# Evaluation properties of symmetric polynomials

Pierrick Gaudry, Eric Schost, Nicolas Thiéry

# Evaluation properties of symmetric polynomials

Pierrick Gaudry, LIX, École polytechnique
91128 Palaiseau, France
gaudry@lix.polytechnique.fr

Éric Schost, LIX, École polytechnique
91128 Palaiseau, France
schost@lix.polytechnique.fr

Nicolas M. Thiéry,
Laboratoire de Probabilités, Combinatoire et Statistiques,
Université Claude Bernard Lyon I, France
Laboratoire de Mathématiques,
Université Paris Sud, France
nthiery@users.sourceforge.net

March 23, 2005

**Abstract**

By the fundamental theorem of symmetric polynomials, if $P \in \mathbb{Q}[X_1, \ldots, X_n]$ is symmetric, then it can be written $P = Q(\sigma_1, \ldots, \sigma_n)$, where $\sigma_1, \ldots, \sigma_n$ are the elementary symmetric polynomials in $n$ variables, and $Q$ is in $\mathbb{Q}[S_1, \ldots, S_n]$.

We investigate the complexity properties of this construction in the straight-line program model, showing that the complexity of evaluation of $Q$ depends only on $n$ and on the complexity of evaluation of $P$.

Similar results are given for the decomposition of a general polynomial in a basis of $\mathbb{Q}[X_1, \ldots, X_n]$ seen as a module over the ring of symmetric polynomials, as well as for the computation of the Reynolds operator.

## 1 Introduction

Already known to Newton, the fundamental theorem of symmetric polynomials asserts that any symmetric polynomial is a polynomial in the elementary symmetric polynomials. To be more precise, let us define the symmetric polynomials $\overline{\sigma}_1, \ldots, \overline{\sigma}_n$ by letting $\overline{\sigma}_i$ be the coefficient of $T^{n-i}$ in the polynomial $(T - X_1) \cdots (T - X_n)$; that is, $\overline{\sigma}_i = (-1)^i \sigma_i$, where $\sigma_1, \ldots, \sigma_n$ are the usual elementary symmetric polynomials (this sign convention happens to simplify some of the subsequent developments).

Then if $P \in \mathbb{Q}[X_1, \ldots, X_n]$ is a symmetric polynomial in $n$ variables, it is known that there exists a unique polynomial $Q \in \mathbb{Q}[S_1, \ldots, S_n]$ such that the equality $P =$

$Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$ holds. For this point, as well as for other questions related to symmetric polynomials, our general references will be [12, 15].

From the complexity viewpoint, one may wonder what properties pass from $P$ to $Q$. For instance, the (weighted) degree is preserved. On the other hand, important features such as sparseness are lost: Consider $P = X_1^d + X_2^d \in \mathbb{Q}[X_1, X_2]$, and the polynomial $Q$ such that $P = Q(\overline{\sigma}_1, \overline{\sigma}_2)$ with $\overline{\sigma}_1 = -X_1 - X_2, \overline{\sigma}_2 = X_1 X_2$; then the number of monomials of $Q$ is linear in $d$.

This phenomenon is intimately related to the basic approach on symmetric polynomials by means of rewriting techniques. Indeed, the classical proof of the fundamental theorem involves an explicit rewriting process for a suitable elimination order [18, 5], and such techniques do not preserve sparseness.

In this note, we adopt a different point of view, working in the *straight-line program* model. Roughly speaking, a straight-line program is a sequence of basic instructions $(+, -, \times)$ that are used to compute a given polynomial; the relevant complexity measure of such an object is its *size*, *i.e.* the number of its instructions (see Subsection 2.1 for definitions). The *complexity of evaluation* of a polynomial $P$ is then the minimum size of a straight-line program that computes $P$.

Straight-line programs have proved to be an appropriate data-structure to derive complexity estimates in polynomial elimination theory (see references below). One of the salient results is that the complexity of evaluation remains stable throughout elimination processes: eliminating polynomials (e.g., Chow forms) that are obtained from polynomials with a low complexity of evaluation also have a low complexity of evaluation. This is the key to the algorithms with the best known complexity for polynomial system solving.

Our main goal is to present results in a similar vein for computations with symmetric polynomials: If $P$ is a symmetric polynomial in $\mathbb{Q}[X_1, \ldots, X_n]$ with a good complexity of evaluation, and $Q$ is such that $P = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$, then $Q$ itself has a good complexity of evaluation. The precise form of this result is given below. The statement involves a quantity denoted by $\Delta(n)$, which will be defined in Subsection 3.1 as the complexity of multiplication in a suitable algebra; for the moment, we can content ourselves with the estimate $\Delta(n) \leq 4^n (n!)^2$.

**Theorem 1.** *Let $P$ in $\mathbb{Q}[X_1, \ldots, X_n]$ be a symmetric polynomial that can be computed by a straight-line program of size $L$. Let $Q$ be the unique polynomial in $\mathbb{Q}[S_1, \ldots, S_n]$ such that $P = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$. Then $Q$ can be computed by a straight-line program of size $\Delta(n)L + 2$, with $\Delta(n) \leq 4^n (n!)^2$.*

Note that the degree of $P$ does not appear in this estimate: passing from $P$ to $Q$, the complexity of evaluation increases by a factor that only depends on $n$. As an application, consider again the polynomials $P = X_1^d + X_2^d$ and $Q$ such that $P = Q(-X_1 - X_2, X_1 X_2)$. Using binary powering techniques, $P$ can be computed by a straight-line program of size $O(\log(d))$. Theorem 1 then shows that this is also the case for $Q$; this should be compared with the number of monomials of $Q$, which is linear in $d$.

Our interest for this topic originates from [8], where a problem of solving some polynomial systems with symmetries is raised (a more general version of that question was already discussed in [4]). To solve that particular problem, the above theorem suffices. However, the proof techniques easily give more general results.

Let us write $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$ for the algebra of symmetric polynomials. Then the polynomial ring $\mathbb{Q}[X_1, \ldots, X_n]$ becomes a free module of rank $n!$ over $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$. Thus, a first generalization is to determine the coordinates of any polynomial $P$ in a basis of this free module. The proof of Theorem 1 readily gives this generalization for a standard monomial basis, but other bases are of interest, such as, for instance, the Schubert basis. Such bases have cardinality $n!$ and are thus commonly indexed by the permutations in the $n$-th symmetric group $\mathfrak{S}_n$; we will use this indexation below. Besides, the techniques can be generalized to other families of algebra generators for $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$, such as the complete symmetric polynomials and power sums.

We obtain results that generalize those of Theorem 1: roughly speaking, the complexity of evaluation only grows by the factor $\Delta(n)$, up to an additional factor that depends on the chosen bases. To give a precise statement, fix $n \geq 1$, and consider a family $\mathfrak{b} = (b_1, \ldots, b_n)$ of $\mathbb{Q}$-algebra generators of $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$ and a basis $\mathfrak{c} = (c_s)_{s \in \mathfrak{S}_n}$ of the $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$-module $\mathbb{Q}[X_1, \ldots, X_n]$. The most general form of Theorem 1 involves some constants depending on $\mathfrak{b}$ and $\mathfrak{c}$, denoted by $L(\mathfrak{b})$ and $L(\mathfrak{c})$.

**Theorem 2.** *Let $n \geq 1$, $\mathfrak{b}$ and $\mathfrak{c}$ be as above. Then there exists $L(\mathfrak{b})$ and $L(\mathfrak{c})$ in $\mathbb{N}$ with the following property: Let $P$ be a polynomial in $\mathbb{Q}[X_1, \ldots, X_n]$ and let $(P_s)_{s \in \mathfrak{S}_n}$ be the unique polynomials in $\mathbb{Q}[B_1, \ldots, B_n]$ such that*

$$P = \sum_{s \in \mathfrak{S}_n} P_s(b_1, \ldots, b_n) c_s \ .$$

*If $P$ can be computed by a straight-line program of size $L$, then there exists a straight-line program of size $\Delta(n)L + L(\mathfrak{b}) + L(\mathfrak{c}) + 2$ which computes all the polynomials $P_s$, with $\Delta(n) \leq 4^n (n!)^2$.*

Theorem 1 is actually a particular case of this result, when $P$ is symmetric, $b_1, \ldots, b_n$ are the symmetric polynomials $\overline{\sigma}_1, \ldots, \overline{\sigma}_n$, and $\mathfrak{c}$ is the standard monomial basis; in this case we have $L(\mathfrak{b}) = L(\mathfrak{c}) = 0$. Note that we could incorporate the term $+2$ that appears in the estimate of the theorem in either $L(\mathfrak{b})$ or $L(\mathfrak{c})$, but this would conflict with this last statement.

