Kobayashi, H., Fujise, T. and Furukawa, A. (1988). Solving systems of algebraic equations by general elimination method. *J. Symb. Comp.* **5**, 303–320.

Kobayashi, H., Moritsugu, S. and Hogan, R.W. (1988). On solving systems of algebraic equations. *Symbolic and Algebraic Computation, International Symposium ISSAC '88* (P. Gianni, ed.), Lect. Notes in Comp. Sc. **358**, 139–149.

Lakshman, Y. N. (1990a). On the complexity of computing a Gröbner basis for the radical of a zero dimensional ideal. *Proc. STOCS'90.*

Lakshman, Y. N. (1990b). A single exponential bound on the complexity of computing Gröbner bases of zero dimensional ideals. *Proceedings of MEGA '90.* Progress in Math., Birkhauser *(to appear).*

Lakshman, Y. N. and Lazard, D. (1990). On the complexity of zero–dimensional algebraic systems. *Proceedings of MEGA '90.* Progress in Math., Birkhauser *(to appear).*

Langemyr, L. (1990). Algorithms for a multiple algebraic extension. *Proceedings of MEGA '90.* Progress in Math., Birkhauser *(to appear).*

Lazard, D. (1981). Résolution des systèmes d'équations algébriques. *Theor. Comp. Sci.* **15**, 77–110.

Lazard, D. (1983). Gröbner bases, gaussian elimination and resolution of systems of algebraic equations. Proc. EUROCAL 83. Lect. Notes in Comp. Sci. **162**, Springer, 146–157.

Lazard, D. (1990). A new method for solving algebraic systems of positive dimension. Technical Report LITP 89–77 *To appear in Discr. App. Math.*

Möller, H.M. (1989). To appear.

Wilkinson, J.H. (1959). The Evaluation of the Zeros of Ill–conditioned polynomials. *Num. Math.* **1**, 150–166, 167–180.

Wu Wen–Tsün (1987). A Zero Structure Theorem for Polynomial Equation Solving. *Math.–Mechanization Research Preprints* **1**, 2–12, Academica Sinica, Beijing.

Davenport, J.H., Siret, Y. and Tournier, E. (1986). Calcul Formel, Systèmes et algorithmes de manipulation algébriques. Masson. English translation : Computer algebra. Academic Press, 1988.

Della Dora, J., Dicrescenzo, C. and Duval, D. (1985). About a new method for computing in algebraic number fields. *Proc. EUROCAL 85, Vol. 2.* Lect. Notes in Comp. Sci. **204**, Springer Verlag, 289–290.

Dicrescenzo, C. and Duval, D. (1985). Algebraic computations on algebraic numbers. Computers and Computing (ed. Chenin et al.). Masson and Wiley, 54–61.

Dicrescenzo, C. and Duval, D. (1988). Algebraic extensions and algebraic closure in SCRATCHPAD II. *Symbolic and Algebraic Computation, International Symposium ISSAC '88* (P. Gianni, ed.), Lect. Notes in Comp. Sc. **358**, 440–455.

Duval, D. (1987). Diverses questions relatives au calcul formel avec des nombres algébriques. *Thèse d'Etat,* Grenoble.

Faugère, J.C., Gianni, P., Lazard, D. and Mora, T. (1988). Efficient computation of zero–dimensional Gröbner bases by change of ordering. Technical Report LITP 89–52 *submitted to J. Symb. Comp.*

Gianni, P. (1987). Properties of Gröbner bases under specializations. *European Conference on Computer Algebra, Leipzig, GDR, 1987.* (J.H. Davenport, ed.) Lect. Notes in Comp. Sc. **378**, 293–297.

Gianni, P. and Mora T. (1987). Algebraic solution of systems of polynomial equations using Gröbner bases. *Applied Algebra, Algebraic Algorithms and Error–Correcting Codes (AAECC 5), Menorca, Spain, 1987.* Lect. Notes in Comp. Sc. **356**, 247–257.

Gianni, P., Trager, B. and Zacharias, G. (1988). Gröbner bases and primary decomposition of polynomial ideals. *J. Symb. Comp.* **6**, 149–167.

Kalkbrenner, M. (1987). Solving systems of algebraic equations by using Gröbner bases. *European Conference on Computer Algebra, Leipzig, GDR, 1987.* (J.H. Davenport, ed.) Lect. Notes in Comp. Sc. **378**, 282–292.

end.

Proof of the algorithm : This algorithm replaces x_1 by a linear combination of it and the other variables and expresses the variables as a polynomial in this new variable. It is clear that it works if the *until* test eventually succeeds and if the resultant is not 0. If the resultant is 0, then f and r have a common zero in x_i for each value of z ; substituting back we get that f_0 and p have an infinity of common zeros, which is not possible, because f_0 does not depends on x_i and p is monic in x_i. Thus the resultant is never 0. If the last non constant subresultant has a degree greater than 1, or if its leading coefficient has a non trivial GCD with the resultant r, then for some root of r, the GCD of f and q is of degree at least 2. This may be the case for an infinity of values of λ only if f_0 and p have a common non simple zero. This is in contradiction with the reduced hypothesis on T.

Remark 6 — *In pratice, it is much better to factor f_0 at each step and to split the triangular set being constructed : Lower is the degree of f_0, faster is the computation and smaller are the coefficients which appear. For the same reason, if the leading coefficient of the subresultant of degree 1 has a common factor with the resultant, this may be used for splitting the problem (for example with D5). The same decomposition comes from the factorization of the resulting f_0, but an earlier decomposition leads to lower values of λ.*

Proposition 7 — *The number of arithmetic operations of algorithm Generic is polynomial in the degree D of the input triangular set.*

Clearly, at each step, the degree of f_0 is less than D. Thus the result follows from the fact that resultant computations and factorizations are polynomial, when we remark that the number of iterations on λ is bounded by the number of lines passing through 2 points among D.

