[5] R. Steinberg, An occurrence of the Robinson- Schensted correspondence, J. Algebra 113 (1988), 523-528.

[6] M. van Leeuwen, The Robinson-Schensted and Schützenberger algorithms and interpretations, CWI Tract 84 (1991), 65-88.

[7] D. White, Some connections between the Littlewood- Richardson rule and the construction of Schensted, J. Comb. Theory Ser. A 30 (1981), 237-247.

Computing the Hilbert-Poincaré series of monomial ideals, applications to Gröbner bases*

Carlo Traverso
Dipartimento di Matematica
Università di Pisa
traverso@dm.unipi.it

Abstract

The Hilbert-Poincaré series of an homogeneous ideal, or of the homogenization of an affine ideal, can be computed through the associated staircase of a Gröbner basis of the ideal. In this paper we review some recent results on algorithms to compute the Hilbert-Poincaré series of a staircase, see [BCRT], and some applications of the computation of the Hilbert-Poincaré series to the computation of Gröbner bases, see [GT], [Ca].

1 The computation of the Hilbert-Poincaré series.

This section is a summary of the paper [BCRT], to which we refer for complete proofs and results.

In the computation of the Hilbert-Poincaré series of an homogeneous ideal I, the known algorithms, [MM1], [MM3], [KP], [BS], [BCR] have a first algebraic step coinciding with the computation of the associated Gröbner basis w.r.t. any term-ordering and the corresponding initial ideal (the *associated staircase*), and a second combinatorial step that from the staircase computes the Hilbert-Poincaré series.

The algorithms of [MM1] and [MM3] use techniques similar to the computation of a resolution: the algorithms of [KP] and [BCR] proceed by induction on the dimension; the algorithm of [BS] proceeds by induction on the number of generators of the initial ideal (the cogenerators of the staircase).

Usually, combinatorial algorithms can be speeded by a "Divide and Conquer" approach: splitting the problem into two smaller problems of approximately the same size. In successful cases this trades a linear step for a logarithmic step, and can reduce from exponential to polynomial complexity.

Our approach explains how to split a staircase through the choice of a monomial (the *pivot*), then we discuss how to design a strategy for the choice of the pivot. The worst case complexity is not improved, since in some extreme cases every splitting is bad, (the computation of Hilbert-Poincaré series is at least as difficult as a NP-complete problem in the number of variables, see [BS]) but in several practical cases the situation is much better; in particular, our algorithm in the best case has a complexity that is a linear factor better than the best case of [BS], and can be specialized, with a choice of the splitting strategy, to the algorithm of [BCR]. In practice, a simple random strategy is quite good, avoids the costly computations involved in choice of an optimal variable of [BCR], and marginally improves the performance even in the optimal Borel-normed case.

The algorithms have been implemented, both in CoCoA, [GN] and AlPi, [TD]. Some test cases are given.

*This research was performed with the contribution of C.N.R., M.U.R.S.T, and CEC contract ESPRIT B.R.A. n.6846 POSSO

1.1 Staircases

A *staircase* S (also called *Ferrer diagram* or *order ideal of monomials*) is a set of elements of $\mathbf{N}^n$ such that if $(a_1, \ldots, a_n) \in S$ and $b_i \leq a_i$ then $(b_1, \ldots, b_n) \in S$.

On $\mathbf{N}^n$ there is a partial ordering $(a_1, \ldots, a_n) \leq (b_1, \ldots, b_n)$ iff $a_i \leq b_i$ for each i. The corresponding lattice operations $\alpha \wedge \beta$, $\alpha \vee \beta$ are the componentwise min and max.

Elements of $\mathbf{N}^n$ correspond to terms (power-products, monic monomials) of $k[x_1, \ldots, x_n]$; the partial ordering corresponds to divisibility ($\alpha \leq \beta \iff X^\alpha \mid X^\beta$). We often identify α with X^α, and use notations referring to both indifferently. In particular, if $\alpha = (a_1, \ldots, a_n) \in \mathbf{N}^n$, $|\alpha| = \sum a_i$ will be called the *degree* of α. The operations $\wedge$ and $\vee$ correspond to GCD and lcm.

We say that an element of $\mathbf{N}^n$ is a *pure power* if it has only one coordinate that is non zero. Otherwise it is called *mixed*.

If $I \subseteq k[x_1, \ldots, x_n]$ is an ideal, and we have a term-ordering, then the staircase associated to I is the set of exponents $\alpha = (a_1, \ldots, a_n)$ such that no element of I has leading term equal to X^α. The staircase associated to I can be computed through a Gröbner basis, taking all exponents such that no leading term of the Gröbner basis divides them, and the X^α are a linear basis of $k[X]/I$. Conversely, from the staircase we can recover the leading terms of the reduced Gröbner basis, corresponding to the minimal elements of the complementary of the staircase.