Our last question of interest is the computation of the Reynolds operator, which we treat as an application of the previous results. The Reynolds map $R$ is a projector $\mathbb{Q}[X_1, \ldots, X_n] \to \mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$, so for any $P$ in $\mathbb{Q}[X_1, \ldots, X_n]$, there exists $Q$ in $\mathbb{Q}[S_1, \ldots, S_n]$ such that $R(P) = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$ (other choices of algebra generators for $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$ would do as well, of course). Based on our previous results, the last theorem shows that if $P$ can be computed in time $L$, then $R(P)$ can be computed in time $\Delta(n)L$, up to about $n!$ additional operations.

**Theorem 3.** *Let $n \geq 1$, and $P$ in $\mathbb{Q}[X_1, \ldots, X_n]$ that can be computed by a straight-line program of size $L$. Let $Q$ be the unique polynomial in $\mathbb{Q}[S_1, \ldots, S_n]$ such that $R(P) = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$. Then $Q$ can be computed by a straight-line program of size $\Delta(n)L + n! + 2 \cdot 8^n$, with $\Delta(n) \leq 4^n (n!)^2$.*

In the above theorems, we used $\mathbb{Q}$ as base field for simplicity; we mention however that all results extend to any base ring of characteristic zero; all results that involve neither the Reynolds operator nor the symmetric power sums actually extend to any base ring.

**Related work.** Many techniques used in this paper, notably the so-called algebra of universal decomposition and the related Cauchy modules, were already used in the study of symmetric polynomials. Explicitly, the idea of obtaining the expression of a symmetric polynomial in terms of the elementary symmetric ones by reduction modulo what we call Cauchy modules is already present in [6, 16], and is discussed in details (without using the same denomination) in [7], together with some generalizations to other groups.

However, none of the above references mentions complexity. Our contribution is a first exploration of the complexity-related aspects of this question, showing that evaluation techniques give an appropriate computational model for handling symmetric polynomials.

It turns out that our basic algorithms are somehow relevant from polynomial elimination. Then, the fact that evaluation techniques are the key to good complexity results supports ideas initiated by Giusti, Heintz, Pardo and collaborators in [11, 10, 9], who showed that, generally speaking, straight-line programs are an appropriate data-structure for algorithms in effective elimination theory.

Of course, we expect that our results generalize to finite groups actions, even though several closed form formulas (e.g., explicit descriptions of the Cauchy modules) that are available here have probably no equivalent in the more general case. Then, we might have to rely on effective elimination theory tools.

**Optimal bounds.** At the moment, we do not know whether the factor $\Delta(n)$, which grows polynomially with $n!$, is optimal. To put it more precisely, let us write $L(A)$ for the minimal size of a straight-line program that computes a polynomial $A$. Let then $\delta(n)$ be the supremum of the ratios $L(Q)/L(P)$, where $P$ runs through the symmetric polynomials in $\mathbb{Q}[X_1, \ldots, X_n]$ and $Q$ is such that $P = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$. Theorem 1 shows that $\delta(n) \leq \Delta(n) \in (n!)^{O(1)}$; an open question is to give a non-trivial lower bound for $\delta(n)$.

**Organization of the paper.** In Section 2, we define our computational model, give the details of our construction on the example $P = X_1^d + X_2^d$, and recall some classical facts about symmetric polynomials. In Section 3, we give the proofs of Theorems 1, 2 and 3.

# 2 Preliminaries

## 2.1 Straight-line programs

From an informal point of view, straight-line programs enable us to represent polynomials by means of a sequence of operations $(+, -, \times)$ without test nor division. Formally, let $k$ be a field and $L \geq 0, n \geq 1$. Following [3], we define a straight-line program $\Gamma$ in $k[X_1, \ldots, X_n]$ as a sequence of polynomials $G_{-n+1}, \ldots, G_L$ in $k[X_1, \ldots, X_n]$. For $-n+1 \leq i \leq 0$, we take $G_i = X_{i+n}$; for $i > 0$, suppose that $G_{-n+1}, \ldots, G_{i-1}$ are defined. Then, we require that one of the following holds:

- $G_i = \lambda$, with $\lambda \in k$.
- $G_i = \lambda + G_{a_i}$, $G_i = \lambda - G_{a_i}$ or $G_i = \lambda G_{a_i}$, with in any case $\lambda \in k$ and $-n+1 \leq a_i < i$.

- $G_i = G_{a_i} + G_{b_i}$, $G_i = G_{a_i} - G_{b_i}$ or $G_i = G_{a_i} G_{b_i}$, with in any case $-n+1 \leq a_i, b_i < i$.

In this situation, we say that $\Gamma$ *computes* $G_{-n+1}, \ldots, G_L$ and has *size* $L$. If $F_1, \ldots, F_m$ are polynomials in $k[X_1, \ldots, X_n]$, then we say that $F_1, \ldots, F_m$ can be *computed by a straight-line program of size* $L$ (or *in time* $L$) if there exists a straight-line program that computes polynomials $G_{-n+1}, \ldots, G_L$ such that $\{F_1, \ldots, F_m\}$ is included in the set $\{G_{-n+1}, \ldots, G_L\}$.

The following lemma gives a basic property of the straight-line model, which is useful in the sequel.

**Lemma 1 (Composition of straight-line programs).** *Let* $\mathbf{a} = (a_1, \ldots, a_n)$ *be polynomials in* $k[X_1, \ldots, X_m]$ *and let* $\mathbf{b} = (b_1, \ldots, b_m)$ *be polynomials in* $k[Y_1, \ldots, Y_s]$. *Suppose that* $\mathbf{a}$ *can be computed by a straight-line program* $\Gamma_{\mathbf{a}}$ *of size* $L_{\mathbf{a}}$, *and* $\mathbf{b}$ *by a straight-line program* $\Gamma_{\mathbf{b}}$ *of size* $L_{\mathbf{b}}$.

*Then there exists a straight-line program* $\Gamma$ *of size* $L_{\mathbf{a}} + L_{\mathbf{b}}$ *that computes the same polynomials as* $\Gamma_{\mathbf{b}}$, *as well as the polynomials*

$$a_1(b_1, \ldots, b_m), \ldots, a_n(b_1, \ldots, b_m).$$

*Proof.* Let $G_{-m+1}, \ldots, G_{L_{\mathbf{a}}}$ (resp. $H_{-s+1}, \ldots, H_{L_{\mathbf{b}}}$) be the polynomials computed by $\Gamma_{\mathbf{a}}$ (resp. $\Gamma_{\mathbf{b}}$). For $i = 1, \ldots, L_{\mathbf{a}}$, define $K_i = G_i(b_1, \ldots, b_m)$. Then the sequence of polynomials $H_{-s+1}, \ldots, H_{L_{\mathbf{b}}}, K_1, \ldots, K_{L_{\mathbf{a}}}$ satisfies our requirement. □

## 2.2 A detailed example

We now show the use of the straight-line program representation for handling symmetric polynomials, by computing the symmetrized form of the polynomial $P = X_1^8 + X_2^8 \in \mathbb{Q}[X_1, X_2]$. Let us thus consider a sequence of instructions that computes $P$:

$$G_1 = X_1^2; \quad G_2 = G_1^2; \quad G_3 = G_2^2; \quad H_1 = X_2^2; \quad H_2 = H_1^2; \quad H_3 = H_2^2;$$

so that $P = G_3 + H_3 = X_1^8 + X_2^8$. We now show how to compute the unique polynomial $Q \in \mathbb{Q}[S_1, S_2]$ such that $P = Q(-X_1 - X_2, X_1 X_2)$.

Let us introduce two new indeterminates $S_1$ and $S_2$ and the ideal $I$ generated by $S_1 - (-X_1 - X_2)$ and $S_2 - X_1 X_2$ in $\mathbb{Q}[S_1, S_2][X_1, X_2]$. Our strategy is to compute the coordinates of the polynomials $G_i$ and $H_i$ in the $\mathbb{Q}[S_1, S_2]$-algebra $K = \mathbb{Q}[S_1, S_2][X_1, X_2]/I$. From this, we will recover the polynomial $Q$.

The monomials $(1, X_1)$ form a basis of $K$ as a $\mathbb{Q}[S_1, S_2]$-algebra and the relation $X_1^2 + S_1 X_1 + S_2 = 0$ holds in $K$. We deduce that for all $A_0, A_1, B_0, B_1$ in $\mathbb{Q}[S_1, S_2]$, multiplication in $K$ is given by the following rule:

$$(A_0 + A_1 X_1)(B_0 + B_1 X_1) = (A_0 B_0 - S_2 A_1 B_1) + (A_1 B_0 + A_0 B_1 - S_1 A_1 B_1) X_1 \ .$$

This multiplication can be written using the following straight-line program $\Gamma$, which uses Karatsuba's trick to lower the number of multiplications:

$$\Gamma \left| \begin{array}{llll} V_1 = A_0 B_0; & V_2 = A_1 B_1; & V_3 = A_0 + A_1; & V_4 = B_0 + B_1; \\ V_5 = V_3 V_4; & V_6 = V_5 - V_1; & V_7 = V_6 - V_2; & V_8 = S_2 V_2; \\ V_9 = -S_1 V_2; & V_{10} = V_1 - V_8; & V_{11} = V_7 + V_9. & \end{array} \right.$$

Then $V_{10}$ and $V_{11}$ are respectively the polynomials $(A_0B_0 - S_2A_1B_1)$ and $(A_1B_0 + A_0B_1 - S_1A_1B_1)$; note that $\Gamma$ performs 11 operations.