Remark 7 — *If we take into account the growth of coefficients, it appears that Generic is exponential in n. It seems that it is unavoidable : we are faced with the difficulty of computing a primitive element of a field extension.*

References.

Davenport, J.H. (1987). Looking at a set of equations. Thechnical report 87–06, University of Bath.

only univariate polynomials with integer coefficients. This is easy from a triangular system.

The following algorithm starts with a triangular system and returns a list of univariate polynomials $f_0, \ldots, f_n$ such that the set of the evaluations of $(f_1, \ldots, f_n)$ at the roots of f_0 is exactly the set of the common zeros of the triangular set. This algorithm is probably not new, but has to be restated in terms of triangular sets.

Procedure 7 Generic

Input : $T = (T_1, \ldots, T_n)$: a triangular reduced set in the variables $x_1, \ldots, x_n$;

Output : $f = (f_0, \ldots, f_n)$: a list of univariate polynomials as specified above ;

Begin

$f_0 := subst(z,\ x_1,\ T_1)$;

$f_1 := z$;

for $i = 2$ *to* n *do*

$p := subst([f_1, \ldots, f_{i-1}],\ [x_1, \ldots, x_{i-1}],\ T_i)$;

if $degree(p,\ x_i) = 1$ *then*

$f_i := p - x_i$ $\{p$ is monic$\}$;

else

for λ *in* 1, -1, 2, -2, ... *repeat*

$f := subst(z + \lambda x_i),\ z,\ f_0)$;

$q := subst(z + (\lambda x_i),\ z,\ p)$;

$r := resultant(f,\ q,\ x_i)$

until the last non constant (in x_i) subresultant is of degree 1 and the coefficient of x_i in it has a constant GCD with r ;

$s :=$ this subresultant ;

$f_0 := r$;

$f_i :=$*subst*$(0,\ x_i,\ s)/$ *coeff*$(x_i,\ s)$ mod r {this quotient is congruent to a polynomial modulo r} ;

for $j = 1$ *to* $i - 1$ *do* $f_j := subst(z + \lambda f_i,\ z,\ f_j)$;

$p := \mathit{first}(H)$;

$H := \mathit{rest}(H)$;

$q := \mathit{Leading}(p,\ x_i)$;

$q := \mathit{Inverse}(q,\ \mathit{mod})$;

until $q \neq 0$;

$p := \mathit{Reduce}(p * q,\ \mathit{mod})$;

$\mathit{mod} := \mathit{cons}(p,\ \mathit{mod})$

return(mod)

end.

Both algorithms are directly based on theorem 1 : at each step of the loop on i, the polynomials in $x - 1, \ldots, x_i$ modulo *mod* form a field (or a D5–field) and Gianni–Kalkbrenner theorem applies.

In each step of the loop on the moduli these algorithms need only one inversion of the leading coefficient, one product by the inverse and one factorization or splitting for each element of G which is visited. Let D be the number of monomials which are irreducible by the input Gröbner base ; the sum of the degrees of the triangular sets (see definition 1) in the triangular family which is returned is at most D (equality if all solutions are simple). Thus the time needed by *Lextriangular* or *D5lextriangular* is at most the length of G times the time needed for one inversion and one factorization or product of a polynomial by a constant. The length of G is at most nD (see Faugère et al, 1989), where n is the number of variables ; thus the above algorithms are polynomial in D and n provided that the needed arithmetic operations are polynomial in the degrees of the triangular set of moduli and of the polynomial to be inverted or factorized. This has been done in Langemyr (1990).

8 Numeric resolution.

We have seen above that numeric solutions may be easily obtained from triangular sets by solving successive univariate polynomials. In fact, this is not so easy because, after the first step, the coefficients are not integers. This is a problem on some computer algebra systems which only provide a fonction for computing the roots for integer coefficients. More important, finding the roots of a polynomial is a very unstable problem (see Wilkinson, 1959 or Davenport, Siret, Tournier, 1986). Thus it is necessary to solve *repeat*

$p :=$ *first*(H) ;

$H :=$ *rest*(H) ;

$q :=$ *Leading*$(p,\ x_i)$;

$q :=$ *Inverse*$(q,\ mod)$;

until $q \neq 0$;

for q in Factor(p, mod) do

$T := cons(cons(q,\ mod),\ T)$

return(T)

end.

The D5 variant of *Lextriangular* differs only by suppressing the loop on U which is managed by D5 and by replacing the loop on the factors by the reduction of $q * p$ by *mod*.

Procedure 6 D5lextriangular

Input : G : a Gröbner base of a zero–dimensional ideal of polynomials in x_1, $\ldots$, x_n, sorted by increasing leading monomials, for the lexicographical ordering such that $x_1 < \cdots < x_n$;

Output : T : a triangular family equivalent with G ;

Subfonctions :

Reduce$(p,\ mod)$: reduces p modulo *mod* ;

Inverse$(p,\ mod)$: returns the inverse of p modulo *mod* or 0 if p reduces to 0 modulo *mod* ;

Leading$(p,\ x)$: returns the leading coefficient of p as a univariate polynomial in x ;

Begin

mod $:= [\,]$;

for $i := 1$ *to* n *do*

$H :=$ the sublist of the elements of G which depends on x_i but not on $x_{i+1}, \ldots, x_n$;

repeat

of G which depend only on $x_1, \ldots, x_k$. Then the first element of G which is not mapped to 0 is the first one which depends only on $x_1, \ldots, x_{k+1}$ with a leading term (as a polynomial in x_{k+1}) not mapped to 0. Moreover, the image by f of this polynomial is the GCD of the images by f of all the elements of G depending only on $x_1, \ldots, x_{k+1}$.