The staircase is usually given through these minimal elements, that are called the *minimal cogenerators* of the staircase. Given a set G of elements of $\mathbf{N}^n$, there is a maximal staircase disjoint from G, and it is called the staircase *cogenerated* by G. It is the complementary of the monoideal generated by G.

We denote with $[\alpha_1, \ldots, \alpha_m]$ the staircase cogenerated by $\alpha_1, \ldots, \alpha_m$; if S is a staircase, denote with $[S]$ the minimal set of cogenerators of S. In particular, $[[\alpha_1, \ldots, \alpha_m]]$ is obtained deleting from $\{\alpha_1, \ldots, \alpha_m\}$ all the elements that are multiple of another element (the notation implicitly assumes that no duplications appear in $\{\alpha_1, \ldots, \alpha_m\}$). The algorithm for operating such deletions is an essential tool, and its efficient implementation is very important.

Given two staircases S_1, S_2, we say that S_1 is *strictly smaller* than S_2 if S_1 is a proper subset of S_2 and moreover an injective map ϕ exists from $[S_1]$ to $[S_2]$ such that $\phi(\alpha) \geq \alpha$. Given a staircase, only a finite number of strictly smaller staircases exists.

A *T-staircase* is a translate of a staircase; all that is said for staircases applies, with minor modifications, to T-staircases. Most of what we will prove will be applicable to T-staircases without modifications, and we will not even quote it.

Given a staircase (or a T-staircase) S one defines its Hilbert-Poincaré series being the formal power series in the indeterminate T defined by $\sum_{i=0}^{\infty} d_i T^i$, where d_i is the number of elements of S of degree i, and is denoted by H_S (of course, this being an infinite formula, it is not an algorithm). Clearly, if $S' = \alpha + S$, then $H'_S = T^{|\alpha|} H_S$.

If I is an homogeneous ideal of $k[X]$, the Hilbert-Poincaré series of $k[X]/I$ is defined, and coincides with the Hilbert-Poincaré series of the associated staircase. This is the motivation of the interest in computing the Hilbert-Poincaré series of staircases.

1.2 Splitting a staircase

We will consider two types of splitting of staircases: as a product of staircases (a *vertical splitting*), and as disjoint union of a staircase and a T-staircase (a *horizontal splitting*).

A vertical splitting is possible if and only if we can identify two disjoint subsets X_1 and X_2 of the variables such that any minimal cogenerator is a term in either the variables X_1 or X_2. In that case, S is a product of two staircases S_1 and S_2, in X_1 and X_2, each one cogenerated by the corresponding cogenerators of S. We have $H_S = H_{S_1} H_{S_2}$, $\langle S \rangle = \langle S_1 \rangle \langle S_2 \rangle$, since the elements of S of degree d correspond to pairs (α, β), $|\alpha| = d_1$, $|\beta| = d_2$, $d = d_1 + d_2$.

Moreover $e_S = e_{S_1} e_{S_2}$, $\delta_S = \delta_{S_1} + \delta_{S_2}$, $c_S = c_{S_1} + c_{S_2}$ (just apply the definition of multiplicity and dimension).

A horizontal splitting is always possible, unless S is reduced to $\{0\}$: if $\alpha \neq 0$ is an element of the staircase, let $S' = \{\beta \in S \mid \alpha \leq \beta\}$, and $S_1 = S \setminus S'$; then S_1 is a staircase, and S' is a translate of a staircase S_2 by α, $S' = \alpha + S_2$. The cogenerators of S_1 are obtained by deleting from the cogenerators of S all the multiples of α and adding α. The cogenerators of S_2 are obtained as follows: let $\{\beta_i\}$ be a set of cogenerators of S; then a set of cogenerators of S_2 is given by $\beta_i : \alpha$, where the operation $:$ is defined as follows: $(b_1, \ldots, b_m) : (a_1, \ldots, a_m) = (c_1, \ldots, c_m)$, where $c_i = \max(b_i - a_i, 0)$. The operation corresponds to the operation $I : J$ between ideals.

In this case $H_S = H_{S_1} + H_{S'} = H_{S_1} + T^{|\alpha|} H_{S_2}$, by the definition of the Hilbert-Poincaré series; moreover $\langle S \rangle = \langle S_1 \rangle + T^{|\alpha|} \langle S_2 \rangle$, $d_S = \max(d_{S_1}, d_{S_2})$, $c_S = \min(c_{S_1}, c_{S_2})$, $e_S = e_{S_1} + e_{S_2}$ if $d_{S_1} = d_{S_2}$, $e_S = e_{S_1}$ if $d_{S_1} > d_{S_2}$, $e_S = e_{S_2}$ if $d_{S_1} < d_{S_2}$.

