ОБЪЕДИНЕННЫЙ
ИНСТИТУТ
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ

Дубна

E5-95-353

V.P.Gerdt, V.V.Kornyak

CONSTRUCTION OF FINITELY PRESENTED LIE ALGEBRAS AND SUPERALGEBRAS

Submitted to «Journal of Symbolic Computation»

1995

Гердт В.П., Корняк В.В. E5-95-353

Построение конечно-представленных алгебр и супералгебр Ли

Мы рассматриваем следующую задачу: по данной системе уравнений в форме полиномов Ли построить наиболее общую алгебру или супералгебру Ли, удовлетворяющую этой системе. Представление (супер)алгебр Ли конечным числом генераторов и определяющих соотношений является одной из наиболее общих математических и алгоритмических схем их анализа, имеющая важное значение для широкого класса проблем математической физики и комбинаторной алгебры. Например, для метода Уолквиста-Эстабрука в теории нелинейных интегрируемых уравнений в частных производных, для изучения конкретных (супер)алгебр Ли, возникающих в различных (супер)симметричных физических моделях, для исследования, в общем случае неразрешимой, проблемы тождества слов в некоммутативной и неассоциативной комбинаторной алгебре. Для решения рассматриваемой задачи необходим большой объем алгебраических вычислений, быстро возрастающий с ростом числа генераторов и соотношений. Поэтому, на практике, необходимо использование средств компьютерной алгебры. В этой работе мы описываем алгоритм и его реализацию на языке *C* для построения базиса и таблицы коммутаторов конечно-представленных (супер)алгебр Ли. Представлены некоторые компьютерные результаты, иллюстрирующие алгоритм и его реализацию.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Препринт Объединенного института ядерных исследований. Дубна, 1995

Gerdt V.P., Kornyak V.V. E5-95-353

Construction of Finitely Presented Lie Algebras and Superalgebras

We consider the following problem: what is the most general Lie algebra or superalgebra satisfying the given set of Lie polynomial equations? The presentation of Lie (super)algebras by a finite set of generators and defining relations is one of the most general mathematical and algorithmic schemes of their analysis. That problem is of great practical importance covering applications ranging from mathematical physics to combinatorial algebra. Among such applications there are: Wahlquist-Estabrook prolongation method in theory of integrable nonlinear partial differential equations; investigation of particular Lie (super)algebras arising in different (super)symmetric physical models; studying, generally algorithmically unsolvable, the word problem in non-commutative and non-associative combinatorial algebra. To solve this problem, one should perform a large volume of algebraic transformations which is sharply increased with growth of the number of generators and relations. By this reason, in practice, one needs to use a computer algebra tool. We describe here an algorithm and its implementation in *C* for constructing the basis of finitely presented Lie (super)algebra and its commutator table. Some computer results illustrating our algorithm and its actual implementation are presented.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Preprint of the Joint Institute for Nuclear Research. Dubna, 1995

1 Introduction

A *Lie algebra* L is an algebra over the commutative ring K with unit. Non-commutative and non-associative multiplication in Lie algebra is called *Lie product* and denoted usually by commutator $[\ ,\ ]$. The Lie product satisfies the following axioms for any $u, v, w \in L$

$$[u,v] = -[v,u], \qquad skew-symmetry, \tag{1}$$
$$[u,[v,w]] + [v,[w,u]] + [w,[u,v]] = 0, \qquad Jacobi\ identity. \tag{2}$$

A *Lie superalgebra* is $\mathbf{Z}_2$-graded algebra $L = L_{\bar{0}} \oplus L_{\bar{1}}$ with product $[\ ,\ ]$, i.e. if $u \in L_\alpha$, $v \in L_\beta$, $\alpha, \beta \in \mathbf{Z}_2 = \{\bar{0}, \bar{1}\}$, then $[u,v] \in L_{\alpha+\beta}$. The elements of $L_{\bar{0}}$ and $L_{\bar{1}}$ are called *even* and *odd*, respectively. The Lie product satisfies now the modified axioms

$$[u,v] = -(-1)^{\alpha\beta}[v,u], \tag{3}$$
$$[u,[v,w]] = [[u,v],w] + (-1)^{\alpha\beta}[v,[u,w]], \tag{4}$$
$$u \in L_\alpha, \ v \in L_\beta.$$

These definitions can be generalized in the following way (Bahturin, Mikhalev, Petrogradsky and Zaicev, 1992). Let G be an abelian additive group grading certain algebra $L = \oplus_{g \in G} L_g$. Let ε be a *bilinear alternating form*

$$\varepsilon : G \times G \to K^*$$

satisfying the following properties

$$\varepsilon(\alpha+\beta,\gamma) = \varepsilon(\alpha,\gamma)\varepsilon(\beta,\gamma), \ \ \varepsilon(\alpha,\beta+\gamma) = \varepsilon(\alpha,\beta)\varepsilon(\alpha,\gamma), \ \ \varepsilon(\alpha,\beta) = \varepsilon(\beta,\alpha)^{-1},$$

where $\alpha, \beta, \gamma \in G$; K^* is the multiplicative group of invertible elements in K. If we replace $(-1)^{\alpha\beta}$ by $\varepsilon(\alpha,\beta)$ in (1.3-1.4), we obtain the definition of (ε-)*colour Lie superalgebra* combining generally more than two (even and odd) features in the same structure. If $G = \{0\}$, then L is an ordinary Lie algebra. The case of ordinary Lie superalgebra corresponds to $G = \mathbf{Z}_2$, $\varepsilon(\bar{0},\bar{0}) = \varepsilon(\bar{0},\bar{1}) = \varepsilon(\bar{1},\bar{0}) = 1$, $\varepsilon(\bar{1},\bar{1}) = -1$.

