Generating Subfields

Mark van Hoeij[*]
Florida State University
Tallahassee, FL 32306
hoeij@math.fsu.edu

Jürgen Klüners[†]
Mathematisches Institut
der Universität Paderborn
Universität Paderborn
Warburger Str. 100
33098 Paderborn, Germany
klueners@math.uni-paderborn.de

Andrew Novocin
Laboratoire LIP (U. Lyon, CNRS, ENS Lyon, INRIA, UCBL)
46 Allée d'Italie
69364 Lyon Cedex 07, France
andy@novocin.com

ABSTRACT

Given a field extension K/k of degree n we are interested in finding the subfields of K containing k. There can be more than polynomially many subfields. We introduce the notion of generating subfields, a set of up to n subfields whose intersections give the rest. We provide an efficient algorithm which uses linear algebra in k or lattice reduction along with factorization in any extension of K. Our implementation shows that previously difficult cases can now be handled.

1. INTRODUCTION

Let K/k be a finite separable field extension of degree n and α a primitive element of K over k with minimal polynomial $f \in k[x]$. We explore the problem of computing subfields of K which contain k. We prove that all such subfields (there might be more than polynomially many) can be expressed as the intersections of at most n particular subfields which we will call the 'generating subfields'. We give an efficient algorithm to compute these generating subfields.

Previous methods progress by solving combinatorial problems on the roots of f, such as [4, 5, 8, 13]. Similar to our algorithm [11] starts by factoring f over K and then tries to find all subfield polynomials (see Definition 2.5) by a combinatorial approach. Such approaches can be very efficient, but in the worst cases they face a combinatorial explosion. While [14] proceeds by factoring resolvent polynomials of degree bounded by $\binom{n}{\lfloor n/2 \rfloor}$. By introducing the concept of generating subfields we restrict our search to a small number of target subfields. This new fundamental object allows for polynomial time algorithms.

We can find the generating subfields whenever we have a factorization algorithm for f over K or any $\tilde{K}/K$ and the ability to compute a kernel in k. For $k = \mathbb{Q}$ this implies a polynomial-time algorithm as factoring over $\mathbb{Q}(\alpha)$ and linear algebra over $k = \mathbb{Q}$ are polynomial time. When one desires all subfields we give such an algorithm which is additionally linear in the number of subfields.

For the number field case we are interested in a specialized and practical algorithm. Thus we replace exact factorization over $\mathbb{Q}(\alpha)$ by a p-adic factorization and the exact kernel computation by approximate linear algebra using the famous LLL algorithm for lattice reduction [15]. We take advantage of some recent practical lattice reduction results [18] and tight theoretical bounds to create an implementation which is practical on previously difficult examples.

ROADMAP: The concept of the principal and generating subfields are introduced in Section 2.1. In Section 2.2 we explain how to compute all subfields in a running time which is linearly dependent on the number of subfields. For the number field case we will use the LLL algorithm and this case is handled in detail in Section 3. Finally we compare our approach with the state of the art in Section 4.

NOTATIONS: For a polynomial g we let $\| g \|$ be the ℓ_2 norm on the coefficient vector of g. For a vector $\mathbf{v}$ we let $\mathbf{v}[i]$ be the i^{th} entry. Unless otherwise noted $\| \cdot \|$ will represent the ℓ_2 norm.

2. A GENERAL ALGORITHM

2.1 Generating subfields

In this section we introduce the concept of a generating set of subfields and prove some important properties. Let $\tilde{K}$ be a field containing K. We remark that we can choose $\tilde{K} = K$, but in some case it might be better to choose a larger $\tilde{K}$ from an algorithmic point of view. E.g. in the number field case we choose a p-adic completion (see Section 3). Let $f = f_1 \cdots f_r$ be the factorization of f over $\tilde{K}$ where the $f_i \in \tilde{K}[x]$ are irreducible and $f_1 = x - \alpha$. We define the fields $\tilde{K}_i := \tilde{K}[x]/(f_i)$ for $1 \leq i \leq r$. We denote elements of K as $g(\alpha)$ where $g \in k[x]$ is a polynomial of degree $< n$, and define for $1 \leq i \leq r$ the embedding

$$\phi_i : K \to \tilde{K}_i, \qquad g(\alpha) \mapsto g(x) \bmod f_i.$$

Note that ϕ_1 is just the identity map $\mathrm{id} : K \to \tilde{K}$. We define for $1 \leq i \leq r$:

$$L_i := \mathrm{Ker}(\phi_i - \mathrm{id}) = \{g(\alpha) \in K \mid g(x) \equiv g(\alpha) \bmod f_i\}.$$

[*]Supported by NSF Grant number 0728853 and 1017880

[†]Research supported by the Deutsche Forschungsgemeinschaft

The L_i are closed under multiplication, and hence fields, since $\phi_i(ab) = \phi_i(a)\phi_i(b) = ab$ for all $a, b \in L_i$.

THEOREM 2.1. *If L is a subfield of K/k then L is the intersection of L_i, $i \in I$ for some $I \subseteq \{1, \ldots, r\}$.*

PROOF. Let f_L be the minimal polynomial of α over L. Then f_L divides f since $k \subseteq L$, and $f_L = \prod_{i\in I} f_i$ for some $I \subseteq \{1, \ldots, r\}$ because $L \subseteq \tilde{K}$. We will prove

$$L = \{g(\alpha) \in K \mid g(x) \equiv g(\alpha) \bmod f_L\} = \bigcap_{i\in I} L_i.$$

If $g(\alpha) \in L$ then $h(x) := g(x) - g(\alpha) \in L[x]$ is divisible by $x - \alpha$ in $K[x]$. The set of polynomials in $L[x]$ divisible by $x - \alpha$ is the principal ideal (f_L) by definition of f_L. Then $h(x) \equiv 0 \bmod f_L$ and hence $g(x) \equiv g(\alpha) \bmod f_L$. Conversely, $g(x) \bmod f_L$ is in $L[x]$ (mod f_L) because division by f_L can only introduce coefficients in L. So if $g(x) \equiv g(\alpha) \bmod f_L$ then $g(\alpha) \in K \cap L[x] = L$.

By separability and the Chinese remainder theorem, one has $g(x) \equiv g(\alpha) \bmod f_L$ if and only if $g(x) \equiv g(\alpha) \bmod f_i$ (i.e. $g(\alpha) \in L_i$) for every $i \in I$. $\square$

LEMMA 2.2. *The set $S := \{L_1, \ldots, L_r\}$ is independent of the choice of $\tilde{K}$.*

PROOF. Let $f = g_1 \cdots g_s \in K[x]$ be the factorization of f into irreducible factors over K. Suppose that f_i divides g_l. Let L resp. L_i be the subfield corresponding to g_l resp. f_i. Assume $g(\alpha) \in L$, in other words $g(x) \equiv g(\alpha) \bmod g_l$. Then $g(x) \equiv g(\alpha) \bmod f_i$ because f_i divides g_l. Hence $g(\alpha) \in L_i$.

