{defect}(a) = \min\{d \in \mathbb{N} \text{ such that } R \subset \left(\frac{1}{d}\mathbb{Z}[a]\right)\}\).
If \( f \in \left( \frac{1}{\mathfrak{d}} \mathbb{Z}[a])][X] \) then certainly \( f \in \left( \frac{1}{\mathfrak{d}} \mathbb{Z}[X] \). Applying Lemma 7.1, we see that every monic factor of \( f \) over \( \mathbb{Q}(a) \) will be an element of \( \left( \frac{1}{\mathfrak{d}} \mathbb{Z}[X] \). But we can also find \( c \in \mathbb{N} \) such that \( R \in \frac{1}{c} \mathbb{Z}[a] \). We take \( c = \text{defect}(a) \) if \( \text{defect}(a) \) is known, and else we use the \( b \) from Lemma 7.2. Combining these results we obtain the \( D \) we wanted to have: \( D = d \cdot c \).

**EXAMPLE 7.1.** Let \( f = \mathfrak{d}^2 + \mathfrak{d}^1 - 1 \) (so \( d = 1 \)) and \( F(T) = T^2 - 5 \), so \( a = \sqrt{5} \).

We compute \( \text{discr}(F) \):

\[
\begin{vmatrix}
1 & 0 & -5 \\
-2 & 0 & 0 \\
0 & 2 & 0
\end{vmatrix}
= 20 \Rightarrow b = 2.
\]

Therefore \( D = 1 \cdot 2 = 2 \), and \( f = (X + \frac{a}{2} + \frac{1}{2}) \cdot (X - \frac{a}{2} + \frac{1}{2}) \).

**EXAMPLE 7.2.** Let \( F = \mathfrak{d}^2 + \mathfrak{d}^1 \frac{1}{2} \) (so \( d = 2 \)) and \( F(T) = T^2 + 7 \), so \( a = \sqrt{-7} \).

We compute \( \text{discr}(F) \):

\[
\begin{vmatrix}
1 & 0 & 7 \\
-2 & 0 & 0 \\
0 & 2 & 0
\end{vmatrix}
= -28 \Rightarrow b = 2.
\]

Therefore \( D = 2 \cdot 2 = 4 \), and \( f = (X + \frac{a}{4} + \frac{1}{4}) \cdot (X - \frac{a}{4} + \frac{1}{4}) \).

Now we give the factorization scheme for polynomial factorization over \( \mathbb{Q}(a) \).

**ALGORITHM 7.1.** (factorization of polynomials in \( (\mathbb{Q}(a))[X] \))

Let \( f \) be a monic, square-free polynomial in \( (\frac{1}{\mathfrak{d}} \mathbb{Z}[a)][X] \), with \( D \) chosen as above, this algorithm computes the monic, irreducible factors of \( f \) over \( \mathbb{Q}(a) \).

1. Choose a prime \( p \) and factorize \( F \) modulo \( p \):

\[
F = \prod_{i=1}^{t} F_i \mod p.
\]

We now have \( t \) finite fields \( F_i \), where the elements of \( F_i \) are represented as \( \sum_{j=0}^{\deg(F_i)-1} a_j b_j \), \( a_j \in \mathbb{Z}/\mathfrak{d} \mathbb{Z}, \) \( j = 0, \ldots, \deg(F_i)-1 \) and \( b \) a zero of \( F_i \), for \( i = 1, \ldots, t \).

2. Factorize \( f \) over \( F_i \) for \( i = 1, \ldots, t \).
(3) Choose \( k \geq 1 \) and lift the factorization of \( F \) from \( \mathbb{Z}/p\mathbb{Z} \) to \( \mathbb{Z}/p^k\mathbb{Z} \):

\[
F = \prod_{i=1}^{t} F_i \mod p^k.
\]

We now have \( t \) truncated Witt-rings \( \mathbb{W}_k(\mathbb{F}_{q_1}) \), where the elements of \( \mathbb{W}_k(\mathbb{F}_{q_1}) \)
are represented as \( \sum_{j=0}^{\deg(F_i)-1} a_j \beta^j \), \( a_j \in \mathbb{Z}/p^k\mathbb{Z} \), \( j = 0, \ldots, \deg(F_i)-1 \), and
\( \beta \) a zero of \( \tilde{F}_i \), for \( i = 1, \ldots, t \).