Note that if we consider an ordinary or colour Lie superalgebra over a field of characteristic 2 or 3, we have to add some extra axioms. Characteristic 2 requires also the existence of certain *quadratic operator* (see Ufnarovsky, 1990; Bahturin, Mikhalev, Petrogradsky and Zaicev, 1992).

Finitely presented algebra is determined by a finite number of some its elements called generators subject a finite number of relations having a form of polynomials in the algebra. Any finite-dimensional algebra is, obviously, finitely presented one. Nevertheless, the concept of a finite presentation covers also a wide classes of infinite-dimensional algebras. Some of these algebras have a natural constructive definition in terms of a finite number of generators and relations.

Some examples of infinite-dimensional finitely presented Lie (super)algebras:

1. Kac and Kac-Moody (super)algebras (Kac, 1990) with their generalization known as Borcherds algebras (Gebert, 1994).

2. Lie (super)algebras of the string theories: Virasoro, Neveu-Schwarz and Ramond algebras (Leites, 1984).

3. Any simple finite-dimensional Lie algebra can be generated by two elements only with the number and structure of relations independent of the rank of the algebra. This allows to define such objects as Lie algebras of matrices of a complex size $\mathrm{sl}(\lambda)$, $\mathrm{o}(\lambda)$ and $\mathrm{sp}(\lambda)$, where λ may be any complex number or even ∞ (Grozman and Leites, 1995a). In a similar way, one can define some Lie superalgebras of supermatrices of a complex size (Grozman and Leites, 1995b).

Below we describe an algorithm and its C implementation for determining the explicit structure of finitely presented Lie (super)algebra from the defining relations, i.e., for constructing its basis and commutator table. In fact, our algorithm produces the Gröbner basis (Ufnarovsky, 1990) for non-commutative and non-associative case of Lie (super)algebras. The algorithm and its actual implementation is illustrated by rather simple example arising in investigation of some supersymmetric model equation of mathematical physics. We also present the table containing computational statistics for the standard relations of all simple Lie algebras up to rank 10.

2 Algorithm

Let us explain, first of all, some terms used in the text.

The set $X = \{x_1, x_2, \ldots, x_k\}$ of *generators* is a set of Lie (super)algebra elements from which any other element may be constructed by Lie product, addition and multiplication by elements in K (scalars).

A *basis* $B(X)$ of Lie (super)algebra is a minimal set of elements such that any other element is their linear combination.

A *Lie monomial* ("word") $m(X)$ is any element of L constructed from the generators x_i by Lie products. A Lie polynomial $P(X, C)$ is a sum of Lie monomials multiplied by scalar coefficients $C = \{c_1, \ldots, c_l\}$, $c_i \in K$.

The set of *defining relations* R is the set of Lie polynomial equalities of the form $P(X, C) = 0$.

Lie (super)algebra L is called *finitely presented* one if the both sets X and R are finite.

The finitely presented Lie (super)algebra L_F without defining relations, i. e., with the empty set R, is called *free Lie (super)algebra*.

Any finitely presented Lie (super)algebra can be considered as the quotient algebra of L_F by the two-sided ideal generated by relations R. Thus, it makes sense to deal with only those Lie monomials which constitute a basis of the free Lie (super)algebra,

i.e., a set of Lie monomials which are not expressible in terms of others by means of (1.1-1.4).

It is known that a suitable basis of free Lie (super)algebra can be formed by *regular* (ordinary Lie algebra), *s-regular* (Lie superalgebra in characteristic 0) and *ps-regular* (Lie superalgebra in characteristic p) Lie monomials (Bahturin, Mikhalev, Petrogradsky and Zaicev, 1992).

Monomials are regular if they are either generators or commutators of the form $[u, v]$ or $[w, [u, v]]$, where u, v, w are regular and $u < v$ and $w \geq u$ with respect to some linear ordering of Lie monomials. Depending on the ordering chosen, one obtains a particular basis for a free Lie algebra. Among the whole variety of bases, the most often used ones were introduced by Hall and Shirshov (Ufnarovsky, 1990). Without getting into details, we remark only, Shirshov and Hall orderings are analogous, in some sense, to the pure lexicographical and graded lexicographical orderings for associative words. In the present algorithm we use Hall ordering because it is compatible with the natural grading generated by Lie product. The use of Shirshov ordering may give, as we hope, an additional information about the structure of Lie algebra but, as well as in the associative case, decreases the efficiency because a Lie monomial may contain a greater, w.r.t. the ordering, submonomial that complicates the structure of data and algorithm. So we put its consideration off for the future.