Conversely, assume that $g(\alpha) \in L_i$. Now $h(x) := g(x) - g(\alpha)$ is divisible by f_i, but since $h(x) \in L_i[x] \subseteq K[x]$ it must also be divisible by g_l since g_l is irreducible in $K[x]$ and divisible by f_i. So $g(x) \equiv g(\alpha) \bmod g_l$ in other words $g(\alpha) \in L$. It follows that $L = L_i$. $\square$

DEFINITION 2.3. *We call the fields $L_1, \ldots, L_r$ the* principal subfields *of K/k. A set S of subfields of K/k is called a* generating set *of K/k if every subfield of K/k can be written as $\bigcap T$ for some $T \subseteq S$. Here $\bigcap T$ denotes the intersection of all $L \in T$, and $\bigcap \emptyset$ refers to K. A subfield L of K/k is called a* generating subfield *if it satisfies the following equivalent conditions*

1. *The intersection of all fields L' with $L \subsetneq L' \subseteq K$ is not equal to L.*
2. *There is precisely one field $L \subsetneq \tilde{L} \subseteq K$ for which there is no field between L and $\tilde{L}$ (and not equal to L or $\tilde{L}$).*

The field $\tilde{L}$ in condition 2. is called *the field right above L*. It is clear that $\tilde{L}$ is the intersection in condition 1., so the two conditions are equivalent.

The field K is a principal subfield but not a generating subfield. A maximal subfield of K/k is a generating subfield as well. Theorem 2.1 says that the principal subfields form a generating set. By condition 1., a generating subfield can not be obtained by intersecting larger subfields, and must therefore be an element of every generating set. In particular, a generating subfield is also a principal subfield.

If S is a generating set, and we remove every $L \in S$ for which $\bigcap\{L' \in S | L \subsetneq L'\}$ equals L, then what remains is a generating set that contains only generating subfields. It follows that

PROPOSITION 2.4. *S is a generating set if and only if every generating subfield is in S.*

Here we just want to illustrate the requirements for finding a generating set of subfields in polynomial time. Suppose that K/k is a finite separable field extension and that one has polynomial time algorithms for factoring over K and linear algebra over k (for example when $k = \mathbb{Q}$). Then applying Theorem 2.1 with $\tilde{K} = K$ yields a generating set S with $r \leq n$ elements in polynomial time. We may want to minimize r by removing all elements of S that are not generating subfields, then $r \leq n - 1$.

Note that the computation of the principal subfields L_i is trivial when we know a factorization of f over K. In this case we get a k-basis of L_i by a simple kernel computation. In the number field case, the factorization of f over K is the bottleneck. Therefore for some fields k we prefer to take a larger field $\tilde{K} \supsetneq K$ where the factorization is faster. In Section 3 this is done for $k = \mathbb{Q}$, but this can be generalized to an arbitrary global field. Then we let $\tilde{K}$ be some completion of K. This reduces the cost of the factorization, however, one now has to work with approximations for the factors f_i of f, which means that we get approximate (if $\tilde{K}$ is the field of p-adic numbers then this means modulo a prime power) linear equations. Solving approximate equations involves LLL in the number field case and [2, 7] in the function field case.

2.2 All subfields

Now suppose that one would like to compute all subfields of K/k by intersecting elements of a generating set $S = \{L_1, \ldots, L_r\}$. We present an algorithm with complexity proportional to the number of subfields of K/k. Unfortunately there exist families of examples where this number is more than polynomial in n. Note that we have represented our subfields $k \leq L_i \leq K$ as k-vector subspaces of K. This allows the intersection $L_1 \cap L_2$ to be found with linear algebra as the intersection of two subspaces of a vector space. To each subfield L of K/k we associate a tuple $e = (e_1, \ldots, e_r) \in \{0, 1\}^r$, where $e_i = 1$ if and only if $L \subseteq L_i$.

Algorithm AllSubfields
Input: A generating set $S = \{L_1, \ldots, L_r\}$ for K/k.
Output: All subfields of K/k.

```
1. Let $e := (e_1, \ldots, e_r)$ be the associated tuple of $K$.
2. ListSubfields := $[K]$.
3. Call NextSubfields($S, K, e, 0$).
4. Return ListSubfields.
```

The following function returns no output but appends elements to ListSubfields, which is used as a global variable. The input consists of a generating set, a subfield L, its associated tuple $e = (e_1, \ldots, e_r)$, and the smallest integer $0 \leq s \leq r$ for which $L = \bigcap\{L_i \mid 1 \leq i \leq s,\ e_i = 1\}$.

Algorithm NextSubfields
Input: S, L, e, s.

```
For all $i$ with $e_i = 0$ and $s < i \leq r$ do
    1. Let $M := L \cap L_i$.
    2. Let $\tilde{e}$ be the associated tuple of $M$.
```

3. **If** $\tilde{e}_j \leq e_j$ for all $1 \leq j < i$ **then** append M to ListSubfields and call NextSubfields$(S, M, \tilde{e}, i)$.

DEFINITION 2.5. *Let L be a subfield of K/k. Then the minimal polynomial f_L of α over L is called the* **subfield polynomial** *of L.*

REMARK 2.6. *Let $g \in K[x]$ be a monic polynomial. Then the following are equivalent:*

1. *$g = f_L$ for some subfield L of K/k.*
2. *$f_1 \mid g \mid f$ and $[\mathbb{Q}(\alpha) : \mathbb{Q}(\text{coefficients}(g))] = \text{degree}(g)$.*
3. *$f_1 \mid g \mid f$ and the $\mathbb{Q}$–vector space $\{h(x) \in \mathbb{Q}[x] \mid \deg(h) < \deg(f), h \mod g = h \mod f_1\}$ has dimension $\deg(f)/\deg(g)$.*

REMARK 2.7. *For each subfield L, we can compute subfield polynomial f_L with linear algebra. Testing if $L \subseteq M$ then reduces to testing if f_L is divisible by f_M. For many fields K this test can be implemented efficiently by choosing a non-archimedian valuation v of K with residue field $\mathbf{F}$ such that the f mod v (the image of f in $\mathbf{F}[x]$) is defined and separable. Then f_L is divisible by f_M in $K[x]$ if and only if the same is true mod v, since both are factors of a polynomial f whose discriminant does not vanish mod v.*

Subfields that are isomorphic but not identical are considered to be different in this paper. Let m be the number of subfields of K/k. Since S is a generating set, all subfields occur as intersections of $L_1, \ldots, L_r$. The condition in Step (3) in Algorithm NextSubfields holds if and only if M has not already been computed before. So each subfield will be placed in ListSubfields precisely once, and the total number of calls to Algorithm NextSubfields equals m. For each call, the number of i's with $e_i = 0$ and $s < i \leq r$ is bounded by r, so the total number of intersections calculated in Step (1) is $\leq rm$. Step (2) involves testing which L_j contain M. Bounding the number of j's by r, the number of subset tests is $\leq r^2 m$. One can implement Remark 2.7 to keep the cost of each test low.f

THEOREM 2.8. *Given a generating set for K/k with r elements, Algorithm AllSubfields returns all subfields by computing at most rm intersections and at most r^2m subset tests, where m is the number of subfields of K/k.*

2.3 Quadratic subfields

We've mentioned that there might be more than polynomially many subfields. We have presented an algorithm which efficiently computes a set of generating subfields. This set includes all maximal subfields. As a theoretical application, to illustrate this framework, we note that all quadratic subfields can be computed in polynomial time when we already know the generating subfields. Note that during our discussion we encounter a field extension with Galois group C_2^s, which is the simplest example of a field extension which has more than polynomially many subfields.