The element α is uniquely identified by the splitting, (it is the minimal element of S') and is called the *pivot* of the splitting.

In our applications, we assume that α is smaller than one of the cogenerators of S (in multiplicative notation, properly divides it), and different from 0; this is possible unless the staircase has no cogenerators or all cogenerators are of degree one. In that case, both S_1 and S_2 are strictly smaller than S. This implies that any chain of such splittings must terminate.

1.3 Terminating the algorithm

We can proceed in splitting the staircase until each piece is cogenerated by one element of degree 1, or by no element, but this is impractical. We terminate the splitting when we are reduced to a set of cogenerators consisting of some pure powers and a few elements, pairwise coprime, that are not pure powers.

The following theorem holds:

THEOREM 1. *Assume that a staircase S has a minimal set of cogenerators $\{\pi_1, \ldots, \pi_r, \mu_1, \ldots, \mu_s\}$, such that the π_i are pure powers, and the μ_j are mixed and pairwise coprime. Consider the partition of the variables $\{\Pi_0, \ldots, \Pi_s\}$, where Π_0 is the set of the variables not appearing in the μ_j, and Π_l is the set of the variables appearing in μ_l.*
Then S is split vertically according to $\{\Pi_0, \ldots, \Pi_s\}$ into $s+1$ staircases $S_0, \ldots, S_s$, and every $[S_i]$ contains at most one mixed power.

The proof is immediate. Of course, the degenerate case that Π_0 is empty is possible.

The computation of the Hilbert-Poincaré series of these simple staircases is done in the two following theorem:

THEOREM 2. *Let S be a staircase in $\mathbf{N}^n$ such that $[S] = \{\pi_1, \ldots, \pi_m\}$, $\pi_i = x_i^{c_i}$.*
Then $H_S = \prod(1 - T^{c_i})/(1 - T)^n$

The proof is obtained through a further vertical splitting in staircases in $\mathbf{N}$, that have either no cogenerators (they coincide with $\mathbf{N}$) or one cogenerator π_i (they coincide with $\{0, 1, \ldots, c_i - 1\}$).

In the first case $H_S = 1 + T + T^2 + \ldots + T^i + \ldots = (1 - T)^{-1}$; in the second case $H_S = 1 + T + \ldots + T^{c_i - 1} = (1 - T)^{-1}(1 - T^{c_i})$.

THEOREM 3. *Let S be a staircase in $\mathbf{N}^{m+r}$, and assume $[S] = \{\pi_1, \ldots, \pi_m, \tau\}$, $\pi_i = x_i^{a_i}$, $\tau = x_1^{b_1} \cdots x_m^{b_m} y_1^{c_1} \cdots y_r^{c_r}$, $a_i > b_i > 0$, $c_j > 0$.*
Then

$$H_S = \left(\prod (1 - T^{a_i}) - T^{|c|} \prod (T^{b_i} - T^{a_i}) \right) / (1 - T)^{m+r}$$

where $|c| = \sum c_j$.

The proof is done by considering two special subcases.

If $r = 0$ consider the staircase S' cogenerated by $x_1^{a_1}, \ldots, x_m^{a_m}$ and split it with pivot $\alpha = x_1^{b_1} \cdots x_m^{b_m}$: then S' is disjoint union of S and $\alpha + (S' : \alpha)$, and $S' : \alpha$ is cogenerated by $x_1^{a_1 - b_1}, \ldots . x_n^{a_n - b_n}$: hence H_S is computed by difference.

If $m = 0$ then consider the staircase $S' = \mathbf{N}^r$ and split it with pivot $\beta = y_1^{c_1} \cdots y_r^{c_r}$. Then S' is disjoint union of S and $\beta + S'$, and H_S is computed by difference.

In the general case, split S with pivot $y_1^{c_1} \cdots y_r^{c_r}$; then $S = S_1 \cup (\gamma + S_2)$, $\gamma = y_1^{c_1} \cdots y_r^{c_r}$, S_1 cogenerated by $x_1^{a_1}, \ldots, x_m^{a_m}, y_1^{c_1} \cdots y_r^{c_r}$, S_2 cogenerated by $x_1^{a_1}, \ldots, x_m^{a_m}, x_1^{b_1} \cdots x_m^{b_m}$ and we are reduced to the two previous subcases.

1.4 The choice of a splitting

The vertical splittings appear occasionally, but when they are possible in an early stage of the algorithm their importance is dramatic. They are not so easy to discover, so it is wise to search for them when we have an hint that one might exist. We will see an example in which the algorithm is exponential without vertical splittings, but becomes very simple if we look for them. With this example the other known algorithms perform badly.

1.4.1 Vertical splittings

The search for the most general vertical splitting is easy, but often useless; probably it is convenient only when the sum of the degrees of the non-pure powers is not much larger than the number of variables.