The easiest application of this theorem is for numerically solving systems, i.e. when the target field is an algebraic closure of K. But this results works with any target field and make the computation of an equivalent triangular family very easy and possible without any new computation of a Gröbner base. As before, we give two versions of the corresponding algorithm ; the first one use factorizations and works with algebraic extensions as target fields ; the second one, using D5, works without any factorization.

Procedure 5 Lextriangular

Input : G : a Gröbner base of a zero–dimensional ideal of polynomials in x_1, $\ldots$, x_n, sorted by increasing leading monomials, for the lexicographical ordering such that $x_1 < \cdots < x_n$;

Output : T : a triangular family equivalent with G ;

Subfonctions :

Reduce(p, *mod*) : reduces p modulo *mod* ;

Inverse(p, *mod*) : returns the inverse of p modulo *mod* ;

Leading(p, x) : returns the leading coefficient of p as a univariate polynomial in x ;

Factor (p, mod) : returns the list of ireducible monic factors of p modulo *mod*

Begin

$T := [[\,]]$;

for $i := 1$ *to* n *do*

$H :=$ the sublist of the elements of G which depends on x_i but not on $x_{i+1}, \ldots, x_n$;

$U := T$; $T := [\,]$;

for mod in U do

end.

The matrix constructed in *ElimPolChar* is that of the product by x_m in the algebra of the polynomials modulo the ideal generated by G. The result of *ElimPolMin* is the minimal polynomial of 1 for this endomorphism ; thus it divides the result of *ElimPolChar*. Moreover, we have the following.

Proposition 6 — *The multiplicity of a root x of the result of ElimPolChar is the sum of the multiplicities of the solutions of G (viewed as a system of equations) with x as x_m–component.*

This was essentially proved in (Lazard, 1981). Here is a direct proof : We may suppose w.l.o.g. that $m = 1$. The ring $A = K[x_1, \ldots, x_n]/G$ is a product of local artinian rings with maximal ideal (over an algebraic closure of K) of the form $(x_1 - \alpha_1, \ldots, x_n - \alpha_n)$ The multiplicity of the solution is the multiplicity of the corresponding local ring, that is its length or its dimension as a vector space. If $x_1 - \alpha_1$ is in the maximal ideal of a local artinian ring, some power of it is zero ; i.e. there exists a k such that $(x_1 - \alpha_1)^k$ is zero in this local ring. This means that the product of the local rings corresponding to the solutions with first component x_1 is exactly the union of the kernels of the endomorphisms "product by a power of $x_1 - \alpha_1$", that is the multiplicity of x_1 as an eigenvalue of the endomorphism "product by x_1" in A.

This function *ElimPolChar* is very easy to implement but it needs $O(D^4)$ field operations instead of $O(D^3)$ for *ElimPolMin*. On the other hand, on most examples, the factors given by the square–free decomposition of the result of *ElimPolChar* suffice to avoid further splittings by D5. For example, on system (7) they give all the decomposition needed to find the solution (11).

7 From lexicographical Gröbner base to triangular sets.

For computing a triangular family from a lexicographical Gröbner base, the main tool is the following theorem by Gianni (1987) and Kalkbrenner (1987).

Theorem 1 — *Let G be a Gröbner base of a zero–dimensional ideal in $K[x_1, \ldots, x_n]$, for the lexicographical ordering such that $x_1 < x_2 < \cdots < x_n$; suppose that G is sorted by increasing leading monomial. Let f be a ring homomorphism of $K[x_1, \ldots, x_k]$ into a field which maps to 0 the elements* *monom* := 1 *; nfm* := *NormalForm(G, monom) ;*

lnf := [ ] *; listmonom* := [ ] *;*

while nfm is not a linear combination over K of elements in *lnf do*

lnf := *cons(nfm, lnf) ;*

listmonom := *cons(monom, listmonom) ;*

monom := $x_m * monom$ *;*

nfm := *NormalForm(G, monom)*

{We have $nfm = \sum_{e \in lnf} a_e e$ with $a_e \in K$}

return monom $- \sum_{e \in lnf} a_e$ *mon* with *mon* beeing the element of *listmonom* corresponding to e in *lnf*

end.

As *NormalForm* is a linear map, it is clear that this algorithm returns the relation of least degree (modulo G) between $1, x_m, x_m^2, \ldots$ It is clear that the linear algebra part needs $O(D^3)$ operations. We refer to (Faugère et al., 1989), for an implementation of *ElimPolMin* with $O(D^3)$ field operations and for a complexity analysis ; in fact, *ElimPolMin* is the beginning of *NewBase* described there ; in the generic case were all solutions of the system are simple and have distinct x_m–component, *ElimPolMin* and *NewBase* become equivalent.

The second implementation of *ElimPol* is less efficent ; howewer it may be easier to implement in some Computer Algebra systems. Moreover it gives information on multiplicities ; in most cases, these informations permit us to avoid D5–splitting in *D5triangular* ; thus an implementation of it is possible without D5, which works well on most (not too singular) problems.

Procedure 4 ElimPolChar(G, K, m, n)

Input : G : a Gröbner base of an ideal in $K[x_m, \ldots, x_n]$ *;*

Output : an univariate polynomial in $K[x_m]$ of degree D *;*

Begin

For each monomial *mon* which is in normal form relatively to G, let $NormalForm(x_m * mon, G) = \sum a_{mon,m} * m$ (sum over the monomials in normal form) *;*

return the characterisic polynomial of the matrix $(a_{mon,m})$

P := *D5elimpol(G, mod, m, n) ;*

mod := *endcons(P, mod) ;*

G := *D5gröbner(G, mod,* $m+1$*, n) ;*

D5triangular(G, mod, $m+1$*, n) ;*

end.

This algorithm is exactly the same as *Triangular*, except that the loop does not appear explicitly but is implicitly created by D5.