For $i = 1, 2, 3$, let us write $G_i \bmod I = G_{i,0} + G_{i,1}X_1$, with $G_{i,0}$ and $G_{i,1}$ in $\mathbb{Q}[S_1, S_2]$. Using $\Gamma$, we first design a straight-line program that computes the polynomials $G_{i,0}$ and $G_{i,1}$, for $i = 1, 2, 3$. To this effect, let us take $G_{0,0} = 0$ and $G_{0,1} = 1$, so that $G_{0,0} + G_{0,1}X_1 = X_1$. Since $G_1 = X_1^2$, we can adapt $\Gamma$ to obtain $G_{1,0}$ and $G_{1,1}$:

$$\begin{array}{llll} V_{1,1} = 0; & V_{1,2} = 1; & V_{1,3} = 1; & V_{1,4} = 1; \\ V_{1,5} = V_{1,3}V_{1,4}; & V_{1,6} = V_{1,5} - V_{1,1}; & V_{1,7} = V_{1,6} - V_{1,2}; & V_{1,8} = S_2V_{1,2}; \\ V_{1,9} = -S_1V_{1,2}; & G_{1,0} = V_{1,1} - V_{1,8}; & G_{1,1} = V_{1,7} + V_{1,9}. & \end{array}$$

Iterating the process, we obtain $G_{2,0}, G_{2,1}$ and $G_{3,0}, G_{3,1}$ in a similar fashion.

$$\begin{array}{llll} V_{2,1} = G_{1,0}^2; & V_{2,2} = G_{1,1}^2; & V_{2,3} = G_{1,0} + G_{1,1}; & V_{2,4} = G_{1,0} + G_{1,1}; \\ V_{2,5} = V_{2,3}V_{2,4}; & V_{2,6} = V_{2,5} - V_{2,1}; & V_{2,7} = V_{2,6} - V_{2,2}; & V_{2,8} = S_2V_{2,2}; \\ V_{2,9} = -S_1V_{2,2}; & G_{2,0} = V_{2,1} - V_{2,8}; & G_{2,1} = V_{2,7} + V_{2,9}; & \end{array}$$

$$\begin{array}{llll} V_{3,1} = G_{2,0}^2; & V_{3,2} = G_{2,1}^2; & V_{3,3} = G_{2,0} + G_{2,1}; & V_{3,4} = G_{2,0} + G_{2,1}; \\ V_{3,5} = V_{3,3}V_{3,4}; & V_{3,6} = V_{3,5} - V_{3,1}; & V_{3,7} = V_{3,6} - V_{3,2}; & V_{3,8} = S_2V_{3,2}; \\ V_{3,9} = -S_1V_{3,2}; & G_{3,0} = V_{3,1} - V_{3,8}; & G_{3,1} = V_{3,7} + V_{3,9}. & \end{array}$$

We are now almost done: we have obtained polynomials $G_{3,0}$ and $G_{3,1}$ in $\mathbb{Q}[S_1, S_2]$ such that $G_3 = X_1^8 = G_{3,0} + G_{3,1}X_1$ holds modulo $I$. Starting from $X_2 = -S_1 - X_1$, we can use the same techniques to obtain polynomials $H_{3,0}$ and $H_{3,1}$ such that $H_3 = X_2^8 = H_{3,0} + H_{3,1}X_1$ holds modulo $I$. The sum $G_3 + H_3$ being symmetric, it equals $G_{3,0} + H_{3,0}$ modulo $I$, so that $G_{3,0} + H_{3,0}$ is the polynomial $Q$ we are looking for.

Computing $Q$ requires $2 \times 3 \times 11 + 3 = 69$ operations $(+, -, \times)$. Had we considered the polynomial $X_1^{16} + X_2^{16}$ instead, the cost would be 91, due to an additional squaring in $K$. Similarly, treating the polynomial $X_1^{2^k} + X_2^{2^k}$ would require $22k + 3$ instructions. In particular, in view of Waring's formula, this shows that given $X_1 + X_2$ and $X_1X_2$, one can evaluate the sum

$$X_1^d + X_2^d = \sum_{j=0}^{\lfloor d/2 \rfloor} (-1)^j \frac{d}{d-j} \binom{n-j}{j} (X_1X_2)^j (X_1 + X_2)^{d-2j}$$

within $O(\log(d))$ arithmetic operations, whereas the number of terms in the right hand side is linear in $d$.

Section 3 provides with a generalization of this process. Subsection 3.3 shows how to further save a constant factor (here, that would be 2), using the Reynolds operator.

## 2.3 Symmetric polynomials and the algebra of universal decomposition

We gather here a few classical facts about symmetric functions that will be used throughout the rest of this paper, and refer to [12] for a systematic exposition. Let $R$ be a ring, and consider the ring of symmetric polynomials in $R[X_1, \ldots, X_n]$. Aside from the family of elementary symmetric polynomials which we have already introduced, we will make

use of the family $h_i, i \geq 0$ of *complete symmetric polynomials*, and the family $p_i, i > 0$ of *symmetric power sums* which are respectively defined by

$$h_i = \sum_{\alpha_1+\cdots+\alpha_n=i} X_1^{\alpha_1} \cdots X_n^{\alpha_n} \quad \text{and} \quad p_i = \sum_{j=1}^{n} X_j^i \ .$$

We have mentioned that the family $\overline{\sigma} = (\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$ freely generates $R[X_1, \ldots, X_n]^{\mathfrak{S}_n}$ as an $R$-algebra. The same property holds for the families $\mathbf{h} = (h_1, \ldots, h_n)$, and, if $R$ contains $\mathbb{Q}$, $\mathbf{p} = (p_1, \ldots, p_n)$.

These families of symmetric functions can be conveniently encoded via their respective generating series in $R[X_1, \ldots, X_n][[z]]$:

$$\begin{aligned} S(z) &= \sum_{i\geq 0} \overline{\sigma}_i z^i = (1 - zX_1) \cdots (1 - zX_n) \ , \\ H(z) &= \sum_{i\geq 0} h_i z^i = \frac{1}{1 - zX_1} \cdots \frac{1}{1 - zX_n} \ , \\ P(z) &= \sum_{i\geq 1} \frac{p_i}{i} z^i \ . \end{aligned}$$

These generating series satisfy the relations

$$S(z) = \frac{1}{H(z)} \quad \text{and} \quad S(z) = \exp(-P(z)) \ .$$

Exceptionally, the *Schur symmetric polynomials* do not play any special role in the theory developed here. Indeed, they do not form a multiplicative basis, and are thus not well suited for evaluation techniques. On the other hand, we will make a quick use of the monomial symmetric functions, which are directly related to the Reynolds operator; for a partition $\lambda$, the *monomial symmetric function* $m_\lambda$ is defined as the sum of all the monomials in the orbit of $X_1^{\lambda_1} \ldots X_n^{\lambda_n}$.

### 2.3.1 Divided difference operators

Fix $i$ in $1, \ldots, n-1$, and let $\tau_i$ be the elementary transposition, acting naturally on $R[X_1, \ldots, X_n]$ by permuting the variables $X_i$ and $X_{i+1}$. The $i$-th *divided difference operator* $\partial_i$ maps a polynomial $f \in R[X_1, \ldots, X_n]$ onto $\partial_i f = (f - \tau_i f)/(X_i - X_{i+1}) \in R[X_1, \ldots, X_n]$. These operators play a fundamental role in the theory of symmetric functions [13, 12].