1.4.2 Horizontal splittings: choice of the pivot

The horizontal splittings can always be found; an optimal strategy would be to find every time a splitting such that the two pieces have sets of cogenerators that have one half of the cogenerators of the original staircase. This is possible (and easy) when there are two variables, but is impossible in general. A strategy that allows splittings as balanced as possible is very useful. The reason for looking for such a strategy is the following: the algorithm for one splitting is of quadratic complexity (the interreduction of $S : \alpha$); splitting the cost in two at every step is as good as possible.

There are several possible heuristics for the choice of the pivot. The choice that has appeared more convenient is the following: choose a variable that appears in at least two mixed terms, and some terms in which the variable appears. Then choose as pivot the GCD of these terms. In particular, choosing a random variable among those that appears in most terms, and three random terms (or two if only two exist) among those that contain this variable, the practical performance is often quite satisfying. (In some special cases it seems convenient not to choose the three terms at random, but to choose them in a way to have a larger GCD; this heuristics has however not yet been implemented.)

If no variable appears in more than one mixed term, these terms are all coprime, and we can terminate the algorithm as described in the previous subsection.

Probably an uniform strategy like this one is not convenient for every case, or at every point of the algorithm, and the issue of a good heuristics is widely open. However the results even with this rough strategy are quite good.

1.5 Comparison with the other known algorithms

The algorithms of [KP] and [Ho] coincide with the present algorithm when the pivot is chosen to be a variable.

The algorithm of [BCR] coincides with the present algorithm, in the following variant: choose one variable x_i, and take as pivot x_i^n, where n is the minimum degree in which x_i appears. The algorithm requires the computation of several splittings for choosing the best variable; the overhead can be frequently reduced by special considerations. When there are several variables and none is especially good, and we have to consider them all, it seems that the cost of computing several splittings is difficult to recover by discovering a relatively better variable.

The algorithm of [BS] is related to ours, with a difference. A staircase S is represented as the difference set $S_1 \setminus S_2$, S_1 obtained removing one of the cogenerators of S, and the $S_2 = S_1 \setminus S$, the T-staircase contained in S_1 composed of the multiples of this cogenerator. Both staircases are simpler (in a different sense than ours) and the termination can be done as in our algorithm. The choice of the cogenerator to remove is guided by an heuristics dependent on the term-ordering.

The algorithm of [MM2] has mainly theoretical interest; it derives the Hilbert-Poincaré series in a simple way from the construction of a resolution, and it is known to be practically inefficient.

The algorithm as explained above was implemented in COMMON-LISP and included in AlPi, and in Pascal and included in CoCoA.

The practical comparison of algorithms implemented in an heterogeneous way is hard, since it is difficult to separate the effect of the algorithm and the effect of the clever implementation; moreover some tricks can considerably speed the algorithms, and sometimes a trick can be applied to an algorithm and not to another.

To allow a fair evaluation of the algorithm, in the COMMON-LISP implementation we have included an approximate measure of the complexity. We consider as unit of complexity an operation on terms, such as GCD or lcm. This allows to compare the algorithms in a way that is relatively implementation-independent.

A very small modification of the implementation in COMMON-LISP (only a few lines of code) implements the algorithm of [BS]. The performance of this rough implementation is not good, compared with the timings given in [BS], and it is not clear if this is due to an optimization of the implementation or to improvements to the algorithm; one can compare anyway the experimental complexity data (the number of steps, the sum of the m^2) and the algorithm of [BS] appears to be inferior (sometimes dramatically inferior) in all but some special cases with very few cogenerators.

The comparison of the timings given in [BS] and the timings obtained in our implementation of the algorithm of [BCRT] show a slightly better performance of our algorithm; the comparisons of both algorithms in our implementation shows an improvement of the performance by a factor of 10 in these examples; for other examples, not reported in [BS], the improvement of our algorithm is varying, ranging from even performance to one minute against one day (and even more for a very special example, see below).. For some very special examples (few generators in many variables) our algorithm is slower, but the overall time is very low anyway.

In general, the algorithm of [BS] has good performance when the staircase is good (but in this case all the algorithms perform well), but in the bad case our algorithm is clearly superior.

For an extended report of the computations, see [BCRT].

The COMMON-LISP sources are available by anonymous FTP on `gauss.dm.unipi.it` (131.114.6.55) in the directory `pub/alpi-cocoa/hilbert`.

1.6 A simple bad example

The computation of the Hilbert-Poincaré series in general, and even computing the dimension, is a problem that is harder than an NP-complete problem (see [BS]), hence bad examples are unavoidable.

Here we study a very simple example that has a very bad behaviour, unless we allow vertical splittings and randomized algorithms: avoiding the general vertical splittings, or taking a "natural" ordering of the variables requires exponential time.