Remark 4 — *Both Triangular and D5triangular may return not reduced triangular ideals. This may be avoided by a post computation of square–free decompositions or by continuing recursive calls until* $m = n$ *: D5 system considers only square–free* moduli *(here and in the following, we call moduli the members of a triangular system when computations are done modulo this system) ; if a modulus is not square–free, it is automatically split by D5 by square free decomposition ; on the other hand, in Triangular, the loop on the factors removes also the multiplicities.*

Remark 5 *The algorithm D5gröbner is exactly the standard Buchberger's one managed by D5. Thus it may split the moduli. It would be possible to replace it by an algorithm without splittings : it is easy to define Gröbner bases over a reduced artinean ring ; the computation would be essentially the same as the classical Buchberger's algorithm on the ideal generated by append(G, mod) with the following ordering : lexicographical on* $x_1, \ldots, x_{m-1}$ *and considering the powers of* $x_1, \ldots, x_{m-1}$ *only when the powers of* $x_m, \ldots, x_n$ *are the same for the monomials to be compared. However, splitted problems are generally much easier than initial problem and it is probably more efficient to use D5gröbner.*

We present now two ways for implementing *ElimPol.* The first one is classical and appears at least in (Kobayashi, Moritsugu and Hogan, 1988).

Procedure 3 ElimPolMin(G, K, m, n)

Input : G : a Gröbner base of an ideal in $K[x_m, \ldots, x_n]$ *;*

Output : the minimal univariate polynomial in $K[x_m]$ which is in the ideal generated by *G ;*

Begin

Remark 2 — *Very similar algorithms have been suggested in (Lazard, 1981) and described in (Kobayashi, Moritsugu and Hogan, 1988). However we do not know of any implementation. In fact, this is not easy on most computer algebra systems : we need functions parametrized by a field. Moreover, even if such a parametrization is available as in SCRATCHPAD, we need also a good factorization algorithm on towers of simple algebraic extensions.*

Remark 3 — *Many Gröbner bases over field extensions are computed during this algorithm. This may appear to be prohibitive. In fact the polynomials in these bases are generally of very low degree and it seems that in most cases, the total cost of these Gröbner bases computations is dominated by computation of the Gröbner base of the initial ideal. In any case, we give in the following sections a method which avoids multiple Gröbner bases computations.*

Before describing *ElimPol*, we give the D5 variant of *Triangular*.

Procedure 2 D5triangular(G, mod, m, n)

Input :

mod : a triangular family of $m-1$ polynomials in $x_1, \ldots, x_{m-1}$ which is empty for the main call *;*

m *:* current index ($m = 1$ for the main call) *;*

n *:* the number of variables *;*

G *:* the Gröbner bases of polynomials in $x_m, \ldots, x_n$ over the D5–field of the polynomials in $x_1, \ldots, x_{m-1}$ modulo *mod ;*

Output : a triangular family equivalent to G *;*

Auxiliary functions :

D5elimpol(G, mod, m, n) : returns a polynomial in x_m over the D5–field of the polynomials in $x_1, \ldots, x_{m-1}$ modulo *mod ;*

D5gröbner(G, mod, m, n) : returns the Gröbner bases of the ideal generated by G over the D5–field of the polynomials in $x_1, \ldots, x_{m-1}$ modulo *mod ;*

Begin

If G is triangular *then return append(mod, G) ;*

n : index of the last variable = initial number of variables

G : Gröbner base of an ideal in $K[X_m, \ldots, X_n]$*;*

Auxiliary functions :

Factor(P, K) : factorize the polynomial P over the field K *;*

ElimPol(G, K, m, n) : returns a polynomial in $K[X_n]$ the roots of which are the values of x_n for the common zeros of G (algorithm given below) *;*

Gröbner(G, K, m, n) : returns the Gröbner bases of the ideal generated by G in $K[X_m, \ldots, X_n]$ *;*

Output : a triangular family equivalent to G *;*

Begin

if G is triangular *then return* $[G]$ *;*

$P :=$ *ElimPol*(G, K, m, n) *;*

family $:= [\,]$ *;*

for Q in Factor(P, K) do

$L := K[x_m]\,/Q$ *;*

$H :=$ *Gröbner(G, L,* $m+1$*, n) ;*

$T :=$ *Triangular*$(H,\ L,\ m+1,\ n)$ *;*

for x in T replace x by cons(Q, x) ;

family $:=$ *append(T, family)*

return family

end.

Proof of the algorithm : It follows immediately from the definitions that the roots (in an algebraic closure) of the first elements of a triangular system are the values of the first variable in the solutions. The recursion stop when $m = n$ in the worst case, but, in general, there is only one solution with a given value of $x_1, \ldots, x_k$ for some low value of k ; in this case, the algorithm stops for $m = k + 1$, the base being triangular (all polynomials in it are linear).

3 first triangular sets of (8) and

$$\begin{array}{l}(a^{15}+122a^{10}-122a^5-1,\\ \quad 275b^2+(16a^{11}+1958a^6-1149a)b+42a^{12}+5126a^7-4893a^2,\\ \quad 1375c+(11a^{10}+1353a^5+11)b+4a^{11}+517a^6+3604a,\\ \quad 275d-8a^{11}-979a^6+712a,\\ \quad 1375e+(-11a^{10}-1353a^5+1364)b+36a^{11}+4378a^6-5789a)\end{array} \tag{12}$$

Remark 1 — *It is important to note that the non unicity of the family of reduced triangular sets equivalent to a given system is not a drawback, for the reason that passing from a family to an other one is rather easy. However, an unique minimal equivalent reduced family does not alway exist. For example*

$$((x_1,x_2),(x_1,x_2+1),(x_1+1,x_2))$$

may be combined by part ii) of the last proposition in

$$((x_1,x_2^2+x_2),(x_1+1,x_2))$$

or

$$((x_1^2+x_1,x_2),(x_1,x_2+1))$$

or

$$((x_1,x_2),(x_1^2+x_1,x_2+x_1+1))$$

but there is no reduced triangular set equivalent to these families.