(4) Lift the factorization of \( f \) from \( \mathbb{F}_{q_1} \) to \( \mathbb{W}_k(\mathbb{F}_{q_1}) \) for \( i = 1, \ldots, t \).

(5) Combine these \( t \) factorizations of \( f \) to find the true factors of \( f \) over \( \mathbb{Q}(a) \).

Comments:

(1) Choose \( p \), preferably small, such that \( p \nmid D \) (because \( D^{-1} \mod p \) must
exist) and such that \( t \) is small. Furthermore, in view of steps (3) and (4), \( F \) and \( f \) have to be square-free in \( \mathbb{Z}/p\mathbb{Z} \) and \( \mathbb{F}_{q_1} \), \( i = 1, \ldots, t \), respectively. Such a prime can be found.

(3) In Algorithm 6.2 step (3) we have given a coefficient bound for factors
of polynomials in \( \mathbb{Z}[X] \). For polynomials in \( (\mathbb{Q}(a))[X] \) we can also give
such a bound:

Let \( f = \sum_{i=0}^{n} c_j X^j \), \( c_j \in \mathbb{Z} \), \( a \) a zero of \( F \), and
\( F(T) = \sum_{i=0}^{m} F_i(T) \).

Take \( k \) minimal such that

\[
p^k \geq 2Dm! \left( \frac{n}{2} \right)^2 \left( \frac{m-1}{n/4} \right)^m / \left( F(\ell^{1/2}) \right)^{m(m-1)/2} \cdot \left( \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} |c_{ij}| \left( \sum_{\ell=0}^{m-1} |F(\ell)\| \right)^{j} / D \right)^{n/4} \cdot \sqrt{\text{discr}(F)}^{-1}.
\]

(see [23] and [27])

In practice this bound will often be much too large.

(4) We apply one of the lift-algorithms to
\( \tilde{f} = (D^{-1} \mod p^k) \cdot (D \cdot f) \), so that \( \tilde{f} \in (\mathbb{Z}[a])[X] \).
Like step (4) of Algorithm 6.2 this step can become the bottleneck of the computation. Here the situation is far more complicated, because we have factorizations of $f$ over $t$ Witt-rings, and not just one as in chapter 6. We will give an outline of what has to be done to determine only the linear factors of $f$:

- If there is an $i$, $1 \leq i \leq t$, such that $f$ has no linear factors over $W_k(F_q)$, then we are done: $f$ has no linear factors over $\mathbb{Q}(a)$.

- Take for every $i$ a linear factor of $f$ over $W_k(F_q)$, $i = 1, \ldots, t$. Combine these $t$ factors into one linear factor $g$ of $f$ modulo $p^k$ and $F$, using the Chinese Remainder Theorem.

If $h = (D \cdot g \mod p^k)/D$ divides $f$ over $\mathbb{Q}(a)$, then $h$ is a factor of $f$ over $\mathbb{Q}(a)$. This has to be done for every possible combination of linear factors.

In practice it appears to be very important to find a small value for $t$.

**EXAMPLE 7.3.** Let $F(T) = T^3 - 75$, so $a^3 - 75 = 0$, and $f = X^3 - 45 \in (\mathbb{Q}(a))[X]$. We find that $\text{discr}(F) = -3^5 \cdot 5^4$, so the estimate for $\text{defect}(a)$ is $3^2 \cdot 5^2 = 225$.

We choose $D = 225$. The polynomial $f$ lies in $\mathbb{Z}[X]$ and is square-free in $\mathbb{Z}[X]$, so $f$ is square-free over $\mathbb{Q}(a)$.

1. We choose $p = 13$ because $F$ is irreducible modulo 13, $13 \mid 225$ and $f$ remains square-free over $\mathbb{F}_{13^3}$.

