Open access • Proceedings Article • DOI:10.1145/1576702.1576741

Computation schemes for splitting fields of polynomials — Source link

Sébastien Orange, Guénaël Renault, Kazuhiro Yokoyama

Institutions: University of Le Havre, French Institute for Research in Computer Science and Automation, Rikkyo University

Published on: 28 Jul 2009 - International Symposium on Symbolic and Algebraic Computation

Topics: Splitting field, Symmetric polynomial, Generic polynomial, Separable polynomial and Cyclic permutation

Computation Schemes for Splitting Fields of Polynomials

Sébastien Orange, Guénaël Renault, Kazuhiro Yokoyama

Computation Schemes for Splitting Fields of Polynomials

Sébastien Orange
Académie de Rouen
sebastien.orange@lip6.fr

Guénaël Renault
INRIA, Paris-Rocquencourt,
SALSA Project
UPMC, Univ. Paris 06, LIP6
CNRS, UMR 7606, LIP6
guenael.renault@lip6.fr

Kazuhiro Yokoyama
Rikkyo University
3-34-1 Nishi Ikebukuro,
Toshima-ku
Tokyo 171-8501, Japan
yokoyama@rkmath.rikkyo.ac.jp

ABSTRACT

In this article, we present new results about the computation of a general shape of a triangular basis generating the splitting ideal of an irreducible polynomial given with the permutation representation of its Galois group G. We provide some theoretical results and a new general algorithm based on the study of the non redundant bases of permutation groups. These new results deeply increase the efficiency of the computation of the splitting field of a polynomial.

Categories and Subject Descriptors

I.1 [**Computing Methodologies**]: Symbolic and algebraic manipulations

General Terms

Algorithms, Theory

Keywords

Galois Theory, Triangular Set, Splitting Field

1. INTRODUCTION

The computation of the splitting field of a polynomial f plays an important role in Galois theory and more generally in algorithmic number theory. It is the smallest field where all the roots of f lie. Thus, providing a suitable representation of this field which allow symbolic computations with all the roots of the polynomial is interesting.

Such a representation comes from computer algebra and more precisely from Gröbner basis theory. Let f be a univariate irreducible and separable polynomial of degree n with coefficient in a calculable field $\mathbb{K}$ and $\alpha_1, \ldots, \alpha_n$ its roots in an algebraic closure of $\mathbb{K}$, this natural representation is the quotient algebra

$$\mathbb{K}(\alpha_1, \ldots, \alpha_n) \simeq \mathbb{K}[x_1, \ldots, x_n]/\mathcal{M}$$

where $\mathcal{M}$ is the kernel of the surjective morphism from $\mathbb{K}[x_1, \ldots, x_n]$ to $\mathbb{K}(\alpha_1, \ldots, \alpha_n)$ which maps x_i to α_i. The ideal $\mathcal{M}$, called a **splitting ideal** of f, is zero-dimensional and maximal. Knowing a Gröbner basis of $\mathcal{M}$ allows computations in this quotient algebra by means of linear algebra operations (see e.g. [5, 2]) and then symbolic operations with the roots of f.

In [17, 18], new algorithms are proposed for computing the splitting field of a monic irreducible polynomial f with coefficient in $\mathbb{K} = \mathbb{Q}$ (more generally, these methods can be applied in any global fields). These new algorithms are based on the relationship between the representation of the splitting field by a Gröbner basis and the action of the corresponding Galois group on this basis. The core of this new approach, called *computation scheme* (see section 2), uses the internal symmetries of the problem in order to speed up the Gröbner basis computation. This scheme is computed from the knowledge of a permutation representation of the Galois group G of f and provides a shape of the Gröbner basis of the splitting ideal of f. From this shape, these algorithms effectively compute the basis by interpolating its coefficients (see also [12, 24, 11] and more generally [6] for interpolation strategies). The efficiency of these algorithms heavily depends on this computation scheme which is dependent on the choice of representative of the conjugacy class of G in S_n. Thus, we are interested in finding the representative of G in its conjugacy class which gives the best computation scheme.

In [17] a *brute force* method based on the analysis of all the representatives was proposed in order to collect all the best transitive representatives in a database. But today, the best implementation of the Galois group computation, which was implemented by Fieker and Klüners in Magma [4], does not depend on any database of permutation groups invariants and can be applied for rational coefficients polynomials of any degree. If one wants to do the same with the splitting field computation, one should provide non dependent to database too.

To avoid such a brute force strategy or database during the computation of the splitting field, we first explore the theoretical insight into computation scheme based on relation to families of permutation groups. Then we produce a general algorithm by making good use of this insight. A first non trivial result is given in [16] where an algorithm is described for the computation of the splitting ideal of a polynomial with dihedral Galois. In the same way, one can easily deduce same results for general families like alternating and symmetric groups (see Section 3). We present, in Section 3.2, a more technical result providing a general construction of such a good representation in the case where the group G is a wreath product of transitive permutation groups. This construction can be seen as a *divide and conquer* strategy since the computation takes as input the knowledge of the computation schemes of two different groups with smaller degrees. Since we want a general algorithm for computing the spliting field of a polynomial which can be used for any transitive permutation group, we finally present a new algorithm which provides the best possible computation scheme by considering *non redundant bases of* G (see Section 4).

We prove (see Section 5) that the cost of this algorithm is polynomial in the size of the group. Thus the total cost of the splitting field computation using the algorithms developed in [17, 18] would be now dominated by the algebraic part (Galois group computation and interpolation) and not by the combinatorial part (finding efficient computational scheme). Finally we note that in this paper we present our new results for irreducible polynomials. But one can easily modify them for separable polynomials.

2. DEFINITIONS

In this section, we recall some well known facts of Galois theory and results from [17] related to computation schemes. We present some constructions of computation schemes in the case of simple classes of permutation groups. In all this paper we consider a univariate irreducible monic polynomial f of degree n with coefficients in the integer ring of a global field $\mathbb{K}$ (we can think of $\mathbb{K}$ as $\mathbb{Q}$) and its Galois group G as a transitive permutation group of S_n. The roots of f in an algebraic closure of $\mathbb{K}$ will be denoted by $\alpha_1, \alpha_2, \ldots, \alpha_n$ with a fixed numbering.

We denote by $\mathbb{K}[x_1, \ldots, x_n]$ the ring of multivariate polynomials with coefficients in $\mathbb{K}$. The splitting ideal $\mathcal{M} \subset \mathbb{K}[x_1, \ldots, x_n]$ is defined as the kernel of the surjective morphism from $\mathbb{K}[x_1, \ldots, x_n]$ to $\mathbb{K}(\alpha_1, \ldots, \alpha_n)$ which maps the indeterminate x_i to α_i. This definition depends on the choice of the numbering of the roots α_i.