Let $Q(K/k)$ denote the subfield generated over k by $\{a \in K | a^2 \in k\}$, and let C_2 denote the cyclic group of order 2. If $K = Q(K/k)$, in other words the Galois group of f is C_2^s for some s, then $n = 2^s$ and f splits over K into linear factors $f_1 \cdots f_n$ where $f_1 = x - \alpha$. Furthermore, there are precisely $n-1$ generating subfields $L_2, \ldots, L_n$ and n principal subfields $L_1, \ldots, L_n$ where $L_1 = K$.

Conversely, suppose there are n principal subfields. Every principal subfield corresponds to at least one factor of f over K, and hence to precisely one factor since f has degree n. So f must split into linear factors, and each L_i corresponds to precisely one linear factor f_i. Then the minimal polynomial of α over L_i is $f_1 f_i$ when $i \in \{2, \ldots, n\}$. The degree of $f_1 f_i$ is 2, so there are $n-1$ subfields of index 2, which implies that the Galois group is C_2^s for some s.

THEOREM 2.9. *If factoring over K and linear algebra over k can be done in polynomial time then all quadratic subfields of K/k can be computed in polynomial time.*

Note that a subfield of index 2 of K/k corresponds to an autmorphism of K/k of order 2 which can be easily computed. Therefore the knowledge of all principal subfields of $Q(K/k)$ is equivalent to the knowledge of all automorphisms of the Galois group. Hence, the quadratic subfields of $Q(K/k)$ can be computed easily in polynomial time. So it suffices to prove that the following algorithm computes $Q(K/k)$ in polynomial time.

Algorithm Q
Input: A separable field extension K/k where $K = k(\alpha)$.
Output: $Q(K/k)$.

1. Let $n := [K : k]$. If n is odd then return k.
2. Compute the set S of generating subfields.
3. If K/k has $n-1$ distinct subfields of index 2 then return K.
4. Choose a generating subfield $L_i \in S$ with index > 2, and let $\tilde{L}_i$ be the field right above L_i, so $L_i \subsetneq \tilde{L}_i := \bigcap \{L_j \in S \,|\, L_i \subsetneq L_j\}$.
5. If $[\tilde{L}_i : L_i] = 2$ then return $Q(\tilde{L}_i/k)$, otherwise return $Q(L_i/k)$.

In the first call to Algorithm Q, we can compute a generating set in Step (2) in polynomial time using Theorem 2.1 with $\tilde{K} := K$. For the recursive calls we use:

REMARK 2.10. *If S is a generating set for K/k and if L is a subfield of K/k, then $\{L \bigcap L' | L' \in S\}$ is a generating set of L/k.*

For Step (3) see the remarks before Theorem 2.9. If we reach Step (4) then $K \neq Q(K/k)$. The field L_i in Step (4) exists by Lemma 2.11 below. Let $\tilde{L}_i$ be the field right above L_i. If $[\tilde{L}_i : L_i] = 2$ then $\tilde{L}_i \neq K$ so the algorithm terminates.

Let $a \in Q(K/k)$. We may assume that $a^2 \in k$. Now $\tilde{L}_i$ is contained in any subfield L' of K/k that properly contains L_i. So if $a \notin L_i$ then $L_i(a)$ contains $\tilde{L}_i$ and hence equals $\tilde{L}_i$ since $[L_i : L_i(a)] = 2$. Then $a \in \tilde{L}_i$. We conclude $Q(K/k) \subseteq \tilde{L}_i$. If $[\tilde{L}_i : L_i] \neq 2$ then the assumption $a \notin L_i$ leads to a contradiction since $L_i(a)$ can not contain $\tilde{L}_i$ in this case. So $Q(K/k) \subseteq L_i$ in this case, which proves that Step (5) is correct.

LEMMA 2.11. *If K/k does not have $n-1$ distinct subfields of index 2 then there exists a generating field of index > 2.*

PROOF. Assume that every generating (and hence every maximal) subfield has index 2. So the subfields of index 2 form a generating set. Let G be the automorphism group of K/k. If K/L_i and K/L_j are Galois extensions, then so is $K/(L_i \cap L_j)$ since $L_i \cap L_j$ is the fixed field of the group generated by the Galois groups of K/L_i and K/L_j. If $[K : L_i] = 2$ then K/L_i is Galois. Let k' be the intersection of all subfields L_i of index 2. Then K/k' is Galois. However, k' must equal k, otherwise the set of subfields of index 2 can not be a generating set. It follows that K/k is Galois.

If n is not a power of 2, then there exists a maximal subfield of odd index. If $n = 2^s$ with $s > 1$ then the Galois group must have an element of order 4 (G can not be C_2^s since the number of subfields of index 2 is not $n - 1$). This element of order 4 corresponds to a linear factor f_i of f in $K[x]$. Let L_i be its corresponding principal subfield. Then L_i is contained in m maximal subfields where m is either 1 or 3. Let $\check{f}_i$ be the minimal polynomial of α over L_i. If $m = 3$ then every irreducible factor of $\check{f}_i/(x-\alpha)$ corresponds to a subfield of index 2. This is a contradiction since f_i divides $\check{f}_i/(x-\alpha)$. $\square$

3. THE NUMBER FIELD CASE

3.1 Introduction

In this section we describe an algorithm for producing a generating set when $K = \mathbb{Q}(\alpha)$. Factoring f over K, though polynomial time, is slow, thus we prefer to use an approximation of a p-adic factorization and LLL. We show that when the algorithm terminates[1], it returns the correct output.

For a prime number p, let $\mathbb{Q}_p$ denote the field of p-adic numbers, $\mathbb{Z}_p$ the ring of p-adic integers, and $\mathbf{F}_p = \mathbb{Z}/(p)$. We choose a prime number p with these three properties: p does not divide the leading coefficient of $f \in \mathbb{Z}[x]$, the image $\overline{f}$ of f in $\mathbf{F}_p[x]$ is separable, and has at least one linear factor which we denote $\overline{f}_1$ (asymptotically, the probability that a randomly chosen prime p has these properties is $\geq 1/n$, where equality holds when K/k is Galois).

By factoring $\overline{f}$ in $\mathbf{F}_p[x]$ and applying Hensel lifting, we obtain a factorization of $f = f_1 \cdots f_r$ over $\mathbb{Q}_p$ where f_1 has degree 1. By mapping $\alpha \in K$ to the root α_1 of f_1 in $\mathbb{Q}_p$ we obtain an embedding $K \to \mathbb{Q}_p$, and so we can view K as a subfield of $\tilde{K} := \mathbb{Q}_p$.