The example is the following:

$$I = (x_0x_1, x_1x_2, \ldots, x_{n-1}x_n)$$

A randomized algorithm splits the staircase horizontally in two staircases, one with $n-2$ and one with $n-3$ elements; these can be split vertically, and the expected lengths are in ratio $3:1$. Hence the expected complexity is polynomial.

If no vertical splittings are allowed, more than $2^{n/3}$ steps are necessary. Moreover, if we always choose as pivot the lowest (or highest) possible variable appearing in more than one monomial, the splittings are always bad, and no vertical splittings are useful.

The algorithm of [BS] in this case has the same type of behaviour; however their heuristics is in this case the worst possible, and even with vertical splittings the algorithm remains exponential.

Indeed, with 42 variables the example can be computed with our implementation in $2''$. (with 101 variables it takes $40''$); without vertical splittings in 42 variables it takes $6'$, and $13h47'$ with the algorithm of [BS].

2 Applications: change of ordering in Gröbner basis computing

This section reports some recent results, that will be contained in expanded form in [GT].

In the computation of Gröbner bases of ideals of dimension zero the use of linear algebra is a possible useful tool, provided that the zero-dimensionality is explicitly known, see [FGLM], [MMM], [MT]. For most of the algorithms the vector space dimension of the quotient ring has to be known.

In the higher dimensional case, the Hilbert function can be used instead of the vector-space dimension to obtain results of the same type. In this paper we sketch how the knowledge of the Hilbert function can be used in the computation of a Gröbner basis.

Of course, to compute the Hilbert function usually one needs a Gröbner basis, hence the main field of application will be the change of ordering. Remark that quite often a Gröbner basis with respect to an uninteresting ordering is known in advance; this is for example true for the implicitization problems, for the inverse kinematics problems, etc.; in these case the problem is to eleminate some variables, and the original basis is Gröbner in an ordering that eliminates the other variables.

Another case is when the ideal is homogeneous and is a complete intersection; even if we do not know this fact, we can perform the computation "as if" the ideal is a complete intersection, and test from the dimension of the result that ideal is really a complete intersection, hence the computed basis, that is Gröbner under this hypothesis, is proved to be a Gröbner basis at the end.

Other applications are also possible.

2.1 Homogenizing an ideal

In this subsection we recall some well-known easy results, see [MM2].

Let I be an ideal of $k[X] = k[x_1, \ldots, x_n]$; we associate to I the homogeneous ideal I_h of $k[X] = k[x_0, \ldots, x_n]$ obtained homogenizing with the variable x_0 every element of I. This is obtained as follows: the homogenization of $g \in k[X]$ is the unique homogeneous polynomial $\bar{g} \in k[X]$ of the same degree such that $\bar{g}(1, x_1, \ldots, x_n) = g$, and is obtained multiplying every monomial m of g by $x_0^{\deg g - \deg m}$.

If $G = g_1, \ldots, g_m$ is a set of generators of I, the ideal $\hat{I}$ generated by $\bar{g}_1, \ldots, \bar{g}_m$ is smaller than I, and $\bar{I}$ is obtained saturating $\hat{I}$ with respect to x_0.

If $k[X]$ has a term-ordering, a degree-compatible term-ordering is defined naturally on $k[X]$, that compares two terms of the same degree stripping the power of x_0 from them. In this term-ordering, homogenizing a polynomial the monomial remain in the same order. If the term-ordering on $k[X]$ is degree-compatible then homogenizing a polynomial the leading term never contains x_0. This term-ordering is caled the *homogenization* of the term-ordering, and will be the default in this situation.

There are three algorithms for computing a set of generators of the homogenization of an ideal:

a) compute a Gröbner basis G of I w.r.t. a degree-compatible term-ordering, and let G be obtained homogenizing the elements of G. It is immediate to prove that the result is a Gröbner basis of the homogenized ideal $\bar{I}$ with respect to the homogenization of the term-ordering.

b) homogenize a set of generators, compute a Gröbner basis, and divide every polynomial by the highest power possible of x_0. The result is a redundant Gröbner basis.

c) homogenize a set of generators, compute a Gröbner basis, dividing every polynomial by x_0 whenever possible during the algorithm.

(Remark that homogenizing a Gröbner basis with respect to an ordering that is not degree-compatible migh not be sufficient; consider $(x - t^2, y - t^2)$ with lexicographic ordering, $x > y > t$)

If the term-ordering is degree-compatible and x_0 is the smallest variable, the first and third algorithm coincide: comparing two monomials of different degrees or comparing them after multiplying the one of smaller degree by a suitable power of x_0 gives the same result. Hence every correctly sorted polynomial remains correctly sorted homogenizing it, and a polynomial is divisible by x_0 iff x_0 divides the leading term. Hence the Buchberger algorith is exactly the same in both cases, apart from some completely useless x_0 in the trailing terms.