For the natural group action Ψ of S_n on $\mathbb{K}[x_1, \ldots, x_n]$ which permutes the x_i's by acting on the indexes, the stabilizer of $\mathcal{M}$ is the permutation representation of the Galois group which will be denoted by G:

$$G = \{\sigma \in S_n \,|\, \forall g \in \mathcal{M},\, \sigma.g \in \mathcal{M}\}.$$

PROPOSITION 2.1. *Let σ be a permutation of S_n and let $G^\sigma = \sigma G \sigma^{-1}$. The ideal $\sigma.\mathcal{M} = \{\sigma.f \,|\, f \in \mathcal{M}\}$ is the splitting ideal corresponding to the roots $\sigma.\alpha = (\alpha_{\sigma.1}, \ldots, \alpha_{\sigma.n})$ and the corresponding representation of the Galois group is G^σ (a conjugate of G).*

Thus, if one chooses a permutation representation of the Galois group, one also chooses some of the possible numberings of the roots of f and the corresponding splitting ideal.

Here we start from the knowledge of a permutation representation G of the Galois group and we want to deduce a theoretical form for the Gröbner basis $\mathcal{G}$ of $\mathcal{M}$. A first result coming from classical Galois theory (see for example [24]) shows that this basis is triangular for the lexicographical order induced by $x_1 < \cdots < x_n$ (see [10]), that is $\mathcal{G}$ is given as a set of n polynomials $\{f_1, \ldots, f_n\}$ such that f_i has a power of x_i as leading term and is separable as a polynomial in x_i. Moreover, we can deduce from G the degree d_i of the leading term of each f_i. Let E be a subset of $\{1, \ldots, n\}$, we denote by $\mathrm{Stab}_G(E)$ the pointwise stabilizer in G of E (that is the subgroup of G given by $\{\sigma \in G \mid \sigma(e) = e\, \forall e \in E\}$). We have the following classical result:

$$d_i = |\,\mathrm{Stab}_G(\{1, 2, \ldots, i-1\})|/|\,\mathrm{Stab}_G(\{1, 2, \ldots, i\})|. \quad (2.1)$$

Thus, the degrees d_i of elements in a Gröbner basis (not necessarily reduced but minimal) $\mathcal{G}$ of $\mathcal{M}$ can be deduced only from the known stabilizer G of this ideal.

Computation Scheme: From the knowledge of G we want to know more about the basis $\mathcal{G} = \{f_1, \ldots, f_n\}$. We already know the leading degree d_i of each polynomial f_i, what we present now are techniques that can give possible relations between polynomials in $\mathcal{G}$, more precisely from these relations we will deduce polynomial f_j from f_i with $j > i$. We also present results about the size of the polynomials in $\mathcal{G}$, that is, only from this knowledge, one wants to know the variables and their maximal degree in the f_i's.

The first technique, called **Cauchy technique**, is based on the so called *generalized Cauchy modules* (see [17]):

Definition 1. Let $\mathcal{G} = \{f_1, \ldots f_n\}$ be a triangular basis of $\mathcal{M}$ and $\{i = i_1 < \cdots < i_r\}$ the orbit of i under the action of $\mathrm{Stab}_G(\{1, \ldots, i-1\})$. The d_i generalized Cauchy modules of f_i are inductively defined by $C_{i_1}(f_i) = f_i(x_{i_1})$ and for $k \geqslant 2$ the polynomial $C_{i_1, \ldots, i_k}(f_i)$ is given by the divided difference

$$\frac{C_{i_1, \ldots, i_{k-1}}(f_i)(x_{i_k}) - C_{i_1, \ldots, i_{k-1}}(f_i)(x_{i_{k-1}})}{x_{i_k} - x_{i_{k-1}}}.$$

From these constructions, we can deduce polynomials of $\mathcal{G}$ from other ones. The following result explain this relation.

PROPOSITION 2.2. *The Cauchy module $C_{i_1, \ldots, i_k}(f_i)$ is a polynomial of $\mathbb{K}[x_1, \ldots, x_{i_k}]$ and its leading term is $x_{i_j}^{d_i - k + 1}$. Moreover, $C_{i_1, \ldots, i_k}(f_i)$ belongs to $\mathcal{M}$. In particular, if $d_i - k + 1 = d_{i_k}$ then*

$$\{f_1, \ldots, f_{i_k - 1}, C_{i_1, \ldots, i_k}(f_i), f_{i_k + 1}, \ldots, f_n\}$$

is a triangular basis of $\mathcal{M}$.

Cauchy, in [3, Extrait 108], already proved similar results (without the knowledge of Gröbner basis theory) when he studied the application of Ampère's ¨fonctions interpolaires¨ (what we call now Cauchy modules) for eliminating variables in symmetric funtions.

We now present some results about the shape of the polynomials in $\mathcal{G}$. For better understandings we do a slight change of the definition given in [17]. Let i be an integer in $[\![1, n]\!]$. A sequence r of couples $[(i_1, k_1), (i_2, k_2), \ldots, (i_s, k_s)]$ with $\{i_1 < i_2 < \ldots < i_s = i\}$ a part of $\{1, \ldots, i\}$ and $k_j \leqslant d_{i_j}$ is said to be an ***i*-relation** if there exists a polynomial $g_i \in \mathbb{K}[x_{i_1}, \ldots, x_{i_s}]$ such that $\alpha_i^{k_i + 1} + g_i(\alpha_1, \ldots \alpha_i) = 0$ with $\deg_{x_{i_j}}(g_i) \leqslant k_j$ (note that we must have $k_i = k_{i_s} = d_i - 1$ and $k_j < d_{i_j}$ for $j < s$). The polynomial g_i is called the *the tail* polynomial of this i-relation. While we have just defined i-relation with the roots $\alpha_1, \ldots, \alpha_n$, i-relations depend only of the Galois group G:

PROPOSITION 2.3. *There exists an i-relation $[(i_1, k_1), \ldots, (i_s, k_s)]$ as soon as $\forall j \in [\![1, s]\!]$, $k_j = \frac{|\,\mathrm{Stab}_G(\{i_1, \ldots, i_{j-1}\})|}{|\,\mathrm{Stab}_G(\{i_1, \ldots, i_j\})|}$ and $\frac{|\,\mathrm{Stab}_G(\{i_1, \ldots, i_{s-1}\})|}{|\,\mathrm{Stab}_G(\{i_1, \ldots, i_s\})|} = d_i$.*

For all i we have a trivial i-relation given by $[(1, d_1 - 1), \ldots, (i, d_i - 1)]$ but the corresponding polynomial has a lot of monomials. An important quantity attached to an i-relation is its **size** which corresponds to the product $k_{i_1} \times \cdots \times k_{i_s}$ and represents the maximal number of monomials of the corresponding polynomials f_i. Thus, in order to minimize the cost for the real computation of the triangular basis $\mathcal{G}$ by *indeterminate coefficients strategy* (see [17, 18]), we need to know the best i-relation possible, that is the one with minimal size. Such an i-relation is said to be **minimal**.

Now we will study the natural action of G over the polynomials of $\mathcal{G}$ (permutations of the indexes of the variables) to find relations between these polynomials. These special permutations are named **transporters**.