The advantage of taking $\mathbb{Q}_p$ (instead of K) for $\tilde{K}$ is that it saves time on factoring f over $\tilde{K}$. Since p does not divide the denominators of the coefficients of f, the factors $f_1, \ldots, f_r$ of f over $\mathbb{Q}_p$ lie in $\mathbb{Z}_p[x]$. We can not compute these factors with infinite accuracy, but only to some finite accuracy a, meaning that $f_1, \ldots, f_r$ are only known modulo p^a.

For each of the factors, f_i, we will need to find the principal subfield L_i which was defined in Section 2.1 as the kernel of $\phi_i - \mathrm{id}$. To do this we will make use of a knapsack-style lattice in the style of [18]. To get the best performance we would like to design a lattice such that boundably short vectors correspond with elements in L_i.

A natural approach would be to use $1, \alpha, \ldots, \alpha^{n-1}$ as a basis, and search for linear combinations whose images under $\phi_i - \mathrm{id}$ are 0 (mod p^a). However, we will use a different basis. Denote $\mathbb{Z}[\alpha]_{<n} := \mathbb{Z} \cdot \alpha^0 + \cdots + \mathbb{Z} \cdot \alpha^{n-1}$ (note: if f is monic then this is simply $\mathbb{Z}[\alpha]$ but we do not assume that f is monic). Then the basis $\frac{1}{f'(\alpha)}, \ldots, \frac{\alpha^{n-1}}{f'(\alpha)}$ of $\frac{1}{f'(\alpha)} \cdot \mathbb{Z}[\alpha]_{<n}$ allows us to prove more practical bounds (this phenomena has also been observed in other contexts [6]). Using this basis of K we prove the existence of a $\mathbb{Q}$-basis of L_i which has a bounded representation. We delay the proof of this theorem until section 3.4.

[1]a bound for the running time can be obtained in a similar way as in [3]

THEOREM 3.1. *Let L_i, the target principal subfield, have degree m_i over $\mathbb{Q}$. For $\beta \in \frac{1}{f'(\alpha)} \cdot \mathbb{Z}[\alpha]_{<n}$ with $\beta = \sum b_i \frac{\alpha^i}{f'(\alpha)}$ we associate the vector $\mathbf{v}_\beta := (b_0, \ldots, b_{n-1})$. Then there exists m_i linearly independent algebraic numbers $\beta_1, \ldots \beta_{m_i} \in L_i \cap \frac{1}{f'(\alpha)} \cdot \mathbb{Z}[\alpha]_{<n}$ each with $\|\mathbf{v}_{\beta_k}\| \leq n^2 \|f\|_2$.*

3.2 The computation of a principal subfield

Now we can continue the description of the computation of the principal subfield L_i corresponding to the factor f_i of degree d_i. As mentioned before we will represent our elements in the basis $\frac{1}{f'(\alpha)}, \ldots, \frac{\alpha^{n-1}}{f'(\alpha)}$. Each of these basis elements will be represented as the column of an identity matrix to which we attach entries for the image of that basis element under $\phi_i - \mathrm{id}$. Since these images are only known modulo p^a we must also adjoin columns which allow for this modular reduction. Suppose the degree of f_i is d_i, then our lattice is spanned by the columns of the following $(n + d_i) \times (n + d_i)$ integer matrix:

$$B_i := \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ c_{0,0} & \cdots & c_{0,n-1} & p^a & & \\ \vdots & \ddots & \vdots & & \ddots & \\ c_{d_i-1,0} & \cdots & c_{d_i-1,n-1} & & & p^a \end{pmatrix} \quad (1)$$

where $c_{k,j}$ is the kth coefficient of $\frac{x^j}{f'(x)} \bmod f_i - \frac{x^j}{f'(x)} \bmod f_1$ reduced modulo p^a. To interpret a vector $\mathbf{v}$ in the column space of this matrix we take the first n entries $b_0, \ldots, b_{n-1}$ and then compute $(\sum b_j \alpha^j)/f'(\alpha)$. A vector corresponding to an element in L_i will have its final d_i entries be 0 modulo p^a. Thus Theorem 3.1 shows us that the lattice generated by columns of B_i contains a dimension m_i sublattice which has a small basis. This allows us to use the new sub-lattice reduction techniques of [18] on B_i. Thus, rather than standard LLL, we use `LLL_with_removals` which performs lattice reduction but removes any vectors in the final position whose G-S norm is above a given bound. The following lemma is derived from [15] and justifies these removals.

LEMMA 3.2. *Given a basis $\mathbf{b}_1, \ldots, \mathbf{b}_d$ of a lattice Λ, and let $\mathbf{b}_1^*, \ldots, \mathbf{b}_d^*$ be the output of Gram-Schmidt orthogonalization. If $\| \mathbf{b}_d^* \| > B$ then any vector in L with norm $\leq B$ is a $\mathbb{Z}$-linear combination of $\mathbf{b}_1, \ldots, \mathbf{b}_{d-1}$.*

This technique is common and is used in [10, 18]. As the removal condition requires Gram-Schmidt norms we can state that LLL reduced bases tend to be numerically stable for Gram-Schmidt computations so a floating point Gram-Schmidt computation could be used for efficiency (see [19]). Also FLINT 1.6 [9] has an LLL with removals routine which takes a bound and returns the dimension of the appropriate sub-lattice.

In this way using `LLL_with_removals` with the bound from Theorem 3.1 will allow us to reduce the dimension.

In Figure 1 we give a practical algorithm which will create a basis of a subfield of K which is highly likely to be L_i. We will use $D := \mathrm{diag}\{1, \ldots, 1, C, \ldots, C\}$ as a matrix for scaling the last d_i rows of B_i by a scalar C. Since the vectors guaranteed by Theorem 3.1 come from L_i we know that the final d_i entries must be 0. Thus multiplication on the left by D and removals will eventually ensure that vectors with zero entries are found by LLL.

```
Input: $f_i$
Output: $h_k$ which probably generate $L_i$

 1. Create lattice $B_i$ from equation (1)
 2. $A :=$ LLL_with_removals$(B_i, n^2 \parallel f \parallel)$
 3. $m := \dim(A)$
 4. while $\exists l > n, j$ such that $A[l, j] \neq 0$ :
 5.    $A := D \cdot A$
 6.    $A :=$ LLL_with_removals$(A, n^2 \parallel f \parallel)$
 7.    $m := \dim(A)$
 8. if $m \nmid n$ increase precision repeat principal
 9. for $1 \leq k \leq m$:
10.    $h_k := \sum_{j=1}^{n} \frac{A[j,k]x^{j-1}}{f'(x)}$
```

Figure 1: `principal` algorithm

Using LLL on the matrix entire B_i will suffice for this paper. However, in practice the d_i final rows of B_i can also be reduced one at a time. In this way one could potentially arrive at a solution without needing all rows of B_i. Such an approach is seen in [18] and could be adapted to this situation.