6 Triangular families from Gröbner bases.

In this section we give algorithms for computing triangular families from a Gröbner base. These algorithms do not depend explicitly on the ordering ; however, they are mainly designed for degree orderings for which the bases is more easily obtained ; they work also with lexicographical ordering, but, for them, we dispose of a structure theorem which permit us to provide a better algorithm (section 8).

Procedure 1 Triangular(G, K, m, n)

Input :

K *:* a field *;*

m *:* index of the first variable ($n = 1$ for the main call of Triangular) *;*

P_1 is a square–free ; resultant$(P_2, P_2') = X_1$, which is 0 if $X_1 = 0$ and 1 is $X_1 = 1$. Thus the equivalent reduced family is $((X_1,\ X_2),\ \ (X_1-1,\ X_2^2+1))$. There is no reduced triangular set equivalent to (10). Note also that the family obtained by D5 is the same as the family obtained by factorisation over the rationals, but not over the Gaussians.

Proposition 5 *i) If $(P_1, \ldots, P_n)$ is a triangular system such that $P_k = P_k'.P_k''$ modulo $(P_1, \ldots, P_{k-1})$ (this means factorisation, not derivatives), then $(P_1, \ldots, P_n)$ is equivalent to*

$$((P_1, \ldots, P_{k-1},\ P_k',\ P_{k+1}', \ldots, P_n'),\ (P_1, \ldots, P_{k-1},\ P_k'',\ P_{k+1}'', \ldots, P_n''))$$

where P_i' and P_i'' $(i > k)$ are obtained by reducing P_i by $P_1, \ldots, P_{k-1}$ and P_k' or P_k'' respectively.

ii) If $(P_1, \ldots, P_n)$ and $(Q_1, \ldots, Q_n)$ are two triangular sets such that $P_i = Q_i$ for $i < k$, that the degrees of P_i and Q_i are the same for $i > k$, and that GCD$(P_k, Q_k) = 1 \bmod (P_1, \ldots, P_{k-1})$, then $((P_1, \ldots, P_n), (Q_1, \ldots, Q_n))$ is equivalent to $(P_1, \ldots, P_{k-1},\ P_kQ_k,\ R_{k+1}, \ldots, R_n)$ where the R_i are computed by chinese remainder theorem.

Easy. Here are examples of application.

Part *i)* may applyed to each triangular sets appearing in (8), observing that complete factoriasation of $a^5 - 1$ is $(a-1)(a^4 + a^3 + a^2 + a + 1)$ and of $a^{10} + 123a^5 + 1$ is $(a^2 + 3a + 1)(a^5 - 3a^7 + 8a^6 - 21a^5 + 55a^4 - 21a^3 + 8a^2 - 3a + 1)$. Conversely, part *ii)* may be used for combining 1st and 4th or 5th and 6th triangular sets in (8) to obtain

$$\begin{array}{l}
(a^5 - 1,\ b^6 + 4ab^5 + 5a^2b^4 + 5a^3b^3 + 5a^4b^2 + 4a^5b + a^6,\\
\quad 5c + 8ab^5 + 30a^2b^4 + 30a^3b^3 + 25a^4b^2 + 30b + 22a,\\
\quad 5d - 2ab^5 - 10a^2b^4 - 15a^3b^3 - 10a^4b^2 - 10b - 8a,\\
\quad 5e - 6ab^5 - 20a^2b^4 - 15a^3b^3 - 15a^4b^2 - 15b - 9a)\\
(a^5 - 1,\ b - a,\ c - a,\ d^2 + 3ad + a^2,\ e + d + 3a)\\
(a^5 - 1,\ b - a,\ c^2 + 3ac + a^2,\ d + c + 3a,\ e - a)\\
(a^{10} + 123a^5 + 1,\ 55b^2 - 2a^6b - 233ab - 8a^7 - 987a^2,\\
\quad 55c + a^6 + 144a,\ 55d + a^6 + 144a,\ 55e + 55b - 2a^6 - 233a)
\end{array} \tag{11}$$

which is the value given by algorithm *D5lextriangular* described below.

We could also combine 5th and 6th triangular sets in (8) and the result with 4th set of (8). We get another equivalent family which consists of the is a product, and we have found factors without a factorisation algorithm. If we replace P_k by each of its factors (D and P_k/D), we get two triangular sets ; modulo the first one, $Q = 0$ is not invertible ; modulo the second one, the inverse of Q is RD^{-1}, which needs to inverse D modulo P_k/D.

Thus we may compute in A_k as it were a field, under the condition of eventually splitting it. This has been remarked by Dominique Duval and implemented by her and Claire Dicrescenzo in Reduce and Scratchpad II, under the name D5 (Della Dora et al. 1985 ; Dicrescenzo and Duval, 1985 and 1988 ; Duval, 1987). When solving a system of algebraic equations, we want to split it in triangular systems (proposition 2). This may be done by mean of factorisation in algebraic extensions (inefficient) or by mean of D5. Thus most of the algorithms which follow use D5 or factorisation in algebraic extensions, even if we try to avoid them when it is possible.

5 Combining and splitting triangular ideals.

Definition 2 — *Two sets of polynomials (and the ideals they generate) are* equivalent *if they have the same zeros. Two families of sets of polynomials are* equivalent *if the union of their common zeros are the same.*

Definition 3 — *A triangular set* $(P_1, \ldots, P_n)$ *of polynomials in* $K[X_1, \ldots, X_n]$ *is* reduced *if* P_k *is square–free modulo* $P_1, \ldots, P_{k-1}$ *for* $k = 1, \ldots, n$ *; this means that there exist polynomials* R *and* S *such that* $RP_k + SP'_k = 1$ *modulo* $P_1, \ldots, P_{k-1}$ *where* P'_k *is the derivative of* P_k *with repect to* X_k.