Definition 2. Let $[(i_1, k_1), \ldots, (i_s, k_s)]$ be an i-relation and $j \in [\![i+1, n]\!]$. A permutation $\sigma \in G$ is called an (i,j)-transporter if $\sigma(i) = j$ and $j = \max(\sigma(k) \,|\, k \in \{i_1, \ldots, i_s\})$.

As for Cauchy technique, transporters can be used to produce polynomials of $\mathcal{G}$ from others taken in $\mathcal{G}$:

PROPOSITION 2.4. *Let σ be an (i,j)-transporter and $g_i \in \mathbb{K}[x_{i_1}, \ldots, x_{i_s}]$ the tail polynomial corresponding to f_i. If $d_i = d_j$ then $\{f_1, \ldots, f_{j-1}, x_j^{d_j} + \sigma.g_i, f_{j+1}, \ldots, f_n\}$ is a triangular basis of $\mathcal{M}$.*

As one can see, all these techniques and the i-relations can be deduced only by inspecting the corresponding permutation group G.

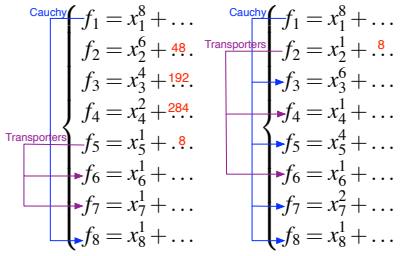
Definition 3. The computation scheme of the permutation group G is defined by the following data:
1. the degree d_i of the greatest variable in each polynomial in $\mathcal{G}$;
2. mathematical objects (shape) computed by Cauchy techniques and transportation ;
3. the minimal i-relation of each polynomial in $\mathcal{G}$ that can not be obtained by the preceding techniques.
The c-size of this computation scheme, denoted by $c(G)$, is defined by the number of monomials over all the i-monomials in 3.

For a given permutation group G, its computation scheme is pre-computed from this definition. At the end of the process, we obtain a scheme of computation for retrieving an actual triangular basis $\mathcal{G}$ of a splitting ideal with Galois group G. Thus, the c-size $c(G)$ represents the total number of coefficients to compute in order to retrieve this triangular basis by interpolation. For different conjugates of a same group G, their respective computation schemes may have different c-sizes.

EXAMPLE 1. *Let G_1 and G_2 be two copies in S_8 of the transitive permutation group $[2^4]S_4$ given by*

$$\begin{aligned} G_1 &= \langle (8,7,6,1)(5,4,3,2), (8,1)(4,5), (5,1) \rangle \\ G_2 &= \langle (2,1), (8,6,4,1)(7,5,3,2), (8,1)(7,2) \rangle \end{aligned}$$

The two corresponding schemes can be represented by the following drawings (G_1 on the left and G_2 on the right).



On these drawings, the techniques are showed on the left side of the triangular bases and the integers, on left side, represents the sizes of the minimal i-relations. Thus, for G_1 we have $c(G_1) = 532$ and for G_2 we obtain only $c(G_2) = 8$.

Thus, giving the permutation group G as the Galois group of f, in order to compute the splitting field of f efficiently, we first find the conjugate of G which gives the computation scheme with the smallest c-size, this is our principal aim here.

3. COMPUTATION SCHEMES OF SOME GROUPS FAMILIES

In this section we present our first results about the construction of computation schemes for families of permutation groups. The first sub section presents already known results and the second one presents a new result for the construction of computation scheme when G is a wreath product.

3.1 First examples

Symmetric and Alternate Groups: the proposition 2.2 gives a first technique for finding relations between the polynomials of $\mathcal{G}$. As a first example, we can easily deduce the Gröbner basis $\mathcal{M}$ when G is the symmetric group. Actually, in this case, we have $f_1 = f(x_1)$ and $f_i = C_{1,\ldots,i}(f_1)$, thus the Gröbner basis $\mathcal{G}$ can be deduced from f (see [19], for example, where this ideal is introduced for resolvent computations). The same result can be given for alternate groups (except that polynomial f_{n-1} must be computed).

Cyclic groups: Assume that the Galois group of a polynomial $f \in k[x]$ of degree n is the cyclic group. Up to a numbering of the roots of f, its Galois group can be identified to the subgroup $G = \langle \sigma := (1, 2, \ldots, n) \rangle$ of S_n. From G we can deduce the degrees d_i of polynomials f_i thus a first theoretical form for the Gröbner basis is deduced: $\mathcal{G} = \{f_1 = f(x_1), f_2 = x_2 + g_2(x_1), \ldots, f_n = x_n + g_n(x_1, \ldots, x_{n-1})\}$. This description can be detailed by using results about i-relations: $\mathcal{G} = \{f_1 = f(x_1), f_2 = x_2 + g_2(x_1), \ldots, f_n = x_n + g_n(x_1)\}$. It can easily be proved that for all $i \in [\![2, n-1]\!]$, the permutation σ is a $(i, i+1)$-transporter. Therefore, $\mathcal{M}$ is generated by $\{f_1 = f(x_1), f_2 = x_2 + g_2(x_1), \sigma.f_2 = x_3 + g_2(x_2), \ldots, \sigma^{n-1}.f_2 = x_n + g_2(x_{n-1})\}$. So, for polynomial with cyclic Galois group, the computation of the generating set $\mathcal{G}$ of a splitting ideal can be reduced to the one of the single polynomial f_2.

Dihedral groups: From [16] we can easily deduce a computation scheme for dihedral groups. In this case $f_1 = f$ and f_2 is a degree two polynomial in x_2 and it depends on the variable x_1, the remaining polynomials in $\mathcal{G}$ will be linear polynomials in their greatest variable. In this case, from the computation of the polynomial f_2 we can deduce a linear relation by Cauchy technique then we can deduce the remaining ones by transporters techniques.

All these examples take the specific particularity of each of these families to produce computation schemes. The next subsection can be viewed as a *divide and conquer* strategy since we study the computation scheme of a family of groups which are construct from two known ones: wreath product.

3.2 Computation scheme of wreath products

In this section we give one of our new result: the computation scheme of a particular family of groups which are constructed from two smaller groups, this study can be seen as a divide and conquer strategy. It is based on the study of the intrinsic blocks action of permutation groups coming

from wreath-products. Wreath-products are already well studied in Galois theory since they are intensively used for computing subfield (see [9, 8] for examples). First, we recall the definition of a wreath product (see [13] for more details) then we will present the result and sketch its proof.

Definition 4. Let m and m' be two positive integers. Let $H < S_m$ and $\tilde{K} < S'_m$ two permutations groups and $\Omega = \{1,\dots,m\} \times \{1,\dots,m'\}$. The *wreath product* $H \wr \tilde{K}$ is the permutation group of Ω generated by the groups
- $\bar{H} = H \times \cdots \times H$ (direct product of m' copies of H) acting on Ω by setting, for all $(h_1,\dots,h_{m'}) \in \bar{H}$ and, for all $(u,v) \in \Omega$, $(h_1,\dots,h_{m'})(u,v) = (h_v(u),v)$;
- K isomorphic to $\tilde{K}$ acting on Ω by the rule $\forall t \in K,\ \forall (u,v) \in \Omega,\ t(u,v) = (u,t(v))$.