**Lemma 2.** *For any $i$ and $d \geq 1$ we have:*

$$\partial_i h_d(X_1, \ldots, X_i) = \begin{cases} 0 & \text{if } d = 0, \\ h_{d-1}(X_1, \ldots, X_{i+1}) & \text{otherwise.} \end{cases}$$

*Proof.* It is easiest to prove the result for all $d$ at once by means of generating series. To this end, we let the divided difference operator $\partial_i$ act naturally on $R[X_1, \ldots, X_n][[z]]$, in

which all the steps of the following computation occur:

$$\sum_{d\geq 0}\partial_i z^d h_d(X_1,\dots,X_i)=\partial_i\sum_{d\geq 0}z^d h_d(X_1,\dots,X_i)=\partial_i\frac{1}{1-zX_1}\cdots\frac{1}{1-zX_i}$$
$$=\frac{1}{1-zX_1}\cdots\frac{1}{1-zX_{i-1}}\partial_i\frac{1}{1-zX_i}=\frac{1}{1-zX_1}\cdots\frac{1}{1-zX_{i-1}}\frac{\frac{1}{1-zX_{i+1}}-\frac{1}{1-zX_i}}{X_{i+1}-X_i}$$
$$=\frac{1}{1-zX_1}\cdots\frac{1}{1-zX_{i-1}}\frac{z}{(1-zX_i)(1-zX_{i+1})}=z\sum_{d\geq 0}z^d h_d(X_1,\dots,X_{i+1})\ .$$

Equating the coefficients of $z$ on both sides of this equation finishes the proof. □

### 2.3.2 The algebra of universal decomposition

To make effective the $\mathbb{Q}[X_1,\dots,X_n]^{\mathfrak{S}_n}$-algebra structure on $\mathbb{Q}[X_1,\dots,X_n]$, we consider two sets of indeterminates $S_1,\dots,S_n$ and $X_1,\dots,X_n$, and work in the polynomial ring $\mathbb{Q}[S_1,\dots,S_n][X_1,\dots,X_n]$, taking $\mathbb{Q}[S_1,\dots,S_n]$ for base ring. We then introduce the following polynomials in $\mathbb{Q}[S_1,\dots,S_n][X_1,\dots,X_n]$:

$$F_i\ : S_i-\overline{\sigma}_i(X_1,\dots,X_n),\quad i=1,\dots,n\ .$$

Let $I$ be the ideal $(F_1,\dots,F_n)\subset\mathbb{Q}[S_1,\dots,S_n][X_1,\dots,X_n]$. Through the isomorphism $\mathbb{Q}[S_1,\dots,S_n]\simeq\mathbb{Q}[X_1,\dots,X_n]^{\mathfrak{S}_n}$, the $\mathbb{Q}[S_1,\dots,S_n]$-algebra $\mathbb{Q}[S_1,\dots,S_n][X_1,\dots,X_n]/I$ is isomorphic to the $\mathbb{Q}[X_1,\dots,X_n]^{\mathfrak{S}_n}$-algebra $\mathbb{Q}[X_1,\dots,X_n]$; it is called the *algebra of universal decomposition* in [1] (see also [6]).

The divided difference operators can be used to define an alternative family of polynomials $(T_1,\dots,T_n)$ in $\mathbb{Q}[S_1,\dots,S_n][X_1,\dots,X_n]$ that will turn out to generate $I$ as well, but are better suited for computations. We first set

$$T_1(X_1)=X_1^n+S_1X_1^{n-1}+\cdots+S_{n-1}X_1+S_n\ ,$$

and then define inductively $T_2,\dots,T_n$ by

$$T_{i+1}(X_1,\dots,X_{i+1})=\partial_i T_i(X_1,\dots,X_i),\quad 1\leq i<n\ .$$

These polynomials are sometimes called the *Cauchy modules*, see [6, 16]. For the sake of selfcontainedness we include here a proof of the following well known fact, that expresses these polynomials in terms of the complete functions.

**Lemma 3.** *The polynomial $T_i$, $1\leq i\leq n$, equals*

$$T_i=\sum_{k=0}^{n+1-i}S_k h_{n+1-i-k}(X_1,\dots,X_i)\ ,$$

*where $S_0$ is set to 1 for convenience. It follows in particular that $T_i$ is monic of degree $n-i+1$ in $X_i$, and has coefficients in $\{1,S_1,\dots,S_i\}$ with respect to the variables $X_1,\dots,X_i$.*

*Proof.* By definition, $T_1$ satisfies the desired formula. Using induction on $i \geq 1$, we get that:

$$T_{i+1} = \partial_i T_i = \partial_i \sum_{k=0}^{n+1-i} S_k h_{n+1-i-k}(X_1, \ldots, X_i) = \sum_{k=0}^{n+1-i} S_k \partial_i h_{n+1-i-k}(X_1, \ldots, X_i) \ .$$

Applying Lemma 2 yields, as desired, that:

$$T_{i+1} = \sum_{k=0}^{n+1-(i+1)} S_k h_{n+1-(i+1)-k}(X_1, \ldots, X_{i+1}) \ .$$

□

In what follows, we denote by $E$ the set of multi-indices

$$E = \{\alpha = (\alpha_1, \ldots, \alpha_n) \mid 0 \leq \alpha_i < n - i + 1, \ 1 \leq i \leq n\} \ .$$

We will call the set of monomials $X^E = \{X^\alpha = X_1^{\alpha_1} \cdots X_n^{\alpha_n} \mid \alpha = (\alpha_1, \ldots, \alpha_n) \in E\}$ the *standard monomial basis*. Let $K$ be the quotient $\mathbb{Q}[S_1, \ldots, S_n][X_1, \ldots, X_n]/(T_1, \ldots, T_n)$. The previous lemma implies in particular that these monomials form a basis of $K$ as a free $\mathbb{Q}[S_1, \ldots, S_n]$-algebra.

The following classical lemma is the basic tool for the main results of this article; it shows that $K$ coincides with the algebra of universal decomposition defined above. The proof is mostly taken from [18, Theorem 1.2.7] and [6, Theorem 6].

**Lemma 4.** *The ideals generated in $\mathbb{Q}[S_1, \ldots, S_n][X_1, \ldots, X_n]$ by $T_1, \ldots, T_n$ and $F_1, \ldots, F_n$ coincide.*

*Proof.* We first prove that $(T_1, \ldots, T_n) \subset (F_1, \ldots, F_n)$. Fix $i$ in $1, \ldots, n$. Taking the coefficient of degree $n+1-i$ on both sides of the following equation on generating series:

$$\left(\sum_{d \geq 0} \overline{\sigma}_d(X_1, \ldots, X_n) z^d\right) \left(\sum_{d \geq 0} h_d(X_1, \ldots, X_i) z^d\right) = \frac{(1 - zX_1) \cdots (1 - zX_n)}{(1 - zX_1) \cdots (1 - zX_i)}$$
$$= (1 - zX_{i+1}) \ldots (1 - zX_n) \ ,$$

yields the following classical identity on symmetric polynomials:

$$\sum_{k=0}^{n+1-i} \overline{\sigma}_k(X_1, \ldots, X_n) h_{n+1-i-k}(X_1, \ldots, X_i) = 0 \ .$$

From Lemma 3, it follows that $T_i = \sum_{k=0}^{n+1-i} S_k h_{n+1-i-k}(X_1, \ldots, X_i)$ is in $(F_1, \ldots, F_n)$.

To prove the other inclusion, let $X$ be a new indeterminate and $f$ be the polynomial

$$f = X^n + S_1 X^{n-1} + \cdots + S_n \in K[X]. \tag{1}$$

In what follows, for $i = 1, \ldots, n$, we write $T_i(X_1, \ldots, X_{i-1}, X)$ for the polynomial $T_i$ with $X_i$ evaluated at $X$, so that $f = T_1(X)$. By construction, the following equality

$$T_i(X_1, \ldots, X_{i-1}, X) = (X - X_i) T_{i+1}(X_1, \ldots, X_i, X) + T_i$$

holds in $\mathbb{Q}[S_1, \ldots, S_n][X_1, \ldots, X_n][X]$ for $i = 1, \ldots, n-1$. Furthermore, using Lemma 3, we have for $i = n$ the equality $T_n(X_1, \ldots, X_{n-1}, X) = (X - X_n) + T_n$. Moduling out $(T_1, \ldots, T_n)$, we deduce the factorization

$$f = (X - X_1) \cdots (X - X_n) \in K[X]. \tag{2}$$

Equating the coefficients of $X$ in Equations (1) and (2) shows that all polynomials $F_i$ are zero in $K$, finishing the proof. □

**Corollary 1.** *Let $P$ be in $\mathbb{Q}[X_1, \ldots, X_n]$ and $(P_\alpha)_{\alpha \in E}$ be the unique polynomials in $\mathbb{Q}[S_1, \ldots, S_n]$ such that*

$$P = \sum_{\alpha \in E} P_\alpha X_1^{\alpha_1} \cdots X_n^{\alpha_n} \bmod (T_1, \ldots, T_n)$$

*holds in $\mathbb{Q}[S_1, \ldots, S_n][X_1, \ldots, X_n]$. Then the polynomials $(P_\alpha)_{\alpha \in E}$ are the unique polynomials satisfying the following equality:*

$$P = \sum_{\alpha \in E} P_\alpha(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) X_1^{\alpha_1} \cdots X_n^{\alpha_n}.$$

Following [18], one could actually prove that in $\mathbb{Q}[S_1, \ldots, S_n, X_1, \ldots, X_n]$ the polynomials $T_1, \ldots, T_n$ form a Gröbner basis for the ideal $(F_1, \ldots, F_n)$ with respect to any elimination order with $X_n > \cdots > X_1 > S_1, \ldots, S_n$. However, we shall not need this result here.