The algorithm in figure 1 will produce m p-adic polynomials h_k, which are likely to correspond with algebraic numbers which generate L_i as a $\mathbb{Q}$-vector space. It is possible that m is not m_i but some other divisor of n. In particular, if the p-adic precision is not high enough then there could be entries in the lattice basis which are 0 modulo p^a but not exactly 0. In that case one of the h_k would not be from L_i. Even so the $\mathbb{Q}$-vector space generated by the h_k must at least contain L_i. The reason is that at least m_i linearly independent algebraic numbers from L_i remain within the lattice after `LLL_with_removals` thanks to the bound of Theorem 3.1 and Lemma 3.2.

Theorem 3.1 can also be used to make a guess for a starting precision of p^a. Since any reduced basis has Gram-Schmidt norms within a factor 2^{n+d_i} of the successive minima and the determinant of B_i is $p^{a \cdot d_i}$ then we should ensure than $p^{a \cdot d_i}$ is at least $(2^{n+d_i} n^2 \|f\|)^n$.

3.3 Confirming a principal subfield

In this section we will assume that we have elements in approximate p-adic form which are likely to generate a principal subfield (in other words, the output of the algorithm in Figure 1). Our goal is to certify that the elements indeed generate the target L_i. We give an algorithm which will construct the subfield polynomial g, of L_i or return failure, in which case more p-adic precision is needed. We choose the subfield polynomial as it will provide a proof that we have a principal subfield and can be stored in a relatively compact way thanks to our new basis. Of course other representations and proofs are possible.

From here on our algorithmic objective will be to output the minimal polynomial $g \in L_i[x]$ of α over L_i. This g is the subfield polynomial of L_i and its coefficients generate L_i. We know m elements h_k modulo p^a, we know that $m|n$ and that $\phi_i - \mathrm{id}(h_k) \equiv 0$ modulo p^a for each k. Recall that the h_k were from columns of a lattice basis A. First we will create a p-adic candidate subfield polynomial which we then subject to 3 certification checks.

Candidate g: Create an index set $T := \{j | \phi_j(h_k) \equiv \mathrm{id}(h_k) \mod p^a \forall h_k\}$, that is find the p-adic factors of f which also agree with f_1 on the elements corresponding to the basis from A. T will contain at least 1 and i. Now let $g_{\mathrm{cand}} := \prod_{j \in T} f_j \mod p^a$. This is done in steps 1–5 of Figure 2

```
Input: $h_1 \ldots h_m, f_1, \ldots f_r \in \mathbb{Q}_p[x]$, precision $a$
Output: $g$ subfield poly, or fail

 1. $T := \{\}$
 2. for each $1 \leq j \leq r$:
 3.    if $(h_k \mod f_j = h_k \mod f_1) \mod p^a \forall k$ then:
 4.       $T := T \cup j$
 5. $g_{\mathrm{cand}} := \mathrm{lc}(f) \cdot \prod_{j \in T} f_j \mod p^a$
       where $\mathrm{lc}(f)$ is the leading coefficient of $f$
 6. Create lattice $M$ using (2)
 7. $M :=$LLL$(M)$
 8. $g_{\mathrm{temp}} = 0$
 9. for each coefficient $g_k$ of $x^k$ in $g_{\mathrm{cand}}$:
10.    create $M_{g_k}$ lattice using (3)
11.    Check 1 find $\mathbf{v}$ in LLL$(M_{g_k})$
          with $\mathbf{v}[n+1] = 0$ and $\mathbf{v}[n+2] = 1$
12.    $g_{\mathrm{temp}} := g_{\mathrm{temp}} + \sum_{j=1}^{n} \frac{\mathbf{v}[j]\alpha^{j-1}}{f'(\alpha)} x^k$
13. $g_{\mathrm{cand}} := g_{\mathrm{temp}} \in \mathbb{Q}(\alpha)[x]$
14. Check 2 ensure $g_{\mathrm{cand}} | f$ exactly
15. Check 3 ensure $(h_k \mod g_{\mathrm{cand}} = h_k \mod f_1) \forall k$
16. return $g := g_{\mathrm{cand}}$
```

Figure 2: `final_check` algorithm

Check 1: Let $\Lambda(A) \subseteq \frac{\mathbb{Z}[\alpha]_{<n}}{f'(\alpha)}$ be the lattice generated by the algebraic numbers corresponding with columns of A. We now attempt to find an exact representation of g_{cand} by converting each coefficient into an algebraic number in $\Lambda(A) \cap \frac{\mathbb{Z}[\alpha]_{<n}}{f'(\alpha)}$. We'll do this by attempting to find linear combinations of h_k which exactly equal each coefficient of g_{cand}.

Note that this g_{cand} is a polynomial with p-adic coefficients, these coefficients can be quickly Hensel lifted using the fact that $f = g \cdot (f/g) \mod p^a$ if more precision is needed. Now we want to express these coefficients in the basis $\frac{\mathbb{Z}[\alpha]_{<n}}{f'(\alpha)} \cap \Lambda(A)$. To do this we will use a lattice basis similar to A with a slight adjustment. Rather than finding algebraic numbers whose images under $\phi_i - \mathrm{id}$ are zero, we'll find combinations of the h_k whose p-adic valuations match a coefficient of g_{cand}.

Lets call $\mathbf{v}_{h_k}$ the coefficient vector of h_k, and the corresponding p-adic valuation $c_j := h_k(\alpha_1)$ (that is, h_k modulo f_1). Also we pick a large scalar constant C (to ensure that LLL works on reducing the size of the p-adic row). We let the columns of the new matrix be $(\mathbf{v}_{h_j}, C \cdot c_j)^T$, and the column $(0, \ldots, 0, C \cdot p^a)$.

$$M := \begin{pmatrix} \mathbf{v}_{h_1}^T & \ldots & \mathbf{v}_{h_m}^T & \mathbf{0} \\ C \cdot c_1 & \ldots & C \cdot c_m & C \cdot p^a \end{pmatrix} \tag{2}$$

A vector in the column space of this matrix is a representation of a combination of the elements from h_k along with a p-adic valuation of that element. Now for each coefficient we'll use this matrix to find a combination which matches that coefficient. In practice we LLL-reduce M before adjoining data from the coefficients of g_{cand}, but here we present an augmented M without altering the columns first (for clarity).

For each coefficient g_k of g_{cand} augment each column of M with a zero, then adjoin a new column $(0, \dots, 0, C \cdot g_k, 1)^T$. This is what the coefficient matching matrix looks like:

$$M := \begin{pmatrix} \mathbf{v}_{h_1}^T & \dots & \mathbf{v}_{h_m}^T & \mathbf{0} & \mathbf{0} \\ C \cdot c_1 & \dots & C \cdot c_m & C \cdot p^a & C \cdot g_k \\ 0 & \dots & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

Run LLL on this matrix (provided C is large enough) then find the vector which has its final two entries as 0,1, the first n entries are an expression of g_k in $\frac{\mathbb{Z}[\alpha]_{<n}}{f'(\alpha)}$. If this works for every coefficient of g_{cand} then the check has passed.

Check 2: Ensure that $g_{\mathrm{cand}} | f$ in $\mathbb{Q}(\alpha)[x]$.

Check 3: Ensure that $h_k \mod g_{\mathrm{cand}} = h_k \mod f_1$ for each h_k.