2.2 Basic facts on Hilbert functions

The main theorem that we will use is the following:

1 THEOREM. *Let $I \subseteq k[X] = A$ be an ideal, and let A be endowed with a term-ordering that is degree-compatible. Let $G = (g_1, \ldots, g_m)$ be a set of elements of I, let $\hat{I}$ be the initial ideal of I and let I' be the monomial ideal generated by the $Lt(g_i)$. Let $h_{\hat{I}}$, $h_{I'}$ be the corresponding Hilbert functions. Then:*
a) $h_{\hat{I}}(n) \leq h_{I'}(n)$ for every n
b) if $h_{\hat{I}} = h_{I'}$ then G is a Gröbner basis.

PROOF: We have that $\hat{I} \subseteq I'$, (and this proves a) and the two are equal iff G is a Gröbner basis: assume that this is false, and let α an element of I' not contained in $\hat{I}$; if n is the degree of α then in degree n, $h_{\hat{I}}$ and $h_{I'}$ are different.

We will use moreover the following lemma:

2 LEMMA. *Let I', I'' be the initial ideals of the ideal I w.r.t. two different degree-compatible term-orderings; then $h_{I'} = h_{I''}$.*

PROOF: In $k[X]/I$ consider a filtration F_n induced by the degree filtration in $k[X]$ (F_n is the image of the elements of degree n). In $k[X]$ the degree of an element is equal to the degree of its leading term; hence F_n is equal to the dimension of the space of polynomials in $k[X]$ of degree n modulo the polynomials whose leading term is of degree n. Hence the k-dimension of F_n is equal to $\sum_{i=0}^{n} h_{I'}(i) = \sum_{i=0}^{n} h_{I''}(i)$.

Remark that the equality test between two Hilbert functions $h = h'$ is computed in finite terms through the corresponding generating function, $\sum h(n)T^n$, that is a rational function: indeed any algorithm for the computation of the Hilbert function explicitly gives this rational function form, from which the value of the Hilbert function at a specific number n is computed through Taylor expansion (or directly through a formula).

Let f, g be polynomials, we say that *the degree drops* in the sum $f+g$ if $f+g \neq 0$ and the degree of $f+g$ is lower than the degree of f (thus implying that the degrees of f and g are equal.

The following theorems hold, see [MM4]:

3 THEOREM. *Let I be an ideal, G a Gröbner basis of I with respect to a degree-compatible term-ordering, and consider the Buchberger algorithm starting from G to compute the Gröbner basis G' with respect to another degree-compatible term-ordering and with normal selection strategy. If during the algorithm in a reduction the degree drops, then the reduction will give eventually result zero*

4 COROLLARY. *In the above Buchberger algorithm new elements appear in increasing degrees.*

PROOF: The degree drops in a sum $f+g$ if and only if considering the homogenized polynomials $\bar{f}$, $\bar{g}$, the sum $\bar{f}+\bar{g}$ is a multiple of the homogenization variable.

If an element h is multiple of a variable x_0 and no leading term of a set of generators contains x_0, then the result of the reduction is again multiple of x_0.

Consider the homogenization $\bar{G}$ of G; it is a Gröbner basis of the homogenized ideal I. The Hilbert-Poincaré series H_I of InI and $H_{\bar{I}}$ of $\bar{I}$ are related by $(1-T)H_{\bar{I}} = H_I$. We have remarked that H_I does not change if we choose another degree-compatible term-ordering.

Consider now the Buchberger algorithm of G and $\bar{G}$; they run in parallel until a new basis element appears that has the homogenization variable x_0 in its leading term in the second one (and this corresonds in the first one in a new basis element of lower degree than its S-polynomial [indeed, of the apparent degree of the S-polynomial - we need a definition etc.]). But we are following the normal selection strategy, hence the elements of the basis in the homogeneous case appear in increasing ordering, and this means that at that moment the coefficients of $H_{\bar{I}}$ in the degrees lower than the current degree is settled, hence this has to be the same for H_I because of the relation remarked above. This means that no new element of lower degree will appear in the Buchberger algorithm for G, and that the simplification that has dropped the degree will eventually give result 0.

2.3 Change of ordering algorithms

We have several algorithms, that can be applied in different situations.

Homogeneous ideals

The first algorithm can be applied to homogeneous ideals, and in this case the organizing properties of the Hilbert function appear in its full form.

Assume that I is an homogeneous ideal, g_i a set of generators, and assume that we know the Hilbert function $h = h_I$, probably having previously computed a Gröbner basis with respect to a term-ordering diferent from the current one.

Let $\alpha_i = Lt(g_i)$, and compute $h' = h_{\alpha_i}$.