# 3 Proof of the main results

The key result of this section is the following proposition, which contains Theorem 1 as a special case when $P$ is symmetric, and is the basis to Theorems 2 and 3. The statement involves the quantity $\Delta(n)$, which will denote the complexity of multiplication in the algebra of universal decomposition; the rest of the notation is that of the previous section.

**Proposition 1.** *Let $P$ be in $\mathbb{Q}[X_1, \ldots, X_n]$ and let $(P_\alpha)_{\alpha \in E}$ be the unique polynomials in $\mathbb{Q}[S_1, \ldots, S_n]$ such that the equality*

$$P = \sum_{\alpha \in E} P_\alpha(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) X_1^{\alpha_1} \cdots X_n^{\alpha_n} \tag{3}$$

*holds. Suppose that $P$ can be computed by a straight-line program of size $L$. Then there exists a straight-line program of size $\Delta(n)L + 2$ which computes all polynomials $(P_\alpha)_{\alpha \in E}$, with $\Delta(n) \leq 4^n (n!)^2$.*

In Subsection 3.1, we define formally $\Delta(n)$, obtain an upper bound for it, and derive the proof of Proposition 1 (and thus of Theorem 1). In Subsection 3.2 and 3.3, we obtain Theorems 2 and 3 as corollaries.

## 3.1 Proof of Proposition 1

We now prove Proposition 1. Defining the quantity $\Delta(n)$ as a mean to estimate the complexity of the multiplication in $K$, we will actually be led to introduce a whole family of quantities $\delta_{1,n}, \ldots, \delta_{n,n}$ with a similar meaning. We will write $\mathbf{S} = S_1, \ldots, S_n$ for conciseness.

For $i$ in $1, \ldots, n$, let $d_i = \deg_{X_i} T_i = n - i + 1$ and let $E_i$ be the set

$$\{\alpha = (\alpha_1, \ldots, \alpha_i) \mid 0 \leq \alpha_j < d_j,\ 1 \leq j \leq i\}.$$

If $\mathbf{R}$ are any indeterminates, then the monomials

$$\{X_1^{\alpha_1} \cdots X_i^{\alpha_i} \mid \alpha = (\alpha_1, \ldots, \alpha_i) \in E_i\}$$

form a basis of the $\mathbb{Q}[\mathbf{R}, \mathbf{S}]$-algebra $\mathbb{Q}[\mathbf{R}, \mathbf{S}][X_1, \ldots, X_i]/(T_1, \ldots, T_i)$. In particular, let $\mathbf{A}_i = (A_\alpha)_{\alpha \in E_i}$ and $\mathbf{B}_i = (B_\alpha)_{\alpha \in E_i}$ be indeterminates. Using them as coefficients, we define

$$\mathfrak{A}_i = \sum_{\alpha \in E_i} A_\alpha X_1^{\alpha_1} \cdots X_i^{\alpha_i}, \quad \mathfrak{B}_i = \sum_{\alpha \in E_i} B_\alpha X_1^{\alpha_1} \cdots X_i^{\alpha_i}$$

in $\mathbb{Q}[\mathbf{A}_i, \mathbf{B}_i, \mathbf{S}][X_1, \ldots, X_i]$. We can then define the polynomials $\mathbf{C}_i = (C_\alpha)_{\alpha \in E_i} \in \mathbb{Q}[\mathbf{A}_i, \mathbf{B}_i, \mathbf{S}]$ by the relation:

$$\mathfrak{A}_i \mathfrak{B}_i = \sum_{\alpha \in E_i} C_\alpha X_1^{\alpha_1} \cdots X_i^{\alpha_i} \bmod (T_1, \ldots, T_i).$$

The cost $\delta_{i,n}$ of the multiplication modulo $(T_1, \ldots, T_i)$ is formally defined as the minimal size of a straight-line program that computes the polynomials $\mathbf{C}_i$. We then define $\Delta(n) = \delta_{n,n}$; in particular, $\Delta(n) \geq n!$. The example of Subsection 2.2 is a particular case of this construction, which showed that $\Delta(2) \leq 11$.

The following lemma gives the basic way to make use of this notion.

**Lemma 5.** *Let $\mathbf{R}$ be indeterminates, let $i$ be in $1, \ldots, n$, and let $\mathbf{a} = (a_\alpha)_{\alpha \in E_i}$ and $\mathbf{b} = (b_\alpha)_{\alpha \in E_i}$ be in $\mathbb{Q}[\mathbf{R}, \mathbf{S}]$. Writing*

$$\mathfrak{a} = \sum_{\alpha \in E_i} a_\alpha X_1^{\alpha_1} \cdots X_i^{\alpha_i}, \quad \mathfrak{b} = \sum_{\alpha \in E_i} b_\alpha X_1^{\alpha_1} \cdots X_i^{\alpha_i}\ ,$$

*define the polynomials $\mathbf{c} = (c_\alpha)_{\alpha \in E_i}$ in $\mathbb{Q}[\mathbf{R}, \mathbf{S}]$ by*

$$\mathfrak{a}\mathfrak{b} = \sum_{\alpha \in E_i} c_\alpha X_1^{\alpha_1} \cdots X_i^{\alpha_i} \bmod (T_1, \ldots, T_i)\ .$$

*If both families of polynomials $\mathbf{a}$ and $\mathbf{b}$ can be computed by a straight-line program $\Gamma$ of size $L$, then there exists a straight-line program of size $L + \delta_{i,n}$ that computes the same polynomials as $\Gamma$ as well as the polynomials $\mathbf{c}$.*

*Proof.* The polynomials $\mathbf{c}$ are obtained by evaluating the polynomials $\mathbf{C}_i$ at $(\mathbf{a}, \mathbf{b}, \mathbf{S})$. The results then follows by Lemma 1. □

The next lemma will be used twice in what follows, and is obtained by inductive application of the previous one.

**Lemma 6.** *Let* $\mathbf{R}$ *and* $Y_1, \ldots, Y_m$ *be indeterminates, and let* $i$ *be in* $1, \ldots, n$. *For* $j$ *in* $1, \ldots, m$, *let* $(y_{j,\alpha})_{\alpha \in E_i}$ *be in* $\mathbb{Q}[\mathbf{R}, \mathbf{S}]$ *and*

$$y_j = \sum_{\alpha \in E_i} y_{j,\alpha} X_1^{\alpha_1} \cdots X_i^{\alpha_i} \in \mathbb{Q}[\mathbf{R}, \mathbf{S}][X_1, \ldots, X_i].$$

*Let* $\Gamma$ *be a straight-line program which computes polynomials* $g_{-m+1}, \ldots, g_L$ *in* $\mathbb{Q}[Y_1, \ldots, Y_m]$. *Writing* $G_j = g_j(y_1, \ldots, y_m) \in \mathbb{Q}[\mathbf{R}, \mathbf{S}][X_1, \ldots, X_i]$, *let* $G_{j,\alpha}$ *be the unique polynomials in* $\mathbb{Q}[\mathbf{R}, \mathbf{S}]$ *such that the equalities*

$$G_j = \sum_{\alpha \in E_i} G_{j,\alpha} X_1^{\alpha_1} \cdots X_i^{\alpha_i} \bmod (T_1, \ldots, T_i)$$

*hold in* $\mathbb{Q}[\mathbf{R}, \mathbf{S}][X_1, \ldots, X_i]$, *for* $-m+1 \leq j \leq L$. *Suppose that all polynomials* $y_{j,\alpha}$ *can be computed by a straight-line program of size* $\ell$. *Then all polynomials* $G_{j,\alpha}$ *can be computed by a straight-line program of size* $\ell + \delta_{i,n} L$.

*Proof.* We prove the lemma by induction on $L$. If $L = 0$, the result follows from the equalities $G_j = y_{j+m}$, for $-m+1 \leq j \leq 0$. Suppose now that $\Gamma$ has size $L+1$, and let $\Gamma'$ be the straight-line program made by keeping only the first $L$ operations of $\Gamma$. Then by the induction assumption, there exists a straight-line program of size $\ell + \delta_{i,n} L$ that computes the coefficients $G_{j,\alpha}$ for $-m+1 \leq j \leq L$. By definition, the polynomial $G_{L+1}$ takes one of the following forms:

1. $G_{L+1} = \lambda$, with $\lambda \in \mathbb{Q}$;
2. $G_{L+1} = \lambda + G_{a_{L+1}}$, $G_{L+1} = \lambda - G_{a_{L+1}}$ or $G_{L+1} = \lambda G_{a_{L+1}}$, with $-m+1 \leq a_{L+1} \leq L$ and $\lambda \in \mathbb{Q}$;
3. $G_{L+1} = G_{a_{L+1}} + G_{b_{L+1}}$, $G_{L+1} = G_{a_{L+1}} - G_{b_{L+1}}$ or $G_{L+1} = G_{a_{L+1}} G_{b_{L+1}}$, with $-m+1 \leq a_{L+1}, b_{L+1} \leq L$.