THEOREM 3.3. *If all checks pass then the $\mathbb{Q}$-linear combination of the elements corresponding to the lattice basis A generate L_i the target principal subfield, and g_{cand} is the subfield polynomial of L_i.*

PROOF. By construction of g_{cand} and A we know that the span over $\mathbb{Q}$ of the elements corresponding to A, the h_k, contains L_i. Let's call this span V, so $L_i \subseteq V$. Since g_{cand} divides f and f_i divides g_{cand} then $h \mod g_{\mathrm{cand}} = h \mod f_1$ implies $h \mod f_i = h \mod f_1$. By check 1 this implies that $V \subseteq L_i$ thus the span over $\mathbb{Q}$ of the elements from the lattice is L_i.

Now $x-\alpha$, $f_i | g_{\mathrm{cand}} \mod p^a$ and $g_{\mathrm{cand}} | f$ exactly then $f_i | g_{\mathrm{cand}}$ and $(x-\alpha) | g_{\mathrm{cand}}$ exactly. Now by Remark 2.6 we know g_{cand} is the subfield polynomial of L_i. $\square$

If check 1 fails then perhaps try a larger constant C, otherwise if any check fails increase the p-adic precision via Hensel lifting and try again.

3.4 Bounds for the coefficients

The only aim of this section is to prove Theorem 3.1. The techniques described in this section are not used in the algorithm.

In order to get our desired bounds it is useful to introduce the notation of a codifferent, see [16, Chapter 4.2] for more details.

LEMMA 3.4. *Let $f \in \mathbb{Z}[x]$ be primitive and irreducible, with degree n. Let α be a root of f. Let $\mathcal{O}_K$ be the ring of integers in $K = \mathbb{Q}(\alpha)$ and let $\mathcal{O}_K^*$ be the* co-different *which is defined as:*

$$\mathcal{O}_K^* = \{a \in K | \forall_{b \in O_K} \mathrm{Tr}(ab) \in \mathbb{Z}\}.$$

Then

$$\mathcal{O}_K^* \subseteq \frac{1}{f'(\alpha)} \mathbb{Z}[\alpha]_{<n} \quad (4)$$

.

PROOF. Let $a \in \mathcal{O}_K^*$, so $\mathrm{Tr}(ab) \in \mathbb{Z}$ for any $b \in \mathcal{O}_K$. The content of a polynomial $g = c_0 x^0 + \dots + c_d x^d \in K[x]$ is defined as the fractional ideal $c(g) = \mathcal{O}_K c_0 + \dots + \mathcal{O}_K c_d$. Let $g_1 = x - \alpha$ and $g_2 = f/g_1$. Gauss' lemma says $c(g_1)c(g_2) = c(g_1 g_2)$. Then $c(g_1)c(g_2) = c(f) = \mathcal{O}_K$, ($f$ is primitive) and since g_1 has a coefficient equal to 1 it follows that $c(g_2) \subseteq \mathcal{O}_K$, in other words $g_2 \in O_K[x]$. Now $ag_2 \in a \cdot \mathcal{O}_K[x]_{<n}$ and by definition of $\mathcal{O}_K^*$ we see that $\mathrm{Tr}(ag_2) \in \mathbb{Z}[x]_{<n}$. So

$$\mathrm{Tr}(a \frac{f(x)}{x - \alpha}) = \sum a^{(i)} \frac{f(x)}{x - \alpha^{(i)}} \in \mathbb{Z}[x]_{<n}$$

where $a^{(i)}$ and $\alpha^{(i)}$ denote the conjugates of a and α. Evaluating the right-hand side at $x = \alpha = \alpha^{(1)}$ gives $af'(\alpha) \in \mathbb{Z}[\alpha]_{<n}$ and hence $a \in 1/f'(\alpha) \cdot \mathbb{Z}[\alpha]_{<n}$. $\square$

Now suppose that we have an $\beta \in \mathcal{O}_K^*$, then we can write

$$f'(\alpha)\beta = \sum_{i=0}^{n-1} b_i \alpha^i \text{ with } b_i \in \mathbb{Z}. \quad (5)$$

In our applications β is an element of a principal subfield and we would like to bound the size of b_i. In the following we need the complex embeddings and some norms of algebraic numbers.

DEFINITION 3.5. *Let $K = \mathbb{Q}(\alpha)$ be a number field of degree n and f be the minimal polynomial of α. Then we denote by $\phi_1, \dots, \phi_n : K \to \mathbb{C}, \alpha \mapsto \alpha_i$ the n complex embeddings, where $\alpha_1, \dots, \alpha_n$ are the complex roots of f. We assume that $\alpha_1, \dots, \alpha_{r_1}$ are real and the complex roots are ordered such that $\alpha_{r_1+i} = \bar{\alpha}_{r_1+r_2+i}$ for $1 \le i \le r_2$.*

For $\beta \in K$ we define the norms

$$\|\beta\|_1 := \sum_{i=1}^n |\phi_i(\beta)| \text{ and } \|\beta\|_2 := \sqrt{\sum_{i=1}^n |\phi_i(\beta)|^2}.$$

Note the well known estimates:

$$\|\beta\|_2 \le \|\beta\|_1 \le \sqrt{n}\|\beta\|_2.$$

We are able to give the promised bounds.

LEMMA 3.6. *Let β be given as in (5) with coefficient vector $b := (b_0, \dots, b_{n-1})$. Then we have $\|b\|_2 \le n\|\beta\|_1\|f\|_2 \le n^{1.5}\|\beta\|_2\|f\|_2$.*

PROOF. Let $h(x) := \sum_{i=0}^{n-1} b_i x^i$. Let $\alpha_i := \phi_i(\alpha)$ and $\beta_i := \phi_i(\beta)$, then we get: $h(\alpha_i) = \beta_i f'(\alpha_i)$ for $1 \le i \le n$. Using Lagrange interpolation we get:

$$h(x) = \sum_{i=1}^n \beta_i f'(\alpha_i) \frac{f(x)/(x - \alpha_i)}{f'(\alpha_i)} = \sum_{i=1}^n \beta_i \frac{f(x)}{x - \alpha_i}.$$

Now:

$$\|b\|_2 = \|h\|_2 = \sum_{i=1}^n |\beta_i| \|f/(x - \alpha_i)\|_2$$

$$\le \max_i \|f/(x - \alpha_i)\|_2 \sum_{i=1}^n |\beta_i| \le n\|f\|_2\|\beta\|_1,$$

$\|f/(x - \alpha_i)\|_2 \le n\|f\|_2$ is proved in [17, cor4.7]. The second estimate follows then trivially from $\| \cdot \|_1 \le \sqrt{n} \| \cdot \|_2$. $\square$

Now our goal is the following. Let L be a principal subfield of degree m which we would like to compute. We want to find a $\mathbb{Q}$-basis of L represented in our $\frac{1}{f'(\alpha)}\mathbb{Z}[\alpha]_{<n}$–basis. Note that $\mathcal{O}_L^* \subseteq \mathcal{O}_K^* \subseteq \frac{1}{f'(\alpha)} \cdot \mathbb{Z}[\alpha]_{<n}$. In order to apply Lemma 3.6 we need to bound $\|\beta_i\|_2$ for m linearly independent elements $\beta_1, \dots, \beta_m \in L$. We will use the following theorem.