Proposition 4 — *Any triangular set of polynomials is equivalent with a family of reduced triangular sets, and such a family may be easily computed by D5 or by factorisation.*

A triangular set which generates a maximal ideal being reduced, this is a corollary of proposition 3. If $(P_1, \ldots, P_n)$ is the given system, the equivalent family may be obtained by replacing successively P_i by its irreducible factors for $k = 1, \ldots, n$. With D5, it suffices to compute, for $k = 1, \ldots, n$, the square–free decomposition of P_k modulo $P_1, \ldots, P_{k-1}$ and to replace P_k by the obtained square–free factors.

Note that computing the square–free decomposition of P_k may induce a splitting of P_i for $i < k$, as it is shown, by the following example :

$$(P_1 := X_1^2 - X_1; \;\; P_2 := X_2^2 + X_1) \tag{10}$$

It is clear from the preceeding proposition. •

Thus, from a triangular set of polynomials, finding the maximal ideals containing it, is as easy (or as difficult) as factorisation in algebraic extensions.

4 Computing modulo triangular ideals ; system D5.

It is clear that to compute numerically the common zeros of a triangular set is easy : this consists in solving monic univariate polynomials obtained successively by substituting the variables by the roots of the preceeding polynomials. Such a resolution is an example of computation with triangular sets of polynomials. On the other hand, field extensions which are not given by a primitive element (which is generally hard to compute) are defining by triangular sets (proposition 1), and computing in such field extensions means computing modulo triangular sets. These triangular sets are not the more general ones, but computing modulo a general triangular set is not very different from computing in algebraic extensions, as we shall see now.

Let $P_1, \ldots, P_n$ be a triangular set in $K[X_1, \ldots, X_n]$. Note that $P_1, \ldots, P_k$ is a triangular set in $K[X_1, \ldots, X_k]$ for $k = 1, \ldots, n$. Let $A_k := K[X_1, \ldots, X_k]/(P_1, \ldots, P_k)$. The polynomial P_{k+1} may be viewed as a polynomial in $A_k[X_{k+1}]$ and A_{k+1} is isomorphic whith $A_k[X_{k+1}]/P_{k+1}$. We have seen in proposition 3 that A_n is a field iff P_{k+1} irreducible as a polynomial in $A_k[X_{k+1}]$ for $k = 0, \ldots, n-1$.

Computing in A_k is very easy and is, more or less implemented in most computer algebra systems : the elements of A_k are represented as polynomials in $X_1, \ldots, X_n$ of degree in X_k less than the degree of P_k (in X_k), for $k = 1, \ldots, n$. The P_k being monic, dividing by them make no problem and the multiplication in A_n is a product of polynomials followed by divisions by $P_n,\ P_{n-1}, \ldots, P_1$.

Inversion in A_n (when it is a field) proceeds as follows : an element Q of A_n is a polynomial with X_k as main variable ; compute the extended GCD of Q and P_k in $A_{k-1}[X_k]$ (this need inversions in A_{k-1}) ; the result is

$$D = QR + P_k S$$

If $D = 1$, then R is the inverse of Q ; if $D = P_k$, then P_k divides Q and $Q = 0$ is not invertible in A_k ; there are no other possibilities if A_n is a field, because P_k is irreducible. If A_n is not a field, and $D \neq 1$, $D \neq P_k$, then P_k

Definition 1 — *The* main variable *of a polynomial is the greatest variable (for the above ordering) which appears in it. A set of n polynomials is* triangular *if the main variable of the i–th polynomial is X_i for $i = 1, \ldots, n$ and if this polynomial is monic as a polynomial in X_i. The* degree *of a triangular set is the product of the degrees (in their main variables) of its polynomials.*

The following three propositions are corollaries of results in Gianni et al. (1988) ; they may also easily deduced from most text books in commutative algebra. For the reader's convenience we give direct proofs, because we have not found any reference where they are explicit.

Proposition 1 — *If K is a field, any maximal ideal in $K[X_1, \ldots, X_n]$ has a triangular system of generators.*

Let I be a maximal ideal and A be the field $K[X_1, \ldots, X_n]/I$. Let us denotes by x_i the image of X_i in A and by $A_i = K(x_1, \ldots, x_i)$ the subfield of A generated by $(x_1, \ldots, x_i)$; thus $A_i = A_{i-1}(x_i)$ is a simple algebraic extension (A, beeing finitely generated as a ring, is an algebraic extension of K) ; let P_i be the minimal monic polynomial of x_i, with coefficients in A_{i-1} ; we have $A_i = A_{i-1}[X_i]/P_i$ and the elements of A_i are polynomials in X_i of degree less than the degree of P_i. Thus an easy recursion shows that $A = K[X_1, \ldots, X_n]/(P_1, \ldots, P_n)$ and that P_i is a polynomial in $X_1, \ldots, X_i$ which is monic in X_i. •

Proposition 2 — *Every system with a finite number of solutions (in an algebraic closure of K) is equivalent with the union of a finite number of triangular systems.*

Let I be the ideal generated by the polynomials of the system. The hypothesis implies that I is zero–dimensional and that its radical is an intersection of maximal ideals. Thus the zeros of the system are zeros of one of these maximal ideals, and the conclusion follows from proposition 1. •

Proposition 3 — *Let $(P_1, \ldots, P_n)$ be a triangular set of polynomials. Let Q_1 be an irreducible factor of P_1, A_1 be the field $K[X_1]/Q_1$; let Q_2 be an irreducible factor of P_2 in $A_1[X_2]$ and A_2 be the field $A_1[X_2]/Q_2$, and so on. The set $(Q_1, \ldots, Q_n)$ generates a maximal ideal containing $(P_1, \ldots, P_n)$ and the set of comnon zeros of the P_i is the union of the sets of common zeros of all $(Q_1, \ldots, Q_n)$.*