When G is isomorphic to a wreath product, we have to choose a suitable symmetric representation of G to take benefits of Cauchy and transporters techniques in order to obtain an efficient computation scheme. Such a representation is given by the choice of the bijection from Ω to $\{1,\dots,n\}$ which gives a representation of G in S_n. By setting the following notations, we fix this choice.
- m and m' are two integers such that $n = mm'$;
- $\tilde{K}$ (resp. H) is a transitive subgroup of $S_{m'}$ (resp. S_m identified to the subgroup of S_n acting on $\{1,\dots,m\}$);
- φ is the bijection from Ω to $\{1,\dots,n\}$ which maps (u,v) to $(v-1)m+u$;
- G is the image in S_n induced by φ of $H \wr \tilde{K}$.

Now, from these notations we state the main result which shows that, if G is the Galois group of f, a triangular Gröbner basis $\mathcal{G} = \{f_1, f_2, \dots, f_n\}$ of $\mathcal{M}$ can be deduced only from $m+m'-2$ polynomials of $\mathcal{G}$ by transporters and generalized Cauchy modules techniques.

THEOREM 3.1. *Let f a polynomial with Galois group G. There exists a triangular Gröbner basis $\mathcal{G} = \mathcal{B}_1 \cup \ldots \cup \mathcal{B}_{m'}$ of $\mathcal{M}$ such that*

$$\mathcal{B}_1 = \begin{cases} f_1 = f(x_1) \\ f_2 = x_2^{d_2} + g_2(x_1,x_2) \\ \vdots \\ f_m = x_m^{d_m} + g_m(x_1,\dots,x_m) \end{cases}$$

and, for all $q \in [\![2, m'-1]\!]$,

$$\mathcal{B}_q = \begin{cases} f_{qm+1} = x_{qm+1}^{d_{qm+1}} + g_{qm+1}(x_1,\dots,x_{qm+1}) \\ f_{qm+2} = x_{qm+2}^{d_2} + g_2(x_{qm+1},x_{qm+2}) \\ \vdots \\ f_{qm+m} = x_{qm+m}^{d_m} + g_m(x_{qm+1},\dots,x_{qm+m}) \end{cases}$$

Moreover, the degree d_i of f_i in x_i is given by

$$d_i = \begin{cases} m\frac{|\operatorname{Stab}_{\tilde{K}}(\{1,\dots,q-1\})|}{|\operatorname{Stab}_{\tilde{K}}(\{1,\dots,q\})|} & \text{if } r = 1 \\ \frac{|\operatorname{Stab}_H(\{1,\dots,r-1\})|}{|\operatorname{Stab}_H(\{1,\dots,r\})|} & \text{otherwise} \end{cases}$$

where $(r,q) = \varphi^{-1}(i)$.

PROOF. Let's consider a triangular basis $\{f_1, f_2, \dots, f_n\}$ of a relations ideal $\mathcal{M}$ of f and set, for all $i \in [\![1,n]\!]$, $f_i = x_i^{d_i} + g_i(x_1,\dots,x_{i-1})$.
Assertion about the degrees of f_i in x_i. Let i be an integer of $\{1,\dots,n\}$ and let's set $(r,q) = \varphi^{-1}(i)$. Since G is the wreath product $H \wr \tilde{K}$, we have a bijection from $\operatorname{Stab}_G(\{1,\dots,i\})$ onto the cartesian product $\operatorname{Stab}_K(\{1,\dots,i\}) \times \operatorname{Stab}_{\bar{H}}(\{1,\dots,i\})$.
Denotes $(r',q') = \varphi^{-1}(i-1)$. Equality (2.1) gives then the degree d_i of f_i in x_i :

$$d_i = \underbrace{\frac{|\operatorname{Stab}_{\tilde{K}}(\{1,\dots,q'\})|}{|\operatorname{Stab}_{\tilde{K}}(\{1,\dots,q\})|}}_{\delta_1} \times \underbrace{\frac{|\operatorname{Stab}_{\bar{H}}(\{1,\dots,i-1\})|}{|\operatorname{Stab}_{\bar{H}}(\{1,\dots,i\})|}}_{\delta_2}.$$

Two cases appear:
1) $r = 1$. In this first case, we have $\delta_1 = \frac{|\operatorname{Stab}_{\tilde{K}}(\{1,\dots,q-1\})|}{|\operatorname{Stab}_{\tilde{K}}(\{1,\dots,q\})|}$ and $\delta_2 = |H||\operatorname{Stab}_H(\{1\})|^{-1} = m$ since h is transitive. This proves the first equality of the assertion.
2) $r \neq 1$. In this second case, we have $\delta_2 = \frac{|\operatorname{Stab}_H(\{1,\dots,r-1\})|}{|\operatorname{Stab}_H(\{1,\dots,r\})|}$ and $\delta_1 = 1$. This shows the second equality of the assertion.
Assertion about the Gröbner basis $\mathcal{G}$. Let's fix $q \in [\![2, m'-1]\!]$. For all $r \in [\![2,m]\!]$, the previous result shows that the leading term of f_r is $x_r^{d_r}$. Since K is transitive, there exists a transporter $\sigma \in K$ which maps each $r \in [\![2,m]\!]$ on $qm+r$, thus $f_r^\sigma = x_i^{d_r} + g_r(x_{qm+1},\dots,x_{qm+r})$. The assertion about the Gröbner basis $\mathcal{G}$ is then a consequence of Proposition 2.4. □

REMARK 2. *The Cauchy technique can be used to produce polynomials of $\mathcal{M}$ each time Proposition 2.2 can be applied. For example, equality (2.1) implies that the degree of the leading term of f_n (resp. $f_{m(m'-1)+1}$) is 1 (resp. m). Since $\deg_{x_1}(f_1) = n$, f_n (resp. $f_{m(m'-1)+1}$) can be replaced by $C_{1,\dots,n}(f_1)$ (resp. $C_{1,\dots,m(m'-1)+1}(f_1)$).*

4. FAST COMPUTATION OF COMPUTATION SCHEME

In this section we present our new algorithm for computing a conjugate of a given transitive permutation group with the smallest computation scheme. For a better reading, we present the different functions in a sequential programming style even that the central structure, the orbits tree, is recursive.

4.1 Proof of concept and first definitions

As we already explain in Section 1, a trivial brute force method is proposed in [17] for computing a conjugate of a permutation group with minimal computation scheme in the sense of the c-size. In this method, the number of candidates considered is equal to the index $|S_n : N_{S_n}(G)|$ which is can be closed to $(n-1)!$ when G is small (i.e. for the cyclic group of degree n, this index is equal to $\frac{(n-1)!}{\phi(n)}$). Thus the brute force method is completely useless in this case. In section 3, we showed how to easily construct a computation scheme for particular types of permutation group. Here we give an algorithm which can be applied for any type of permutation group. We present it in its most general form, some tricks can be used to increase its efficiency but they heavily depend on the type of the given group.