2. Using the adaptation of Berlekamp's factorization algorithm we get

$$f = (X + 2a^2) \cdot (X + 6a^2) \cdot (X + 5a^2) \text{ over } \mathbb{F}_{13^3}.$$

3. Since $t = 1$, we do not have to lift the factorization of $F$.

Let us compute the value for $k$:

$$13^k \geq 2 \cdot 225 \cdot 6 \cdot 1 \cdot (75)^3 \cdot 1/(225 \cdot \sqrt{3}) \approx 2922835,$$ e.g. $k = 6$.

4. Using the linear lift-algorithm we obtain:

$$f = (X - 1158051a^2) \cdot (X + 2123413a^2) \cdot (X - 965362a^2) \text{ over } W_6(\mathbb{F}_{13^3}).$$
(5) It is sufficient to reconstruct the denominator $D = 225$ for each of the linear factors over $\mathbb{F}_3$ and test whether or not it is a true factor.

\[ X - 1158051a^2 : \quad 1158051 \cdot 225 \mod 13^6 = -86211, \quad x + \frac{86211}{225}a^2 \not\mid f, \]
\[ X + 2123913a^2 : \quad 2123913 \cdot 225 \mod 13^6 = -86166, \quad x - \frac{86166}{225}a^2 \not\mid f, \]
\[ X - 965562a^2 : \quad 965562 \cdot 225 \mod 13^6 = 45, \quad x - \frac{a^2}{5} \not\mid f, \]

so we have found the factor $X - \frac{a^2}{5}$ with cofactor $x^2 + \frac{a^2}{5}x + 3a$.

**EXAMPLE 7.4.** Let $F(T) = T^3 - T - 1$, so $a^3 - a - 1 = 0$, and

\[
f = X^6 - X^5 - 3X^4 + 3X^3 + 3X^2 - X - 1 \in (\mathbb{Q}(a))[X]. \text{ We find that } \text{discr}(F) = -23, \text{ and because } 23 \text{ is square-free we choose } D = 1. \text{ The polynomial } f \text{ lies in } \mathbb{Z}[X] \text{ and is square-free in } \mathbb{Z}[X], \text{ so } f \text{ is square-free over } \mathbb{Q}(a).
\]

(1) For the example we take $p = 5$ and find

\[
F(T) = (T^2 + 2T - 2) \cdot (T - 2) \quad \text{over } \mathbb{Z}/5\mathbb{Z}, \text{ so we have two finite fields, } \mathbb{F}_5^2 \text{ and } \mathbb{F}_5.
\]

The prime $p = 13$ would have been a better choice because $F$ is irreducible over $\mathbb{Z}/13\mathbb{Z}$.

(2) In $\mathbb{F}_2$: $f = (X+2a+1) \cdot (X-2a+2) \cdot (X^2+2ax-1) \cdot (X^2-(2a-1)X-1)$ and

\[
in \mathbb{F}_5: \quad f = (X^2-2X-1) \cdot (X^4+X^3-X+1).
\]

We see at once that $f$ can not have a factor of degree 1, 3 or 5.

(3) $p^k \geq 2 \cdot 1.6 \cdot (5^3) \cdot 3^2 \cdot 12 / \sqrt{23} \approx 720$, e.g. $k = 5$.

\[
F(T) = (T^2 - 1228T - 1392) \cdot (T + 1228) \quad \text{over } \mathbb{Z}/5\mathbb{5}\mathbb{Z}.
\]

(4) $f = (X+772a+296) \cdot (X-772a+1437) \cdot (X^2-1228ax-1) \cdot (X+(1228a+1391)X-1) \quad \text{modulo } 5^5$ and $a^2 - 1228a - 1392$,

\[
f = (X^2-1392X-1) \cdot (X^4 + 1391X^3 - 1230X^2-1391X+1) \quad \text{modulo } 5^5.
\]

(5) Combining these two factorizations with the Chinese Remainder Theorem gives: $f = (X^2+(a^2-1)X-1) \cdot (X^4-a^2X^3+(a-2)X^2+a^2X+1)$. 
B. REFERENCES