The non-trivial case of the multiplication $G_{L+1} = G_{a_{L+1}} G_{b_{L+1}}$ is handled by Lemma 5; all others are immediate. □

For what follows, we need estimates on the cost of modular multiplication. Formally, this is defined the following way. Let $d \geq 1$, and let $\beta_d = \beta_{0,d}, \ldots, \beta_{d-1,d}$, $\gamma_d = \gamma_{0,d}, \ldots, \gamma_{d-1,d}$, and $\tau_d = \tau_{0,d} \ldots, \tau_{d-1,d}$ be $3d$ indeterminates. Let next $\lambda_d = \lambda_{0,d}, \ldots, \lambda_{d-1,d}$ be the polynomials in $\mathbb{Q}[\beta_d, \gamma_d, \tau_d]$ such that $\sum_{0 \leq i < d} \lambda_{i,d} X^i$ is the remainder of the Euclidean division of

$$\left( \sum_{0 \leq i < d} \beta_{i,d} X^i \right) \left( \sum_{0 \leq i < d} \gamma_{i,d} X^i \right)$$

by $\sum_{0 \leq i < d} \tau_{i,d} X^i + X^d$ in $\mathbb{Q}[\beta_d, \gamma_d, \tau_d][X]$. Then, the complexity of modular multiplication in degree $d$ is measured by the complexity of evaluating the polynomials $\lambda_d$. To give concrete estimates, we use the next result, which is for instance proved in [19].

**Lemma 7.** *For any $d \geq 1$, there exists a straight-line program $\mathsf{Mod}_d$ of size at most $4d^2$ that computes the polynomials $\lambda_d$.*

We deduce the following estimate on the quantities $\delta_{i,n}$; the content of the next lemma is essentially an estimate on the cost of nested modular multiplications.

**Lemma 8.** *For $i$ in $1, \ldots, n$, the inequality $\delta_{i,n} \leq 4^i (d_1 \cdots d_i)^2$ holds.*

*Proof.* We denote this property by $\mathcal{P}_i$, and prove it by induction on $i = 1, \ldots, n$.

First we prove $\mathcal{P}_1$. We need to estimate the cost of computing the coefficients of $\mathfrak{C}_1 = \mathfrak{A}_1 \mathfrak{B}_1 \bmod T_1$, where $\mathfrak{A}_1$ and $\mathfrak{B}_1$ are defined as above and $T_1 = S_n + S_{n-1} X_1 + \cdots + S_1 X_1^{n-1} + X_1^n$. By definition all the coefficients of $\mathfrak{A}_1$, $\mathfrak{B}_1$ and $T_1$ can be computed by a straight-line program of size 0, since they are coordinates in $\mathbb{Q}[\mathbf{A}_1, \mathbf{B}_1, \mathbf{S}]$. We apply Lemma 7 with the coordinates $\beta_{d_1}$ replaced by the coordinates $\mathbf{A}_1$, the coordinates $\gamma_{d_1}$ replaced by $\mathbf{B}_1$, and the coordinates $\tau_{d_1}$ replaced by the coordinates $S_n, S_{n-1}, \ldots, S_1$. This yields the bound $\delta_{1,n} \leq 4d_1^2$ as required.

We then perform the inductive step: for $2 \leq i \leq n$, we assume that $\mathcal{P}_{i-1}$ holds, and prove $\mathcal{P}_i$. Let thus $\mathfrak{A}_i$, $\mathfrak{B}_i$ and $\mathfrak{C}_i$ be as above; we now estimate the cost of computing all coefficients of $\mathfrak{C}_i = \mathfrak{A}_i \mathfrak{B}_i \bmod (T_1, \ldots, T_i)$. To this effect, we rewrite $\mathfrak{A}_i$, $\mathfrak{B}_i$, $\mathfrak{C}_i$ and $T_i$ as:

$$\begin{array}{rcl}
\mathfrak{A}_i & = & \mathfrak{A}_{i,0} + \mathfrak{A}_{i,1} X_i + \cdots + \mathfrak{A}_{i,d_i-1} X_i^{d_i-1} \\
\mathfrak{B}_i & = & \mathfrak{B}_{i,0} + \mathfrak{B}_{i,1} X_i + \cdots + \mathfrak{B}_{i,d_i-1} X_i^{d_i-1} \\
\mathfrak{C}_i & = & \mathfrak{C}_{i,0} + \mathfrak{C}_{i,1} X_i + \cdots + \mathfrak{C}_{i,d_i-1} X_i^{d_i-1} \\
T_i & = & T_{i,0} + T_{i,1} X_i + \cdots + T_{i,d_i-1} X_i^{d_i-1} + X_i^{d_i} ,
\end{array}$$

where all polynomials $\mathfrak{A}_{i,j}, \mathfrak{B}_{i,j}, \mathfrak{C}_{i,j}$ and $T_{i,j}$ are in $\mathbb{Q}[\mathbf{A}_i, \mathbf{B}_i, \mathbf{S}][X_1, \ldots, X_{i-1}]$. Note that all coefficients (in $\mathbb{Q}[\mathbf{A}_i, \mathbf{B}_i, \mathbf{S}]$) of the polynomials $\mathfrak{A}_{i,j}, \mathfrak{B}_{i,j}, T_{i,j}$ can be computed by a straight-line program of size 0: for $\mathfrak{A}_{i,j}$ and $\mathfrak{B}_{i,j}$, this is a consequence of their definition; for $T_{i,j}$, this follows from Lemma 3. Note also that it suffices to compute all coefficients (in $\mathbb{Q}[\mathbf{A}_i, \mathbf{B}_i, \mathbf{S}]$) of all polynomials $\mathfrak{C}_{i,j}$ to conclude.

Let $\mathsf{Mod}_{d_i}$ be as in Lemma 7, with indeterminates $\beta_{d_i}, \gamma_{d_i}, \tau_{d_i}$, and let $\lambda_{0,d_i}, \ldots, \lambda_{d_i-1,d_i} \in \mathbb{Q}[\beta_{d_i}, \gamma_{d_i}, \tau_{d_i}]$ be its output. Define $\Lambda_0, \ldots, \Lambda_{d_i-1}$ in $\mathbb{Q}[\mathbf{A}_i, \mathbf{B}_i, \mathbf{S}][X_1, \ldots, X_{i-1}]$ by

$$\Lambda_j = \lambda_{j,d_i}(\mathfrak{A}_{i,0}, \ldots, \mathfrak{A}_{i,d_i-1}, \mathfrak{B}_{i,0}, \ldots, \mathfrak{B}_{i,d_i-1}, T_{i,0}, \ldots, T_{i,d_i-1}) .$$

Then, the equality $\mathfrak{C}_{i,j} = \Lambda_j \bmod (T_1, \ldots, T_{i-1})$ holds. Applying Lemma 6 to $\mathsf{Mod}_{d_i}$ (with $\mathbf{R} = \mathbf{A}_i \cup \mathbf{B}_i$) then shows that all coefficients of $\mathfrak{C}_{i,j}$ can be computed by a straight-line program of size $0 + \delta_{i-1,n} \times (4d_i^2)$, and the induction assumption gives an upper bound of $4^i (d_1 \cdots d_i)^2$ for this quantity. □

*Proof of Proposition 1.* Recall from Corollary 1 that the polynomials $P_\alpha$ in Equation (3) can also be defined as the unique polynomials in $\mathbb{Q}[\mathbf{S}]$ satisfying the equality

$$P = \sum_{\alpha \in E} P_\alpha X_1^{\alpha_1} \cdots X_n^{\alpha_n} \bmod (T_1, \ldots, T_n)$$

in $\mathbb{Q}[\mathbf{S}][X_1, \ldots, X_n]$.

Let $\Gamma$ be a straight-line program that computes $P$. We are going to apply Lemma 6 to $\Gamma$, with $\mathbf{R} = \emptyset$. To do so, we have to estimate the complexity of evaluating the

coefficients of all $X_i$ mod $(T_1, \ldots, T_n)$ on the standard monomial basis $X^E$. For $i < n$, the exponent vector $(0, \ldots, 0, 1, 0, \ldots, 0)$ (with 1 at $i$th position) is in $E$, so we have nothing to compute. For $i = n$, since $T_n = X_1 + \cdots + X_n + S_1$ by Lemma 3, $X_n$ equals $-S_1 - X_1 - \cdots - X_{n-1}$ mod $(T_1, \ldots, T_n)$. Thus, we need only compute $-1$ and $-S_1$, hence a cost of 2.