The concept of our algorithm is based on the correspondence between the sequence of pointwise stabilizer subgroups orbits of G and the sequence of factors of a polynomial f with Galois group G in the tower of subfields arising during the computation of its splitting field (see [14, 15]). More precisely, when we construct the splitting field of f by successive factorization (see [22, 23] and [1] for the algorithmic point of view) at the first step we factorize the polynomial f over the algebraic extension K_1 of $\mathbb{K}$ obtained by adjoining

any one root of $f_1 = f$ that we denote α_1; at the second step, we *choose* f_2 among the computed irreducible factors different from $x - \alpha_1$ and then we consider the factorization of f over the algebraic extension of K_1 obtained by adjoining one root of f_2 and we continue until we define f_n. The corresponding triangular ideal is generated by the set of polynomials $\mathcal{G} = \{f_1, f_2, \ldots, f_n\}$ but, during the construction we may have chosen f_2 to be a linear factor of f over K_1 different from $x - \alpha_1$ and f_3 to be a non linear factor of f over $K_1 = K_2$. In this case, if we exchange the two polynomials f_2 and f_3 in the set $\mathcal{G}$ we will obtain two different triangular ideals corresponding to two different conjugates of G and, in terms of the size of their computation scheme, it is easy to see that they may be not equivalent (for example, if the non linear factor is a degree $n-2$ polynomial, we could apply the Cauchy technique when f_3 corresponds to this non linear factor and we could not for the other choice). In fact, the two computations schemes techniques and minimal i-relations depend only on the structure of the different orbits of the pointwise stabilizers of G which correspond to the different factors arising during the computation of the splitting field by successive factorization. To be clear, let sketch some Galois theory. The construction of the splitting field by successive factorizations corresponds to the construction of a chain of stabilizer for G: At the first step we choose one element e_1 of the unique orbit of G (as it is transitive) and we construct the stabilizer $\mathrm{Stab}_G(\{e_1\})$ which will correspond to the Galois group of the polynomial f over the extension K_1. At the second step we choose one element e_2 of the orbit of $\mathrm{Stab}_G(\{e_1\})$ corresponding to the set of roots of f_2 then we construct $\mathrm{Stab}_G(\{e_1, e_2\})$ and so on.

Thus we do not need to inspect all the conjugates of a group but all the different possible sequences of orbits appearing during the process of stabilization of the group G. Then, from all these possibilities we well order the choice of the orbits to obtain the best conjugate of G in regards of its computation scheme. In the sequel, we will show how to compute this representation efficiently. All these possibilities corresponds to a set of different classes of *non redundant bases* of G:

Definition 5. Let G be a permutation group of degree n. A sequence $B = (b_1, \ldots, b_k)$ of different integers from $\{1, \ldots, n\}$ is called a **regular sequence of length** k. A regular sequence B is said to be **non redundant** with respect to G if

$$G = G_B^{[1]} > G_B^{[2]} > \cdots > G_B^{[k+1]},$$

where, for an easier reading, we denote by $G_B^{[i]}$ $(i \geqslant 2)$ the pointwise stabilizer $\mathrm{Stab}_G(\{b_1, \ldots, b_{i-1}\})$ in G. Moreover, for a non redundant regular sequence B, if $G_B^{[k+1]} = 1$, B is said to be a **non redundant base** of G. The largest k such that there is a non redundant base of length k is called the **depth** of G.

Now we introduce an equivalence relation over the set of non redundant bases of G.

Definition 6. Let $B_1 = (a_1, a_2, \ldots, a_l)$ and $B_2 = (b_1, b_2, \ldots, b_l)$ be two non redundant bases of G of the same length. We say that B_1 is G**-equivalent** to B_2 if there exists $g \in G$ such that $B_2 = (g(a_1), g(a_2), \ldots, g(a_l))$

Clearly, the G-equivalence property is an equivalent relation over the set of non redundant bases. In a field theory point of view, two non redundant bases are G-equivalent iff the towers of fields defined by these bases are isomomorphic *level by level* from the ground field $\mathbb{K}$ to the splitting field. From now on, when we will speak about classes of non redundant bases they will be always classes of G-equivalence.

To a base B for G corresponds a generating set S_B for G which verifies

$$\langle S_B \cap G_B^{[i]} \rangle = G_B^{[i]}, \text{ for } 1 \leqslant i \leqslant k+1$$

and is named *Strong Generating Set* (see [20, Chapter 4]). From now on, we assume that the group G is generated by such a basis and suppose that the corresponding *Schreier tree* is constructed too. We can bound the size of S_B by $\mathcal{O}(\log(|G|)$ (see [21]). Theses two objects are useful in algorithmic group theory since they provide a data structure for the group G which lets us compute a lot of operations on this group efficiently.

As shown before, to a non redundant base B of G corresponds different triangular ideals with different computation schemes. In the section 4.3 we will show how to order the polynomials in the triangular set $\mathcal{G}$ to provide the best computation scheme corresponding to the base B. For the moment, we give some theoretical results about the number of classes of non redundant bases of a given group G.

First we provide some additional notations and properties. Let $\mathcal{C}$ be the set of all classes of non-redundant bases, and from each class, one is chosen in our search. So, let $\mathcal{B}$ be the set of representatives of classes, that is, $\mathcal{C} = \{B^G \mid B \in \mathcal{B}\}$. We write d for the depth of G. Then, as there is a non redundant base B of length d, thus, since each B in $\mathcal{B}$ is non redundant, we have $d \leq \min(n, \log_2(|G|))$ but here, the group has a moderate size (not more than 10000) thus we can assume that $\log_2(|G|) < n$. We note that, since $|G_B^{[i]} : G_B^{[i+1]}| \geq 2$ for each $i \leq d$ and $G_B^{[d+1]} = 1$, we obtain $|G| = |G : G_B^{[d+1]}| \geq 2^d$ and $d \leq \log_2(|G|)$ (see [20]).

For each regular sequence B, we write B^G for its G orbit, where G acts naturally on $B = (i_1, \ldots, i_k)$ by $g(B) = (g(i_1), \ldots, g(i_k))$ for $g \in G$. So, for a non redundant base B, B^G coincides with the G-equivalent class $\{g(B) \mid g \in G\}$ of non redundant bases containing B.

Lemma 4.1. *We have*

$$\#\mathcal{B} \leq n(n-1)\cdots(n-d+1)|G|^{-1} \leqslant |S_n : G|.$$

Moreover, as $d \leq \log_2(|G|)$,

$$\#\mathcal{B} < n^{\log_2(|G|)}|G|^{-1} = |G|^{\log_2(n)-1}.$$

Proof. Here we give a brief proof. In order to estimate the size $\#\mathcal{B}$, we consider another set $\hat{\mathcal{B}}$ consisting of regular sequences of length d constructed as follows: For each B in $\mathcal{B}$, if the length of B is smaller than d, we extend B to a regular sequence $\hat{B}$ of length d by padding certain integers in $\{1, 2 \ldots, n\}$. Otherwise, that is, if the length B coincides with d, we set $\hat{B} = B$. Then, it follows that for distinct B_1 and B_2 in $\mathcal{B}$, $\hat{B}_1^G$ does not intersect $\hat{B}_2^G$, as B_1^G does not intersect B_2^G.