Theorem 3.7 (Banaszczyk). *Let $\Lambda \subset \mathbb{R}^m$ be a lattice and denote by $\Lambda^* := \{y \in \mathbb{R}^m \mid \forall x \in \Lambda : \langle x, y\rangle \in \mathbb{Z}\}$ the dual lattice. Furthermore denote by λ_i, λ_i^* the i-th successive minima of Λ, Λ^*, respectively. Then $\lambda_i \lambda_{m+1-i}^* \leq m$ for $1 \leq i \leq m$.*

The proof can be found in [1, Theorem 2.1]. In our application we will have that $\lambda_1 = \sqrt{m}$, so we get the upper bound $\lambda_m^* \leq \sqrt{m}$. There are canonical ways to map number fields to lattices, but we have the slight problem that the bilinear form $L \times L \to \mathbb{Q}, (x, y) \mapsto \mathrm{Tr}(xy)$ is not positive definite, if L has non-real embeddings. We assume the same order of the complex embeddings of L as in Definition 3.5, so we have $m = r_1 + 2r_2$. Defining $\gamma_i = \phi_i(\gamma)$ and $\delta_i = \phi_i(\delta)$ we get:

$$\mathrm{Tr}(\gamma\delta) = \sum_{i=1}^{m} \gamma_i \delta_i.$$

The corresponding scalar product looks like:

$$\langle \gamma, \delta\rangle := \sum_{i=1}^{m} \gamma_i \bar{\delta}_i.$$

For totally real number fields L those two notions coincide and then we get that the dual lattice equals $\mathcal{O}_L^*$ and we can apply Theorem 3.7 directly to get the desired bounds. First we introduce the canonical real lattice $\Lambda := \Psi(\mathcal{O}_L) \subseteq \mathbb{R}^m$ associated to $\langle \gamma, \delta\rangle$ via

$$\Psi : L \to \mathbb{R}^m, \tag{6}$$

$\beta \mapsto (\beta_1, \ldots, \beta_{r_1}, \sqrt{2}\Re(\beta_{r_1+1}), \ldots, \sqrt{2}\Re(\beta_{r_1+r_2}),$
$\sqrt{2}\Im(\beta_{r_1+1}), \ldots, \sqrt{2}\Im(\beta_{r_1+r_2}))$.
Note that now the standard scalar product of $\mathbb{R}^m$ coincides with the (complex) scalar product defined above. This is the reason for the weight $\sqrt{2}$ in the above definition. Denote by $\langle \cdot, \cdot\rangle_1$ the standard scalar product of $\mathbb{R}^m$. Furthermore denote by

$$\langle x, y\rangle_2 := \sum_{i=1}^{r_1+r_2} x_i y_i - \sum_{i=r_1+r_2+1}^{m} x_i y_i.$$

Then we have

$$\langle \gamma, \delta\rangle = \langle \Psi(\gamma), \Psi(\delta)\rangle_1 \text{ and } \mathrm{Tr}(\gamma\delta) = \langle \Psi(\gamma), \Psi(\delta)\rangle_2.$$

Now we are able to compare our two dual objects, the dual lattice Λ^* of Λ corresponding to $\langle \cdot, \cdot\rangle_1$ and the codifferent.

Lemma 3.8. *Using the above notations. Then $\theta : \mathbb{R}^m \to \mathbb{R}^m$,*

$$(x_1, \ldots, x_m) \mapsto (x_1, \ldots, x_{r_1+r_2}, -x_{r_1+r_2+1}, \ldots, -x_m)$$

induces an isomorphism $\Lambda^* \to \Psi(\mathcal{O}_L^*)$ of $\mathbb{Z}$–modules.

Proof. θ is linear and has the property

$$\langle x, y\rangle_1 = \langle x, \theta(y)\rangle_2 \text{ for all } x, y \in \mathbb{R}^m.$$

We need to show that $\theta(\Lambda^*) = \Psi(\mathcal{O}_L)$. Note that θ^2 is the identity and therefore this is equivalent to $\theta(\Psi(\mathcal{O}_L)) = \Lambda^*$. Denote by $\omega_1, \ldots, \omega_m$ a $\mathbb{Z}$–basis of $\mathcal{O}_L$. Then $\Lambda = \mathbb{Z}\Psi(\omega_1) + \ldots + \mathbb{Z}\Psi(\omega_m)$. Choose $\gamma \in \mathcal{O}_L^*$ arbitrarily. Then $\mathrm{Tr}(\omega_i\gamma) \in \mathbb{Z}$ for $1 \leq i \leq m$ and therefore

$$\langle \Psi(\omega_i), \theta(\Psi(\gamma))\rangle_1 = \langle \Psi(\omega_i), \Psi(\gamma))\rangle_2 = \mathrm{Tr}(\omega_i\gamma) \in \mathbb{Z}.$$

Therefore $\theta(\Psi(\gamma)) \in \Lambda^*$ and we have shown $\theta(\Psi(\mathcal{O}_L^*)) \subseteq \Lambda^*$. Denote by $\tau_1, \ldots, \tau_m \in \mathcal{O}_L^*$ the dual basis of $\omega_1, \ldots, \omega_m$. Because of duality (e.g. see [16, Proof of Prop. 4.14]) we know that $\mathrm{disc}(\tau_1, \ldots, \tau_m) = \mathrm{disc}(\omega_1, \ldots, \omega_m)^{-1} = d_L^{-1}$. Furthermore $\theta(\Psi(\tau_i))$ $(1 \leq i \leq m)$ are linearly independent elements of Λ^* and the discriminant of the $\mathbb{Z}$–module generated by those elements is $|d_L^{-1}|$ since the corresponding determinants differ by a power of -1 because we have to consider the twists between our two bilinear forms. Therefore we know a subset $\theta(\Psi(\mathcal{O}_L^*)) \subseteq \Lambda^*$ which has the correct lattice discriminant. Therefore we get equality. □

Now we are able to get our bound by applying Lemma 3.8 and Theorem 3.7.

Lemma 3.9. *Let L be a number field of degree m. Then $\mathcal{O}_L^*$ contains m $\mathbb{Q}$–linearly independent elements $\gamma_1, \ldots, \gamma_m$ such that $\|\gamma_i\|_2 \leq \sqrt{m}$ for $1 \leq i \leq m$.*

Proof. As before let $\Lambda := \Psi(\mathcal{O}_L)$, where Ψ is defined in (6). Now we claim that the first successive mimimum λ_1 equals $\sqrt{m}$ by taking the element $\Psi(1)$. Let $\gamma \in \mathcal{O}_L$. Then

$$1 \leq |\mathrm{Norm}(\gamma)| = \left(\prod_{i=1}^{m} |\gamma_i|^2\right)^{1/2} \leq \left(\frac{\sum_{i=1}^{m} |\gamma_i|^2}{m}\right)^{m/2}$$

$$= \left(\frac{\langle \Psi(\gamma), \Psi(\gamma)\rangle_1}{m}\right)^{m/2},$$

where the inequality is the one between geometric and arithmetic means. Now we get that $\langle \Psi(\gamma), \Psi(\gamma)\rangle_1 \geq m$ which finishes the proof that $\lambda_1 = \sqrt{m}$.