We deduce from Lemma 6 that all polynomials $P_\alpha$ can be computed by a straight-line program of size $\delta_{n,n} L + 2$. Lemma 8 finishes the proof. □

To conclude this subsection, we mention some improved bounds for the function $\Delta$. To this effect, let us introduce the function $\mathcal{M} : \mathbb{N} \to \mathbb{N}$, such that $\mathcal{M}(d)$ is the complexity of univariate polynomial multiplication in degree $d$ (the precise definition is similar to that of the cost of modular multiplication above). Thus, $\mathcal{M}(d) = 2d^2$ for the naive multiplication algorithm, but one can take $\mathcal{M}(d) \in O(d \log d \log\log d)$ using FFT multiplication [19].

A refinement on the results above consists in noticing that the cost of modular multiplication in degree $d$ can actually be reduced from $4d^2$ to $C\mathcal{M}(d)$, for some universal constant $C$, under suitable assumptions on $\mathcal{M}$, see [19, Chapter 9]. It is then easy to deduce that $\Delta(n)$ admits the improved bound $C^n \mathcal{M}(1) \cdots \mathcal{M}(n)$.

## 3.2 Proof of Theorem 2

We conclude the proof of Theorem 2, using two lemmas. Let $\mathfrak{b} = (b_1, \ldots, b_n)$ be $\mathbb{Q}$-algebra generators of $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$, and let $\mathfrak{c} = (c_s)_{s \in \mathfrak{S}_n}$ be a basis of the free $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$-module $\mathbb{Q}[X_1, \ldots, X_n]$.

Let $P$ be a polynomial that can be computed in time $L$. We want to obtain the complexity of evaluation of all the polynomials $P_s$ in the decomposition

$$P = \sum_{s \in \mathfrak{S}_n} P_s(b_1, \ldots, b_n) c_s \ .$$

By Proposition 1, there exists a straight-line program of size $\Delta(n)L + 2$ which evaluates all the polynomials $\overline{P}_\alpha$ in the decomposition

$$P = \sum_{\alpha \in E} \overline{P}_\alpha(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) X_1^{\alpha_1} \cdots X_n^{\alpha_n} \ .$$

We first apply a basis change from the standard monomial basis to $\mathfrak{c}$ to obtain the decomposition

$$P = \sum_{s \in \mathfrak{S}_n} \tilde{P}_s(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) c_s \ .$$

The following lemma estimates the overhead induced by this operation.

**Lemma 9.** *There exists a constant $L(\mathfrak{c})$ such that all polynomials $\tilde{P}_s$ can be computed in time $\Delta(n)L + 2 + L(\mathfrak{c})$.*

*Proof.* Let $M$ be the $n! \times n!$ matrix of change of basis from the standard monomial basis into $\mathfrak{c}$; the coefficients of this matrix are symmetric polynomials, which we choose to represent as polynomials in $\overline{\sigma}_1, \ldots, \overline{\sigma}_n$. Let $\ell(\mathfrak{c})$ be the size of a straight-line program that evaluates all entries of $M$,

The polynomials $\tilde{P}_s$ can be obtained from the polynomials $\overline{P}_\alpha$ by matrix-vector multiplication with the matrix $M$. We deduce that they can be computed in time $\Delta(n)L + 2 + \ell(\mathfrak{c}) + n!(2n! - 1)$, the term $n!(2n! - 1)$ being a coarse upper bound for the cost for matrix-vector product in size $n! \times n!$. Thus, the quantity $L(\mathfrak{c}) = \ell(\mathfrak{c}) + n!(2n! - 1)$ satisfies the claim of the lemma. □

We then apply a change of algebra generators from $\overline{\sigma}$ to $\mathfrak{b}$, to get the requested decomposition

$$P = \sum_{s \in \mathfrak{S}_n} P_s(b_1, \ldots, b_n) c_s \ .$$

The following lemma will then conclude the proof of Theorem 2.

**Lemma 10.** *There exists a constant $L(\mathfrak{b})$ such that all polynomials $P_s$ can be computed in time $\Delta(n)L + 2 + L(\mathfrak{c}) + L(\mathfrak{b})$.*

*Proof.* Since $\mathfrak{b}$ forms a basis of the algebra of symmetric polynomials, there exist unique polynomials $\mathcal{S}_1, \ldots, \mathcal{S}_n$ in $\mathbb{Q}[B_1, \ldots, B_n]$ such that $\overline{\sigma}_i = \mathcal{S}_i(b_1, \ldots, b_n)$ for $i = 1, \ldots, n$; then, $P_s = \tilde{P}_s(\mathcal{S}_1, \ldots, \mathcal{S}_n)$ for all $s \in \mathfrak{S}_n$. Let $L(\mathfrak{b})$ be the size of a straight-line program that evaluates $\mathcal{S}_1, \ldots, \mathcal{S}_n$. The result now follows from applying Lemma 1 to the polynomials $\mathcal{S}_i$ and $\tilde{P}_s$. □

We conclude this subsection by discussing particular cases of the two constructions above.

As an example of $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$-basis change, we evaluate $L(\mathfrak{c})$ in the case of the Schubert basis, introduced in [14] (see [12] for a survey). Schubert polynomials are naturally indexed by $\mathfrak{S}_n$ and are defined as follows. We first define the maximal permutation $\omega = [n, n-1, \ldots, 1]$ in $\mathfrak{S}_n$, and the associated Schubert polynomial $\mathbb{X}_\omega = X_1^{n-1} X_2^{n-2} \cdots X_{n-1}$. Let now $s$ be any permutation, and consider a minimal decomposition of $s^{-1}\omega$ as a product $\tau_{i_1} \cdots \tau_{i_r}$ of elementary transpositions. The *Schubert polynomial* $\mathbb{X}_s$ is defined as $\partial_{i_1} \cdots \partial_{i_r}(\mathbb{X}_\omega)$; this definition is properly independent of the choice of the factorization, because the divided difference operators $\partial_i$ happen to satisfy the braid relations.

The Schubert polynomials form a basis of the $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$-module $\mathbb{Q}[X_1, \ldots, X_n]$; their definition shows that the matrix $M$ that expresses the Schubert polynomials in terms of the standard monomial basis is triangular, with $\{0, 1\}$ entries, and its inverse matrix $M$ also has integer entries. Thus, following the proof of Lemma 9, we see that $L((\mathbb{X}_s)_{s \in \mathfrak{S}_n})$ is at most $n!(2n! - 1)$. However, $M$ being a very sparse matrix, we expect that a much better bound could be found.

We finally illustrate the second construction, the change of algebra generators for $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$ on the complete symmetric polynomials $\mathbf{h} = (h_1, \ldots, h_n)$ and the symmetric power sums $\mathbf{p} = (p_1, \ldots, p_n)$, by giving bounds on $L(\mathbf{h})$ and $L(\mathbf{p})$. To this effect, one could use the Newton relations; however, better can be done. We let $\mathcal{M} : \mathbb{N} \to \mathbb{N}$ denote the complexity of multiplying univariate polynomials (see Subsection 3.1).

**Lemma 11.** *We have $L(\mathbf{h}) \in O(\mathcal{M}(n))$ and $L(\mathbf{p}) \in O(\mathcal{M}(n))$.*

*Proof.* Recall from Subsection 2.3 that the generating series for elementary and complete symmetric polynomials and symmetric power sums satisfy the relations:

$$S(z) = \frac{1}{H(z)}; \quad S(z) = \exp(-P(z)) \ .$$

Hence, using Newton iteration for inverse and exponential of power series [2, 17, 19], one can recover the first $n+1$ coefficients of $S(z)$ from those of either $H(z)$ or $P(z)$ by a straight-line program of size $O(\mathcal{M}(n))$. This means that the elementary symmetric polynomials in $n$ variables can be computed from either the complete symmetric polynomials or the symmetric power sums in time $O(\mathcal{M}(n))$. □

## 3.3 Complexity of evaluation of the Reynolds operator

The Reynolds operator is a $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$-linear projection

$$\mathbb{Q}[X_1, \ldots, X_n] \to \mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n},$$

and as such, is a quite important tool in the study of symmetric polynomials. For $P$ in $\mathbb{Q}[X_1, \ldots, X_n]$, $R(P)$ is given by the formula

$$R(P) = \frac{1}{n!} \sum_{s \in \mathfrak{S}_n} s \cdot P = \frac{1}{n!} \sum_{s \in \mathfrak{S}_n} P\left(X_{s(1)}, \ldots, X_{s(n)}\right) \ .$$

Since $R(P)$ is symmetric, there exists a unique polynomial $Q \in \mathbb{Q}[S_1, \ldots, S_n]$ such that $R(P) = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$: our goal is now to relate the complexity of evaluation of $Q$ to that of $P$.