Now we count all the regular sequences in $\cup_{\hat{B} \in \hat{\mathcal{B}}} \hat{B}$. We remark that the number of all regular sequences of length d is $n(n-1)\cdots(n-d+1)$.

Since $|\hat{B}^G| = |G : Stab_G(\hat{B})|$ and $Stab_G(\hat{B}) \leqslant Stab_G(B) = G_B^{[d+1]} = 1$, we have $\sum_{\hat{B} \in \hat{\mathcal{B}}} |\hat{B}^G| = \sum_{B \in \mathcal{B}} |G| \leqslant n(n-$

$1)\cdots(n-d+1)$. Thus, we obtain $\#\mathcal{B} \leqslant \frac{n(n-1)\cdots(n-d+1)}{|G|} \leqslant |S_n : G|$. Also, from this inequality and the fact $d \leqslant \log_2(|G|)$, we have $\#\mathcal{B} \leq \frac{n^d}{|G|} = |G|^{\log_2(n)-1}$. □

EXAMPLE 3. *Here we show easy examples. For the cyclic group C_n, $d = 1$ and $\#\mathcal{B} = \frac{n}{|C_n|} = 1$. For the dihedral group D_n, $d = 2$ and $\#\mathcal{B} = \frac{n(n-1)}{|D_n|} = \frac{n-1}{2}$. For the symmetric group S_n, $d = n$ and $\#\mathcal{B} = \frac{n!}{|S_n|} = 1$.*

4.2 Orbits tree of G

We now introduce a data structure attached to the permutation group G which let us store all its non redundant bases and the corresponding orbits of the natural action of its stabilizer along these different bases. From now on we say that an orbit is **non trivial** when it is not reduced to one element.

Definition 7. The **orbit tree** $\mathcal{T}$ of G is the recursive structure defined by
1. The root $\mathcal{T}(0)$ of $\mathcal{T}$ is the orbit G: $\{1, \ldots, n\}$;
2. any other node $\mathcal{T}(i_1 = 1, i_2, \ldots, i_s)$ is the set of orbits of the pointwise stabilizer $\mathrm{Stab}_G(\{0, i_1, \ldots, i_s\})$ where i_s is the minimal element of a non trivial orbit in the node $\mathcal{T}(i_1 = 1, i_2, \ldots, i_{s-1})$;
3. the construction is stopped as soon as the node contains only trivial orbits.
Let $T(i_1, \ldots, i_s)$ be a node in $\mathcal{T}$. The **degree** of this node is the integer defined by the index $|G : \mathrm{Stab}_G(\{0, i_1, \ldots, i_s\})|$.

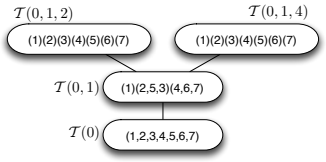
Clearly, we can retrieve all the non redundant bases of G up to the G-equivalence as the sequences defining the leafs of $\mathcal{T}$. Thus, there is a bijective correspondence from the branches of $\mathcal{T}$ to the set $\mathcal{B}$. By using classical orbit computation and Lemma 5 we deduce the following result.

PROPOSITION 4.2. *Let $\#\mathcal{B}$ be the number of non equivalent non redundant bases of G and d its depth. The time complexity for constructing $\mathcal{T}$ is*

$$\mathcal{O}(\#\mathcal{B}dn|S|) = \mathcal{O}(\#\mathcal{B}dn\log(|G|)) = \mathcal{O}(n|G|^{\log_2 n}\log_2^2(|G|)$$

We now use this structure in order to efficiently compute the conjugate of G with minimal computation scheme.

EXAMPLE 4. *Let G be a copy in S_7 of the transitive group $F_{21}(7)$ generated by $\{(1,2,3,4,5,6,7), (1,2,4)(3,6,5)\}$. The following drawing corresponds to its orbits tree.*



4.3 From a Branch to a Computation Scheme

In this subsection we describe the central part of our algorithm which takes as input a branch of the orbits tree of G and returns the best conjugates of G, in terms of c-size, with equivalent stabilizers sequence. In this part we suppose the orbits tree $\mathcal{T}$ of G accessible. We first present the **Application of the Cauchy technique.** Let $f_j \in \mathcal{G}$ be a generalized Cauchy module of degree d_j of a polynomial $f_i \in \mathcal{G}$ of degree d_i. Then, from Definition 1 and Proposition 2.2 we know that the set of roots S_j of f_j are included in the set of roots S_i of f_i (seen as univariate polynomials over the extensions defined respectively by the polynomials in $\mathcal{G}$ with indexes less than j and i) and the relative complement $S_i \backslash S_j$ corresponds to roots defined by polynomials with indexes less than j. Thus, in a stabilizer point of view, this means that the orbit $\mathcal{O}$ of the action of $\mathrm{Stab}_G(\{1, \ldots, i-1\})$ corresponding to the polynomial f_i splits in $d_i - d_j$ orbits of length 1 and one orbit of length d_j under the action of $G_{\{1,\ldots,j-1\}}$. This let us give the next result for ordering the polynomials in $\mathcal{G}$ to obtain the best gain of the Cauchy technique.

PROPOSITION 4.3. *Let B be a base of G. A triangular basis $\mathcal{G}$ corresponding to B with the best gain of the Cauchy technique is obtained by ordering the polynomials following the principles:*
1. Each time a new trivial orbit appears in the branch of $\mathcal{T}$ corresponding to B we add the corresponding linear polynomial in $\mathcal{G}$;
2. The non linear polynomial corresponding to an element of B is added into $\mathcal{G}$ after we added all the preceding linear polynomials of the principle 1;
3. The correspondence between the element of the orbits and the indexes of the variables are stored in order to obtain at the end of the process the conjugate of G stabilizing $\mathcal{G}$.

We can immediately apply this result to the construction of a first shape for the triangular basis $\mathcal{G}$. Such a shape constructed following the principles of the proposition 4.3 will be said to be in **Cauchy shape**. As we state before, after the application of proposition 4.3 we are in the best position for the application of the Cauchy techniques. It only remains to checks if possible Cauchy techniques can be applied by using the knowledge of the orbits tree. We can summarize this first step into the following function.