In triangular from, the solutions are

$$
\begin{aligned}
&(a^5-1,\ \ b^4+ab^3+a^2b^2+a^3b+a^4,\ \ c-a^4b^2,\ \ d-a^3b^3,\\
&\quad e+a^3b^3+a^4b^2+b+a)\\
&\hfill (20 \text{ solutions})\\
&(a^5-1,\ \ b-a,\ \ c-a,\ \ d^2+3ad+a^2,\ \ e+d+3a)\\
&\hfill (10 \text{ solutions})\\
&(a^5-1,\ \ b-a,\ \ c^2+3ac+a^2,\ \ d+c+3a,\ \ e-a)\\
&\hfill (10 \text{ solutions})\\
&(a^5-1,\ \ b^2+3ab+a^2,\ \ c+b+3a,\ \ d-a,\ \ e-a)\\
&\hfill (10 \text{ solutions})\\
&(a^{10}+123a^5+1,\ \ 55b+a^6+144a,\ \ 55c+a^6+144a,\\
&\quad 55d+a^6+144a,\ \ 55e-3a^6-377a)\\
&\hfill (10 \text{ solutions})\\
&(a^{10}+123a^5+1,\ \ 55b-3a^6-377a,\ \ 55c+a^6+144a,\\
&\quad 55d+a^6+144a,\ \ 55e+a^6+144a)\\
&\hfill (10 \text{ solutions})
\end{aligned}
\tag{8}
$$

These solutions are rather simple. However, the two last groups of 10 solutions seem to be more involved then the others. It is an artefact due to the choice of the ordering of the variables. In fact, the 5 groups of 10 solutions are the 5 circular permutations of solutions of the form

$$(a,\ a,\ a,\ ar,\ -a(r+3)) \quad \text{where } a^5=1 \text{ and } r^2+3r+1=0 \tag{9}$$

The simplification of solutions like the two last groups is a very difficult unsolved problem which hardly depends on the Galois structure of the ideal.

3 Triangular ideals ; theoretical results.

We have seen in the last section that simple and useful solutions are *triangular*. We will give a precise meaning to this. However, from now we only consider zero–dimensional systems, i.e. systems with only a finite number of solutions. For not zero–dimensional systems, such triangular systems may also be defined, and all solutions may be defined in term of some kind of triangular systems, but things are much more difficult (see Lazard, 1990).

Here we consider systems in n variables $X_1, \ldots, X_n$ and order them such that

$$X_1 < X_2 < \cdots < X_n.$$

plementation by M.C. Gontard) :

$$\begin{array}{l}
\text{If } a^2\,(b^2-a^2) \neq 0 \text{ then} \\
(d+c+b+a,\;\; (b^2-a^2)\,c+ab^2-a^3,\;\; -a^2b^4+(a^4-1)\,b^2-a^2) \\
\text{else if } -a^2b+a^3 \neq 0 \text{ then} \\
(d+c+b+a,\;\; (-ab^2+a^3)\,c-a^3b+1,\;\; b^2-a^2,\;\; -a^8+2a^4-1) \\
\text{else if } -a^2 \neq 0 \text{ then} \\
(d+c+b+a,\;\; -c^2-2ac-a^2,\;\; -a^2b+a^3,\;\; -a^8+a^4) \\
\text{else } 1
\end{array} \tag{4}$$

It is clear that solutions (2) and (4) are sufficently *triangular* and give the numeric solutions by successively giving values to the variables. But the wanted result is much more simpler : It is not difficult to show that the ideal generated by (1) is the intersection of the three ideals

$$\begin{array}{l}
(a+c,\;\; b+d,\;\; c^2d^2-1) \\
(a+b+2d,\;\; (b+d)^2,\;\; c-d,\;\; d^4-1) \\
(a-d,\;\; b+c+2d,\;\; (c+d)^2,\;\; d^4-1)
\end{array} \tag{5}$$

Even simpler, we may remark that the second polynomial of (2) is a square ; adding its square root $b+d$, we get as new Gröbner base

$$(a+c,\;\; b+d,\;\; c^2d^2-1) \tag{6}$$

which generates the radical of the ideal defined by (1).

The members of (5) and (6) are *triangular* ideals in the sense that the i–th variable may only appear in the first i polynomials. We will show that each zero–dimensional system (system with a finite number of solutions) may be solved as a finite union of such triangular ideals and we will provide efficient algorithms for finding such decompositions.

The preceeding system is not zero–dimensional. Thus, let us give another less easy example, the same as before with one variable more :

$$\begin{array}{c}
a+b+c+d+e \\
ab+bc+cd+de+ea \\
abc+bcd+cde+dea+eab \\
abcd+bcde+cdea+deab+eabc \\
abcde-1
\end{array} \tag{7}$$

This system has 70 solutions. The Gröbner base is not so easy to compute.

2 Some examples.

In this section we give some simple examples and show how the results of the classical algorithms are not satisfactorily. We first compare the brute result of a Gröbner base computation (for two different orderings) and of Wu Wen–Tsün (1987) algorithm with the simple result which would be the natural solution of a solver.