If $h = h'$, then G is a Gröbner basis, otherwise let n be such that $h(j) = h'(j)$ for $j < n$ and $h'(n) = h(n) + k$. This means that:

a) G contains all the elements of degree $< n$ of a Gröbner basis

b) a Gröbner basis contains k further elements in degree n.

With this information we can perform the Buchberger algorithm, but with the following modification:

a) all critical pairs of degree $< n$ are useless (the degree of a critical pair being the degree of the lcm of the two leading terms)

b) precisely k critical pairs of degree n are useful.

When we have found through the Buchberger algorithm k useful critical pairs of degree n, giving rise to new elements $g_{n+1}, \ldots, g_{n+k}$ in the Gröbner basis, we recompute the new Hilbert funcion H', and proceed in the algorithm.

The algorithm is often good since usually in Buchberger algorithm the useful pairs in each degree are the ones that are computed first, and at the end of the algorithm for each degree a good batch of useless pairs is computed; with our algorithm these can be avoided.

Non homogeneous ideals, degree-compatible ordering

Assume that we have a non-homogeneous ideal; the algorithm can be performed as above, but we no longer proceed degree by degree. We can proceed as above, but the degree-wise organization is lost. Hence we expect worse behaviour.

Non homogeneous ideals, starting from a Gröbner basis

If we already know a Gröbner basis w.r.t. a degree-compatible term-ordering, and we want a Gröbner basis w.r.t. a different degre-compatible term-ordering, the best thing may be to perform the Buchberger algorithm starting from the existing Gröbner basis; in this way, the algorithm proceeds degree-wise, as remarked in Corollary 4, and we recover the degree-organization.

There are two further remarks that simplify the algorithm: from Theorem 3, we can abandon a simplification if the degree drops; and moreover, if the previous Gröbner basis was explicitly computed, we know several syzygies, and we can use them as explained in [MMT] to avoid a bunch of useless pairs.

Non-homogeneous term-orderings

Assume that we have a Gröbner basis w.r.t. a degree-compatible term-ordering, and that we want to find a Gröbner basis w.r.t. a term-ordering that is not degree-compatible (this is indeed the usual situation: we want to pass from Degree-Reverse-Lex to Lex). If the ideal is homogeneous, then one can change the term-ordering: indeed, adding the degree, the Gröbner basis does not change. If it is not homogeneous, we can homogenize the generators and de-homogenize the result.

Modular algorithms

In all the cases, if the ground field k is the rational field, we can take advantage of modular computations as follows:

a) compute a Gröbner basis mod p with one of the algorithms above, with respect to a prime p: let it be G_p.

b) compute the Hilbert function of G_p; if it does not coincide with the Hilbert function of the ideal (that we are assuming to know), then p is unlucky, and we have to change p and repeat from a).

c) repeat the computation on $\mathbf{Z}$, discarding the pairs that are useless mod p (a *trace-lifting algorithm* in the terminology of [Tr]). If the computation on $\mathbf{Z}$ diverges from the computation mod p (some leading term in $\mathbf{Z}$ is different from what was expected) then p was unlucky, and we have to choose another prime p and repeat from a). Otherwise the result is a Gröbner basis.

The correctness of the algorithm comes from the fact that the Hilbert function of the result coincides with the Hilbert function mod p, (the leading terms coincide), and this in turn coincides with the Hilbert function of the ideal. Since the result is composed of elements of the ideal by construction, Theorem 1 can be applied, and the result is a Gröbner basis.

We make some incidental remarks:

a prime may be lucky for a term-ordering, unlucky for another, and this may not be apparent in the Hilbert function.

Example: let g_i be any set of polynomials, add new variables t_i, and consider $g_i - t_i^{d_i}$, for suitable d_i (that may be the degree of g_i if we want homogeneous examples). If the t_i are larger variables, then this is a Gröbner basis, if they are smaller then it is the g_i that decide the luckyness.

2.4 Implementation and performance

The algorithm has been implemented in AlPi; it has been tested on some examples. For some of these the original basis is a Gröbner basis for an uninteresting ordering, for others we are interested in a Lex basis, and we try to compute it through a change of ordering.

In this implementation, an incremental form of the algorithm for the computation of the Hilbert-Poincaré series has been used, that assumes that we know the Hilbert-Poincaré series of a staircase, and we want to add one generator. It consists in performing one step of the [BS] algorithm (one of the two branches of the computation is already known) and continuing with the algorithm of [BCRT].

In every case tested, the computation using the Hilbert-Poincaré series is an improvement with respect of the Buchberger algorithm, since the additional cost of computing the Hilbert-Poincaré series is negligeable and many useless pairs are discarded; however sometimes the direct computation is better, since the overhead of considering an homogenized ideal is higher than the improvement in Buchberger algorithm. In some especially good case the improvement has been of a factor of 100 and more.