Function `CauchyBranch`(B)
Input: $B = [b_1, \ldots, b_k]$ a nonredundant base of G
Output: A Cauchy shape of a triangular basis corresponding to B and its stabilizer.
  $i = 1$; $\mathcal{G}[i] = [(1, n)]$
  $\sigma[1] = b_1$ (The permutation σ is represented as a sequence)
  For $j = 1$ to $k - 1$ **do**
    For $\mathcal{O}$ in $\mathcal{T}(b_1, \ldots, b_j) \setminus \mathcal{T}(b_1, \ldots, b_{j-1})$ **do**
      The elements in $\mathcal{O}$ are now depending of $(b_j, |\mathcal{O}_k| - 1)$ $(*)$
      If $\mathcal{O}$ is a trivial orbit $\{e\}$ **then**
        $i = i + 1$; $\mathcal{G}[i] = [(i, 1)]$ cat Dep($\mathcal{O}$); $\sigma[i] = e$
        If Cauchy technique apply **then** mark $\mathcal{G}[i]$ **end if**
      end if
    end for
    Let $\mathcal{O}_{j+1}$ be the orbit in $\mathcal{T}(b_1, \ldots, b_j)$ containing b_{j+1}
    $i = i + 1$; $\mathcal{G}[i] = [(i, |\mathcal{O}_{j+1}|)]$ cat Dep($\mathcal{O}_{j+1}$); $\sigma[i] = b_j$
    If the Cauchy technique apply **then** mark $\mathcal{G}[i]$ **end if**
  end for
  Return $\mathcal{G}$, $\sigma G \sigma^{-1}$, σ

In the function `CauchyBranch`, we use the function `Dep` which takes as input an orbit $\mathcal{O}$ and returns the sequence of dependent couples attached to each of its elements. This list is generated on the line marked by $(*)$ and one of its elements (i, d_i) represents the index i of a variable and its maximal degree d_i appearing in the i-relation corresponding to $\mathcal{O}$. At the end, the function returns the shape of the triangular basis $\mathcal{G}$ as a sequence of i-relations. Thus, the function `CauchyBranch` already provides a way to construct sparse i-relations, but may be not the sparsest ones.

One can easily see that all the computations done during the process of the `CauchyBranch` for a base B can be introduced during the construction of the branches of the orbits tree corresponding to B. Thus, the complexity of this function is not so important.

Application of the Transporter technique. We now study the possibility of finding transporters from the knowledge of a Cauchy shape of a triangular basis $\mathcal{G}$, thus we assume that we already have the three outputs $\mathcal{G}$, G',σ of `CauchyBranch`(B). All the polynomials non marked in $\mathcal{G}$ have tail only depending of variables indexed, after σ^{-1} action on theses indexes, by elements of the unredundant base B. Thus, finding a potential transporter from a polynomial f_i to a polynomial f_j in $\mathcal{G}$ can be done by analyzing the natural action of G. More precisely, finding this transporter can be done by checking the existence of an element $g \in G$ such that B_i^g is included in $\{1,\ldots,j\}^{\sigma^{-1}}$ where B_i is the subset of B corresponding to the tail of f_i and $\sigma g \sigma^{-1}(i) = j$.

Since an element g of G is uniquely determined by the images by g of the elements of B, checking the existence of a transporter in G can be done by *sifting* procedure (see [20, Chapter 4]) using the Schreier tree associated to B. We not give here the detail of this technical procedure which is a generalization of the one given in the proof of [20, Lemma 5.2.1] which presents an algorithm constructing, if there exists one, a permutation $g \in G$ with a given image B^g. From the same lemma we can deduce the next result.

PROPOSITION 4.4. *Let $\mathcal{G}$ be a Cauchy shape corresponding to a nonredundant base B of B. Checking the existence and, if there exists, computing an (i,j)-transporter for $\mathcal{G}$ can be realized in $\mathcal{O}(nt|G_B^{[m]}|)$ time where t is the sum of the depths of the Schreier tree related to B and m is the greatest integer verifying $\sigma^{-1}(b_m) \leqslant i$.*

The integer t can be bounded by $2d\log(|G|)$ where d is the depth of G. Moreover, we can store the successive computations during the construction of the transporters for a better efficiency. When we theoretically add all these time complexities over all the computations of transporters for $\mathcal{G}$ we obtain a time complexity depending of $|G|$ which corresponds to a *brute force* procedure and which is very pessimistic in comparison with the practical efficiency.

Finding the minimal i-relations. For a polynomial shape f_i of degree d_i in its greater variable x_i in $\mathcal{G}$ that is not marked after application of Cauchy or transporter techniques, we can easily find a minimal i-relation corresponding to this polynomial by inspecting some node of the orbits tree $\mathcal{T}$. Actually, we consider the nodes $\mathcal{T}$ of degree less than the size of f_i and we check for the existence of an orbit containing $\sigma^{-1}(i)$ of cardinal d_i inside T. If such a node exists, we obtain a new i-relation with a smaller size than the former. By repeating this operation over all these nodes we can find the minimal i-relation $\mathcal{G}$.

Function `MinimalRelation`(i,σ)
Input: The index of the i-relation and the permutation σ which carry the element of B in the index of the polynomials of $\mathcal{G}$
Output: A minimal i-relation for the triangular basis $\mathcal{G}$.

```
Let r be the i-relation in G
Let s be the size of r and d its degree
For B in all the branches of T do
 For T in successive nodes of B do
  If the degree of T is less than s then
   If ∃O ∈ T such that σ^{-1}(i) ∈ O and |O| = d then
    Form the new i-relation r from O
    Change the value s to the degree of T.
   end if
  else break end if
 end for
end for
Return r
```

These function could be modified in order to find the minimal relation for each non marked polynomial in $\mathcal{G}$ at the same time. We can benefit of the recursive structure of the orbits tree in order to efficiently traverse it with recursive functions. Even in this case, the time complexity estimation would be not tight but it gives some theoretical point of views about our new algorithm:

PROPOSITION 4.5. *Let $\#\mathcal{B}$ be the number of branches in $\mathcal{T}$ and d the depth of the group G. The time complexity for finding all the minimal i-relations in $\mathcal{G}$ is $\mathcal{O}(\#\mathcal{B}nd)$.*

REMARK 5. *As bounds in Lemma are for general cases, and do not seem sharp. Actually, for each group with moderate size, the number $\#\mathcal{B}$ is much smaller, which implies certain efficiency of our computation. Also, for the complexity estimation, we have no chance but to use the bound in Lemma . But, for splitting field computation, algebraic parts are dominant and we can improve the total efficiency with smaller computation on group theoretical part.*

5. THEORETICAL AND PRACTICAL CONCLUSIONS

In this section, we conclude by presenting the total theoretical cost for computing the conjugate of G giving the best computation scheme. We also give tricks to avoid the inspection of some branches of $\mathcal{T}$ and we present some practical results of our implementation done in MAGMA (see [4]) (this implementation could also benefit of all the functionalities of GAP (see [7]) the well known open source CAS).