A brute-force use of the definition would consist in applying Proposition 1 to all conjugates of $P$; this would induce a loss of a factor $n!$ in complexity. Luckily enough, one can essentially read off a straight-line program for $Q$ from the straight-line program giving the coefficients $P_\alpha$ of $P$ on the standard monomial basis. Indeed, since $R$ is a $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$-module morphism, we have, writing for short $X^\alpha = X_1^{\alpha_1} \cdots X_n^{\alpha_n}$:

$$R(P) = R\left(\sum_{\alpha \in E} P_\alpha(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) X^\alpha\right) = \sum_{\alpha \in E} P_\alpha(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) R(X^\alpha) \ .$$

Furthermore, $R(X^\alpha)$ does not depend on the order of the exponents in $\alpha = (\alpha_1, \ldots, \alpha_n)$. Let then $F$ denote the set of all partitions in $E$, that is, those elements that form weakly decreasing sequences. The previous sum can be rewritten as

$$R(P) = \sum_{\mu \in F} \left( \left( \sum_{\alpha \in E, \alpha \text{ permutation of } \mu} P_\alpha(\overline{\sigma}_1, \ldots, \overline{\sigma}_n) \right) R(X^\mu) \right) \ .$$

Now, let us suppose that $P$ can be computed in time $L$, so that all $P_\alpha$ can be computed in time $\Delta(n)L+2$ by Proposition 1. Let next $D(n)$ be the size of a straight-line program that computes the polynomials $R(X^\mu)$ in terms of $\overline{\sigma}_1, \ldots, \overline{\sigma}_n$. Still denoting $Q$ the polynomial such that $R(P) = Q(\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$, the above formula shows that, knowing all $P_\alpha$ and $R(X^\mu)$, $Q$ can be computed for $n!$ additional operations, that is, in total time $\Delta(n)L + n! + D(n) + 2$. Thus, to conclude the proof of Theorem 3, it suffices to give an upper bound on $D(n)$. This is the object of the upcoming lemma.

**Lemma 12.** *For $n \geq 1$, the inequality $D(n) \leq 2(8^n - 1)$ holds.*

*Proof.* To a partition $\mu$, we associate the *monomial symmetric polynomial* $m_\mu$ obtained by summing up the monomials in the orbit of $X^\mu$; these polynomials form a vector-space basis of $\mathbb{Q}[X_1, \ldots, X_n]^{\mathfrak{S}_n}$. Note that $R(X^\mu)$ coincides with $m_\mu$ up to a non-zero constant factor; so we start by computing the $m_\mu$ polynomials, $\mu \in F$. We use induction, following the standard (SAGBI) rewriting process of monomial symmetric polynomials in terms of elementary symmetric polynomials w.r.t. the degree lexicographic term order.

Let $\mu = (\mu_1, \ldots, \mu_{n-1}) \in F$ be fixed, let $k$ be the number of non-zero parts in it, and $\nu$ the unique partition such that $X^\mu$ factors as $X^\nu X_1 \cdots X_k$. Write the expansion of the product $m_\nu \sigma_k$ in the monomial basis as:

$$m_\nu \sigma_k = \sum_\eta c_\eta m_\eta \ ,$$

the coefficients $c_\eta$ being in $\mathbb{N}$. The leading term of this product is the product of the leading terms of its operands, namely $X^\nu X_1 \cdots X_k = X^\mu$; it follows that $c_\mu = 1$. Let now $\eta$ be a partition appearing in the right hand side: it follows from the previous remark that $\eta \leq_{\text{deglex}} \mu$. Furthermore, $\eta_i \leq \mu_i$, whenever $\mu_i > 0$, and otherwise, $\eta_i \in \{0, 1\}$. It is then straightforward to check that:

(*i*) either $\eta$ is right away in $F$,

(*ii*) or $m_\eta$ is of the form $m_\eta = X_1 \cdots X_n m_{\eta'}$ with $\eta'$ in $F$.

Define $c'_\eta = c_\eta$ for all $\eta$ in case (*i*), and $c'_\eta = 0$ otherwise. Define also $d_{\eta'} = c_\eta$, for all $\eta, \eta'$ as in case (*ii*), and $d_{\eta'} = 0$ otherwise. Altogether, $m_\mu$ can be written as

$$m_\mu = m_\nu \sigma_k - \sum_{\eta \in F,\, \eta <_{\text{deglex}}\, \mu} c'_\eta m_\eta - \sigma_n \sum_{\eta' \in F,\, \eta' <_{\text{deglex}}\, \mu} d_{\eta'} m_{\eta'} \ .$$

Furthermore, by considering the maximal possible number of monomials in $\sigma_k$, we see that the total number $N_\mu$ of terms appearing with a non-zero coefficient in the two sums can be bounded by $2^n$. We now switch back to the $R(X^\mu)$, and re-introduce our symmetric polynomials $\overline{\sigma}_k$; this yields:

$$R(X^\mu) = \gamma_\nu R(X^\nu) \overline{\sigma}_k - \sum_{\eta \in F,\, \eta <_{\text{deglex}}\, \mu} c''_\eta R(X^\eta) - \overline{\sigma}_n \sum_{\eta' \in F,\, \eta' <_{\text{deglex}}\, \mu} d'_{\eta'} R(X^{\eta'}) \ ,$$

for some constants $\gamma_\nu$, $c''_\eta$ and $d'_{\eta'}$. Counting the operations appearing in the right-hand side expression, we see that a straight-line program computing all polynomials $\{R(X^\eta), \eta \in F, \eta <_{\text{deglex}} \mu\}$ can be extended to further compute $R(X^\mu)$, for an additional cost of $2N_\mu + 3$ operations. So, there exists a straight line program of size $\sum_{\mu \in F}(2N_\mu + 3)$, which computes $R(X^\mu)$ for all $\mu \in F$.

Now, $F$ is in bijection with the Dyck paths of length $2n$, so that the cardinality of $F$ is given by the $n$-th Catalan number $C(n) = \frac{1}{n+1}\binom{2n}{n}$. The estimate $(2 \cdot 2^n + 3)C(n) \leq 2(8^n - 1)$ proves the lemma. ∎

# References


[1] N. Bourbaki. *Éléments de mathématique. Algèbre. Chapitres 1 à 3*. Hermann, Paris, 1970.

[2] R. P. Brent. Multiple-precision zero-finding methods and the complexity of elementary function evaluation. In *Analytic computational complexity (Proc. Sympos., Carnegie-Mellon Univ., Pittsburgh, Pa., 1975)*, pages 151–176. Academic Press, New York, 1976.

[3] P. Bürgisser, M. Clausen, and A. M. Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, 1997.

[4] A. Colin. Solving a system of algebraic equations with symmetries. *Journal of Pure and Applied Algebra*, 117/118:195–215, 1997.

[5] D. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer-Verlag, 1997.

[6] L. Ducos and C. Quitté. Algèbre de décomposition universelle. Technical report, Université de Poitiers, 1996.

[7] W. Feit. A method for computing symmetric and related polynomials. *Journal of Algebra*, 234:540–544, 2000.

[8] P. Gaudry and É. Schost. Construction of secure random curves of genus 2 over prime fields. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 239–256. Springer, 2004.

[9] M. Giusti, K. Hägele, J. Heintz, J.-E. Morais, J.-L. Montaña, and L.-M. Pardo. Lower bounds for Diophantine approximation. *Journal of Pure and Applied Algebra*, 117/118:277–317, 1997.

[10] M. Giusti, J. Heintz, J.-E. Morais, J. Morgenstern, and L.-M. Pardo. Straight-line programs in geometric elimination theory. *Journal of Pure and Applied Algebra*, 124:101–146, 1998.

[11] M. Giusti, J. Heintz, J.-E. Morais, and L.-M. Pardo. When polynomial equation systems can be solved fast? In *Proceedings of AAECC 11*, volume 948 of *Lecture Notes in Computer Science*, pages 205–231. Springer-Verlag, 1995.

[12] A. Lascoux. *Symmetric functions and combinatorial operators on polynomials*, volume 99 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 2003.

[13] A. Lascoux and P. Pragacz. $S$-function series. *Journal of Physics. A. Mathematical and General*, 21(22):4105–4114, 1988.

[14] A. Lascoux and M.-P. Schützenberger. Polynômes de Schubert. *Comptes Rendus des Séances de l'Académie des Sciences. Série I. Mathématique*, 294(13):447–450, 1982.

[15] I. G. Macdonald. *Symmetric functions and Hall polynomials*. Oxford Mathematical Monographs. The Clarendon Press Oxford University Press, New York, second edition, 1995. With contributions by A. Zelevinsky, Oxford Science Publications.

[16] N. Rennert and A. Valibouze. Calcul de résolvantes avec les modules de Cauchy. *Experimental Mathematics*, 8(4):351–366, 1999.

[17] A. Schönhage. The fundamental theorem of algebra in terms of computational complexity. Technical report, University of Tübingen, 1982.

[18] B. Sturmfels. *Algorithms in invariant theory*. Texts and Monographs in Symbolic Computation. Springer-Verlag, 1993.

[19] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.