Consider the following set of polynomials (which becomes a system of equations by equating them the zero) ; this system and the subsequent one have been suggested by S. Arnborn (Davenport, 1987).

$$\begin{gathered} a+b+c+d \\ ab+bc+cd+da \\ abc+bcd+cda+dab \\ abcd-1 \end{gathered} \tag{1}$$

For the lexicographical ordering, its Gröbner basis is

$$\begin{aligned} & a+b+c+d \\ & b^2+2bd+d^2 \\ & bc-bd+c^2d^4+cd-2d^2 \\ & bd^4-b+d^5-d \\ & c^3d^2+c^2d^3-c-d \\ & c^2d^6-c^2d^2-d^4+1 \end{aligned} \tag{2}$$

For the degree ordering, the basis is

$$\begin{aligned} & a+b+c+d \\ & b^2+2bd+d^2 \\ & b^2c^2+c^2d-bd^2-d^3 \\ & bcd^2+c^2d^2-bd^3+cd^3-d^4-1 \\ & bd^4+d^5-b-d \\ & c^2d^4+bc-bd+cd-2d^2 \\ & c^3d^2+c^2d^3-c-d \end{aligned} \tag{3}$$

The brute result of Wu Wen–Tsün algorithm may be (Scratchpad imand what kind of solutions would be useful (section 2). Then, in section 3 we present our data structure for describing the solutions, *triangular sets of polynomials*. Computations modulo such set of polynomials is very close to computation in algebraic field extensions ; thus, in section 4, we present method D5, which is actually the best tool for such computations. In section 5, we show that the non unicity of the description of the solution by triangular sets of polynomials is not a drawback, because it is easy to pass from a description to another.

In sections 6 and 7 we show how the triangular sets may be obtained from a Gröbner basis with a good efficiency (nearly optimal) in the finite case; we start in section 6 from any Gröbner basis and in section 7 from a Gröbner base for a lexicographical ordering, with a better algorithm. The latter is intended to be used with the efficicent algorithm for changing the ordering described in Faugère et al (1989) giving a polynomial (in some natural meaning) algorithm for all the process of resolution if the set of solutions is finite even at infinity.

In section 8 algorithms are given for passing from solutions given as triangular sets to solutions in the special form used in Kobayashi et al. (1988) and in Gianni and Mora (1987) ; in this form, only the first polynomial in the triangular set (the one which is univariate) is non linear in its main variable. This form is more suitable for numeric resolution, but needs changes of variable. It is the only place in this paper where change of variable are needed.

The fact that our main algorithm is polynomial for a natural measure of complexity shows that the method of resolution is nearly optimal. However, in some cases we get a complicated solution where a simpler form exists, and it would be useful to get this simpler solution. This involves, at least, to have informations on the Galois structure of the set of solutions ; this is far outside of the goal of this paper. But the fact that we do not need any change of variable, has also the advantage that some informations on Galois structure may remain apparent on the result and are not necessarily hidden by the choice of generic coordinates.

Finally, it should also noticed that our problem is a special case of a much more difficult problem, the computation of a primary decomposition which is studied in Gianni et al. (1988).

some parameters which may be some of the variables. Unfortunately such an expression is generally impossible with rational functions. So we are led to another question : How to describe an infinite set of solutions?

We leave this last question to another paper (Lazard, 1990) and shall restrict our selves to the simpler and important case of a finite number of solutions. This number may be a big one and we are faced withe a new question which is not very different of the last one : How to describe a big number of solutions?

Most papers on algebraic systems are mainly concerned with algorithms for finding the solutions. Suprisingly, they do not focalise on the form of the provided solutions and on above questions, unless in most recent papers (Kobayashi et al., 1988 ; Gianni and Mora, 1987). These ask for giving solutions in a form which is a special case of what we call below a triangular set of polynomials. Unfortunately, for getting such a special form, linear change of variables are needed which do not preserve sparcity. The recent paper of Möller (1989) gives the solutions in the same form as in the present paper, but uses another method for computing them.

In this paper we show that "triangular sets of polynomials" is a good data structure for representing the solutions. We give different algorithms for transforming triangular sets and obtaining them from Gröbner bases. They work with the initial set of variables and thus preserve sparcity as far as it is possible.

The algorithm for getting triangular sets from a Gröbner base for lexicographical ordering is particulary simple. It seems (this has not yet be checked) that it follows from Langemyr (1990) that it is polynomial in the number of solutions.

Thus, the best method for solving zero–dimensional systems appears to be the following : compute the Gröbner base for the degree–reverse–lexicographical ordering by Buchberger algorithm ; deduce from it the Gröbner base for the pure lexicographical ordering by the algorithm of Faugère et al. (1989) ; then simplify the result as a family of triangular sets.

If input polynomials have degree d in n variables, the whole process is essentially polynomial in d^n (Lazard, 1983 ; Lakshman, 1990a and 1990b ; Lazard and Lakshman, 1990 ; Faugère et al., 1989 ; Langemyr, 1990). Thus, these algorithms are nearly optimal : the number of solutions may be d^n, by Bezout theorem.

Let us now describe the structure of the paper. We begin by some examples showing what kind of solutions provide some classical algorithms

Solving zero–dimensional algebraic systems*

D.Lazard†
Informatique, Université Paris VI
4, place Jussieu
F–75252 Paris Cedex 05

February 28, 1992

Abstract

It is shown that a good output for a solver of algebraic systems of dimension zero consists in a family of "triangular sets of polynomials" : Such an output is simple, readable and contains all information which may be wanted.

Different algorithm are described for handling triangular systems and getting them from Gröbner bases. These algorithms are practicable, and most of them are polynomial in the number of solutions.

1 Introduction.

In many computer algebra problems it is dificult to define what a good solution is. Algebraic systems are typical for such a situation : an algebraic system is a finite set of multivariate polynomials over some field K. Solving it means finding the common zeros of the input polynomials, in an algebraic closure of K.

What does it mean "finding"? If the solutions are finite in number, it means "giving the list of the solutions" and we are led to a new question : What is a solution?

When the set of solutions is not finite, another question appears : It is no more possible to list them and we have to describe them in some useful way. The most natural way is to express the variables as functions of

*Submited to J. Symbolic Computation

†CNRS, LITP and PRC Math. Info. ; e-mail : dl@frunip11.bitnet or lazard@litp.ibp.fr