5.1 Total theoretical cost

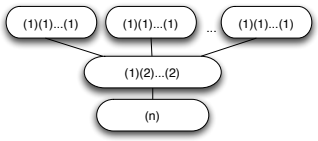
From the results of Section 4, we can compute the conjugate of the transitive permutation group G with minimal computation scheme by searching the best branch of the orbits tree $\mathcal{G}$. Thus, the total cost of our algorithm is dominated by the construction of $\mathcal{T}$ and by the application of the transporter technique which are, for a fixed degree n, dominated by a polynomial complexity in $|G|$. In comparison with the method proposed in [17] for the computation of computation scheme, which has a tight cost depending on $|S_n : N_{S_n}(G)|$, this new algorithm is really more efficient. Hence, this new algorithm is more efficient when the Galois group G not too large which is a natural assumption when we want to compute the splitting field of a polynomial.

5.2 Tricks to avoid branches

During the construction of the orbits tree $\mathcal{T}$, some nodes can be detected in advance to be not necessary continued:
1. In the node T, all the non trivial orbits are all of the same cardinal and one of its sons contains only trivial orbits.
2. Each non trivial orbit in the node T are used to defined a node in a sub branch starting from T.
In these two cases, we not need to construct the complete subtree starting from T, only one branch is necessary. The first case is clear: all the sons of T will give nonredundant bases with same properties in regards of the computation scheme. The second is coming from the fact that the non

trivial orbits in T correspond to extensions over the field defined by the node T so that they have not common subfield. To take account of this property, we need to make a slight modification in the function `CauchyBranch` in order to produce the best Cauchy shape possible by considering only one branch from T.

EXAMPLE 6. *The case* 1 *appears when G is a dihedral group. Assume that n is odd, from section 3, we know that $\mathcal{T}$ would have the following form (the cardinals of orbits are indicated between parenthesis).*



In this case, we only have to study one branch since they are all equivalent in a computation scheme point of view.

5.3 Practical results and final conclusion

Usually, when we present practical results of our implementations we draw some graphics or give some experimentations times tables. In the present case, it would be not so interesting since we try our implementation (on a MacBookPro 2.16GHz with MAGMA 32 bits ver. 2.14) over all the transitive groups of degree up to **25** and order up to **10 000** and we obtain an average time less than **1.0** seconds : the different steps take almost the same time and the difference between timings for different groups is not significant. As a final conclusion, we can say that our first aim which was to provide an efficient way to produce the splitting field of a polynomial without any database is reached. Actually, we can use this new algorithm as a link between the one implemented by Fieker and Klüners in MAGMA, for the computation of the action of the Galois group over approximations of the roots of f, and the ones of [17, 18] for computing the splitting field from these inputs.

6. REFERENCES

[1] ANAI, H., NORO, M. AND YOKOYAMA, K. Computation of the splitting fields and the Galois groups of polynomials. In *Algorithms in algebraic geometry and applications (Santander, 1994)*, vol. 143 of *Progr. Math.* Birkhäuser, Basel, 1996, pp. 29–50.

[2] BECKER, T. AND WEISPFENNING, V. *Gröbner bases*, vol. 141 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1993. A computational approach to commutative algebra, In cooperation with Heinz Kredel.

[3] CAUCHY, A. *Œuvres complètes*, Gauthier-Villars (Paris), Série 1, tome 5, 1882-1974. Gallica distribution.

[4] BOSMA, W., CANNON, J., AND PLAYOUST, C. The Magma algebra system. I. The user language. *J. Symbolic Comput. 24*, 3-4 (1997), 235–265.

[5] COX, D., LITTLE, J. AND O'SHEA, D. *Ideals, varieties, and algorithms*, second ed. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1997.

[6] DAHAN, X., AND SCHOST, É. Sharp estimates for triangular sets. In *ISSAC '04: Proc. of the 2004 International Symposium on Symbolic and Algebraic Computation* (New York, 2004), ACM Press, pp. 103–110.

[7] THE GAP GROUP, *Groups, Algorithms, and Programming, Ver. 4.4.12*, 2008. `http://www.gap-system.org`

[8] GEISSLER K. AND KLÜNERS, J. Galois group computation for rational polynomials. *J. Symbolic Comput.*, 30(6):653–674, 2000.

[9] HULPKE, A., *Block systems of a Galois group*. Experiment. Math., Vol. 4, 1995, nb. 1, pp 1–9.

[10] LAZARD, D. Solving zero-dimensional algebraic systems. *J. Symbolic Comput. 13*, 2 (1992), 117–131.

[11] LEDERER, M., Explicit constructions in splitting fields of polynomials. *Riv. Mat. Univ. Parma (7)*, 3* (2004), 233–244.

[12] MCKAY, J. AND STAUDUHAR, R. Finding relations among the roots of an irreducible polynomial. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (Kihei, HI)* (New York, 1997), ACM, pp. 75–77 (electronic).

[13] MELDRUM, J. D. P. *Wreath products of groups and semigroups* Pitman Monographs and Surveys in Pure and Applied Mathematics, vol. 74. Longman, 1995. ISBN 0-582-02693-8.

[14] ORANGE, S. Calcul de corps de décomposition - Utilisation fine d'ensembles de permutation en thorie de Galois effective - *PhD Thesis, LIP6, University Paris 6, 2006.*

[15] ORANGE, S. AND RENAULT, G. AND VALIBOUZE, A., Calcul efficace de corps de décomposition. LIP6 Research Report 005, Laboratoire d'Informatique de Paris 6, 2003.

[16] RENAULT, G. Computation of the Splitting Field of a Dihedral Polynomial. In *Proc. of the 2006 International Symposium on Symbolic and Algebraic Computation (Genova, Italy)* (New York, 2006). ACM Press.

[17] RENAULT, G. AND YOKOYAMA, K. A modular method for computing the splitting field of a polynomial. In *Proc. of the 7th Algorithmic Number Theory Symposium* ANTS-VII, Berlin, Germany, 2006, LNCS 4076, Springer.

[18] RENAULT, G. AND YOKOYAMA, K. A multi-modular algorithm for computing the splitting field of a polynomial. In *Proc. of the 2008 International Symposium on Symbolic and Algebraic Computation (Linz, Austria)* (New York, 2008). ACM Press.

[19] RENNERT, N. AND VALIBOUZE, A. Calcul de résolvantes avec les modules de Cauchy. *Exp. Math.*, 8(4):351–366, 1999.

[20] SERESS, Á. *Permutation group algorithms. Cambridge Tracts in Mathematics*, vol 152 Cambridge University, 2003 ISBN 0-521-66103-X

[21] SIMS, C. C., Computation with permutation groups, In *SYMSAC '71: Proceedings of the second ACM symposium on Symbolic and algebraic manipulation*, 1971, pp 23–28.

[22] TCHEBOTAREV, N. *Gründzüge des Galois'shen Theorie*. P. Noordhoff, 1950.

[23] TRAGER, B. Algebraic factoring and rational functiun integration. In *Proceedings of SYMSAC'76* (1976), pp. 219–226.

[24] YOKOYAMA, K. A modular method for computing the Galois groups of polynomials. *J. Pure Appl. Algebra 117/118* (1997), 617–636. Algorithms for algebra (Eindhoven, 1996).