Applying Theorem 3.7 we find m linearly independent elements $y_1, \ldots, y_m \in \Lambda^*$ with euclidean length bounded by $m/\sqrt{m} = \sqrt{m}$. By using Lemma 3.8 we find elements $\theta(y_i) \in \Psi(\mathcal{O}_L^*)$ which have the same euclidean length. By choosing $\gamma_i := \Psi^{-1}(\theta(y_i))$ for $1 \leq i \leq m$ we finish our proof. □

Now we are able to prove our theorem. Note that the field L takes the role of the principal subfield L_i in the statement.

Proof of Theorem 3.1. Using Lemma 3.9 we find m_i linearly independent elements β_j in $\mathcal{O}_L^*$ with 2-norm bounded by $\sqrt{m_i}$. When we interpret those elements in K, we get n/m_i copies of the complex embeddings, which gives that the 2-norm as elements of K is bounded by $\sqrt{n}$. Now apply Lemma 3.6. □

4. AN EXAMPLE

The aim of this section is to compare our algorithm with the previous state of the art. We want to indicate that our approach can be useful in practice. The algorithm most efficient in practice at the time of this paper is based on [12]. That algorithm uses a combinatorial approach in order to find block systems corresponding to a subfield. The drawback of that algorithm is that it might have to test exponentially many possibilities before it finds the right block system.

Our algorithm is more robust. By working only on the generating subfields, and doing that in a practical way, we ensure an attack which is consistently strong. We compare our algorithm with [12] by taking an example which was given in the [12] paper.

We use the degree 60 field generated by a root of the polynomial

$f(t) := t^{60} + 36t^{59} + 579t^{58} + 5379t^{57} + 30720t^{56} + 100695t^{55} + 98167t^{54} - 611235t^{53} - 2499942t^{52} - 1083381t^{51} + 15524106t^{50} + 36302361t^{49} - 22772747t^{48} - 205016994t^{47} - 194408478t^{46} + 417482280t^{45} + 954044226t^{44} + 281620485t^{43} - 366211766t^{42} - 1033459767t^{41} - 8746987110t^{40} - 15534020046t^{39} + 23906439759t^{38} + 104232578583t^{37} + 31342660390t^{36} - 364771340802t^{35} - 547716092637t^{34} + 583582152900t^{33} + 2306558029146t^{32} + 998482693677t^{31} - 3932078004617t^{30} - 5195646620046t^{29} + 2421428069304t^{28} + 10559164336236t^{27} + 3475972372302t^{26} - 22874708335419t^{25} - 33428241525914t^{24} + 21431451023271t^{23} + 90595197659892t^{22} + 50882107959528t^{21} - 67090205528313t^{20} - 117796269461541t^{19} - 74369954660792t^{18} + 25377774560496t^{17} + 126851217660123t^{16} + 104232393296166t^{15} - 29072256729168t^{14} - 83163550972215t^{13} - 24296640395870t^{12} + 14633584964262t^{11} + 8865283658688t^{10} + 5364852154893t^{9} - 1565702171883t^{8} - 7601782249737t^{7} - 2106132289551t^{6} + 3369356619543t^{5} + 3717661159674t^{4} + 1754791133184t^{3} + 573470363592t^{2} + 74954438640t + 3285118944$

which is the splitting field of the polynomial $t^5 + t^4 - 2t^3 + t^2 + t + 1$. The Galois group of this polynomial is the alternating group A_5 and therefore all elements have order $1, 2, 3$, or 5.

In [12] these subfields were found using clues about this particular example by assuming that it was not some random degree 60 polynomial but something specifically constructed. Requiring clues and tricks it was able to reduce an impossible combinatorial problem to something which was solvable in a couple of hours. Our algorithm does not rely on tricks (the polynomial can again be treated as random) and can find each principal subfield in 3–5 seconds on the same machine that ran the [12] code.

5. REFERENCES

[1] W. Banaszczyk. New bounds in some transference theorems in the geometry of number. *Math. Ann*, 296:625–635, 1993.

[2] Bernhard Beckermann and George Labahn. A uniform approach for the fast computation of matrix-type pade approximants. *SIAM J. Matrix Anal. Appl.*, 15:804–823, July 1994.

[3] Karim Belabas, Mark van Hoeij, Jürgen Klüners, and Allan Steel. Factoring polynomials over global fields. *J. Théor. Nombres Bordeaux*, 21:15–39, 2009.

[4] D. Casperson and J. McKay. Symmetric functions, m-sets, and Galois groups. *Math. Comput.*, 63:749–757, 1994.

[5] Henri Cohen and Francisco Diaz y Diaz. A polynomial reduction algorithm. *Séminaire de Théorie des Nombres de Bordeaux 2*, 3:351–360, 1991.

[6] Xavier Dahan and Éric Schost. Sharp estimates for triangular sets. In *Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, ISSAC '04, pages 103–110, New York, NY, USA, 2004. ACM.

[7] H. Derksen. An algorithm to compute generalized Padé-Hermite forms. Preprint, Catholic University Nijmegen, 1994.

[8] J Dixon. Computing subfields in algebraic number fields. *J. Austral. Math. Soc. Series A*, 49:434–448, 1990.

[9] W. Hart. Flint. open-source C-library `http://www.flintlib.org`.

[10] Mark Van Hoeij. Factoring polynomials and the knapsack problem. *J. Number Theory*, 95:167–189, 2002.

[11] A. Hulpke. Block systems of a Galois group. *Exp. Math.*, 4(1):1–9, 1995.

[12] J. Klüners. *Über die Berechnung von Automorphismen und Teilkörpern algebraischer Zahlkörper.* Dissertation, Technische Universität Berlin, 1997.

[13] J. Klüners. On computing subfields - a detailed description of the algorithm. *Journal de Théorie des Nombres de Bordeaux*, 10:243–271, 1998.

[14] D. Lazard and A. Valibouze. Computing subfields: Reverse of the primitive element problem. In A. Galligo F. Eyssete, editor, *MEGA-92, Computational algebraic geometry*, volume 109, pages 163–176. Birkhäuser, Boston, 1993.

[15] A. K. Lenstra, H. W. Lenstra Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[16] Władisław Narkiewicz. *Elementary and Analytic Theory of Algebraic Numbers.* Springer, 2004.

[17] M. v. Hoeij and V. Pal. Isomorphisms of algebraic number fields. arXiv:1012.0096v2, 2010.

[18] Mark van Hoeij and Andrew Novocin. Gradual sub-lattice reduction and a new complexity for factoring polynomials. In *LATIN*, pages 539–553, 2010.

[19] G. Villard. Certification of the QR factor R, and of lattice basis reducedness. In *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation (ISSAC'07)*, pages 361–368, 2007.