More precise timings will be given in [GT]

3 Application: dynamical determination of the term ordering

Here we sketch rapidly those parts of [Ca] that use Hilbert-Poincaré series.

In computing a Gröbner basis, the choice of the term ordering is often free, at least partly. One wants to determine the term-ordering dynamically in a way that forces the algorithm to converge more rapidly.

To simplify the exposition we assume that we want a degree-compatible term-ordering (in general we will fix a weight for the variables, and consider weight-compatible term orderings: the weight is heuristically chosen in a way that makes the polynomials as homogeneous as possible). With this assumption the Hilbert-Poincaré series of the associated graded ideal is fixed (but unknown). The Hilbert function determined by the leading terms of the elements of the basis (in any degree-compatible term-ordering) is larger than the Hilbert function of the associated graded ideal (Theorem 1), and we want to determine the term ordering in such a way that this Hilbert function is as small as possible. The choice of a leading term in a polynomial determines partially the term-ordering, (it determines a convex polyhedron in the set of linear maps from $\mathbf{R}^n$ to $\mathbf{R}$ if we have n variables, since choosing between two terms which one is the larger chooses one out of two half spaces), hence we have to make this choice coherently and incrementally (adding the elements one at a time) in a way that minimizes the Hilbert function.

Different strategies are possible, but the greedy strategy is the obvious one, and apparently it is a reasonable one too.

This algorithm has been implemented, but the experimentation has not been sufficient up to now to decide if the algorithm is convenient; the combinatorial acceleration of the Buchberger algorithm is evident, but this happens sometimes at the expense of the coefficient growth, and the cost of computing repeatedly the Hilbert-Poincaré series is not negligeable.

References

[BCR] Bigatti, A.M., Caboara, M., Robbiano, L., *On the computation of Hilbert-Poincaré series*, AAECC Journal, vol. 2, 1991

[BCRT] Bigatti, A.M., Conti, P., Robbiano, L., Traverso, C., *A "Divide and conquer" algorithm for Hilbert-Poincaré series, multiplicity and dimension of monomial ideals*, AAECC-10 (1993)

[BS] Bayer-Stillman, *Computation of Hilbert functions*, J. of Symbolic Comp. vol. 14 (1992)

[BGK] Boege, W., Gebauer, R., Kredel, H. *Some examples for solving systems of algebraic equations by calculating Gröbner bases*, J. of Symbolic Comp. (1986)

[Ca] Caboara, M. *A dynamic Algorithm for Gröbner basis computation* ISSAC 1993

[FGLM] Faugere, J.C., Gianni, P., Lazard, D., Mora, T. *Efficient computation of zero-dimensional Gröbner bases by change of ordering*, preprint 1989

[GM] Gianni, P., Mora, T., *Algebraic solutions of systems of polynomial equations using Gröbner bases*, AAECC5, LNCS 356 (1989)

[GT] Gianni, P., Traverso, C., *Hilbert function and Gröbner bases*, preprint 1993

[GN] Giovini, A., Niesi, G., *CoCoA: a user-friendly system for commutative algebra*. DISCO-90 Proceedings, LNCS 429, Springer Verlag (1990)

[Ho] Hollman, J., *On the computation of the Hilbert series*, Latin 92, Saõ Paulo. LNCS 583. Springer Verlag (1992)

[KP] Kondrat'eva, M.V., Pankrat'ev, E.V., *A recursive algorithm for the computation of Hilbert polynomial*, EUROCAL 87, LNCS 387, Springer Verlag (1987)

[MM1] Möller, M., Mora, T., *The computation of the Hilbert function*, Proc.EUROCAL 83. LNCS162 (1983) 157-167

[MM2] Möller, M., Mora, T., *New constructive methods in classical ideal theory*, J. Algebra vol. 100 (1986)

[MM3] Möller, M., Mora, T., *Computational aspects of reduction strategies to construct resolutions of monomial ideals*, AAECC2, LNCS 228 (1986), 182-197

[MM4] Möller, M., Mora, T., *Upper and lower bounds for the degree of Gröbner bases*. Proc. EUROSAM 84, LNCS 174(1984)

[MMM] Möller, M., Mora, T., Marinari, M. *Gröbner bases of ideals given by dual bases*, ISSAC 1991

[MMT] Mora, Moeller, Traverso, *Gröbner basis computing using syzygies*, ISSAC 1992

[MT] Mora, Traverso, *Linear Gröbner methods and "natural" representations of algebraic numbers*, in preparation (1991-93)

[TD] Traverso, C., Donati, L., *Experimenting the Gröbner basis algorithm with the AlPi system*. ISSAC 89 Proceedings, A. C. M. (1989)

[Tr] Traverso, C., *Gröbner trace algorithms*, ISSAC 1988