To get a full set of s-regular monomials, we have to add only Lie squares of odd regular monomials.

The set of ps-regular monomials contain also pth associative (in the sense of universal enveloping algebra) powers of s-regular monomials. In this work we consider the case $p = 0$ only and we shall use below the term "regular" assuming any of the above prefixes.

In general terms to reduce a given set of Lie polynomials to Gröbner basis, we should compute all possible consequences of these polynomials and remove all dependencies among them. The problem is to do that in the most efficient way. There were elaborated a number of optimizing criteria to avoid unnecessary reductions in computation of associative (Ufnarovsky, 1990) and commutative (Buchberger, 1985; Becker, Weispfenning and Kredel, 1993) Gröbner bases. Unfortunately, similar criteria have not been found yet for the non-associative case. Nevertheless, we use some simple methods to decrease the volume of computation. Most important of them are the following:

1. *It is sufficient to multiply the relations by generators only* to obtain all the consequences. It is clear from Jacobi identity $[[u, v], r] = \pm[u, [v, r]] + \pm[v, [u, r]]$ that multiplication of relation r by commutator of Lie monomials u, v is equivalent to the linear combination of the subsequent Lie products with the components of a commutator. Using this formula recursively, we come to generators.

2. *There is no need to multiply the relation by the generator which form a regular monomial with the leading monomial* of relation since all such consequences are automatically reduced to zero. Let relation have the form $u + a = 0$ with the leading monomial u and a contains other terms. Multiplying the relation by generator x, we

obtain $[x,u]+[x,a]=0$. If $[x,u]$ is a regular monomial, we must replace u by $-a$ in it that leads to $-[x,a]+[x,a]\equiv 0$.

3. *All computations*, starting with processing the input relations, *are executed modulo identities* (1.1-1.4) *and the relations have been treated to the moment.* This allows to minimize resimplification of the calculated structures and to keep the system of Lie monomials and relations as compact as possible all the time.

The algorithm has the following input and output structure:

Input: The set of generators $X=\{x_1,x_2,\ldots\}$ with prescribed parities $\alpha_i\in\mathbf{Z}_2$ and positive integer weights w_i ($=1$ by default);
the set of scalar parameters $P=\{p_1,p_2,\ldots\}$ if they present in the relations;
the set of defining relations $R=\{r_1,r_2,\ldots\}$, where r_i are Lie polynomials with coefficients from the commutative ring $\mathbf{Z}[p_1,p_2,\ldots]$ of scalar polynomials;
the limiting number for generated relations because it is necessary to stop computation in the case of infinite Gröbner basis.

Output: The reduced set of relations (Gröbner basis) $\tilde{R}=\{\tilde{r}_1,\tilde{r}_2,\ldots\}$;
the list of basis elements $E=\{e_1,e_2,\ldots\}$;
the commutator table $[e_i,e_j]=c_{ij}^k e_k$, where c_{ij}^k are structure constants;
the table of expressions containing p_i and considered as non-zeros during computation. Particular values of p_i may cause the branching of computation and, possibly, of the resulting algebra structure;
dimensions of homogeneous components in obtained Lie (super)algebra.

There are three principal steps in the algorithm:

1. *Reduction of the initial set R to the equivalent Gröbner basis $\tilde{R}=\{\tilde{r}_1,\tilde{r}_2,\ldots\}$.* This step executes the subsequent multiplying of relations by generators adding non-zero results to the set of relations and substituting these new relations into the other ones. The process terminates if either all newly arising relations are reduced to zero or the number of relations goes up to the limit fixed at input. In the first case Gröbner basis consists of a finite number of relations [1]. The second case means that either algebra is infinite-dimensional or the input limiting number of the relations is too small.

2. *Construction of the Lie (super)algebra basis.* Some basis elements obtained at Step 1 as Lie (sub)monomials of $\tilde{r}_i$, but the basis must be completed by the regular commutators of already existing basis elements in the infinite-dimensional case and by the Lie squares of the odd elements in the case of superalgebra.

3. *Construction of the commutator table.* Here the commutators of the basis elements obtained at Step 2 are computed by the direct commutating with the further reduction of the resulting expression modulo the relations $\tilde{R}$.

[1] It does not mean, however, that algebra necessarily be finite-dimensional.

3 Implementation and Sample Session

The algorithm has been implemented in *C* language. The source code has the total length about 7500 lines and contains about 150 *C* functions realizing: top level algorithms, Lie (super)algebra operations, manipulation with scalar polynomials, multiprecision integer arithmetic, substitutions, list processing, input and output etc.

The following session file has been produced on a 25 Mhz MS-DOS based AT/386 computer. We use here 32bit *GCC* compiler and *GO32* DOS extender, though for considered small example the 16bit *Borland C++ 3.1* environment is quite sufficient (and takes twice smaller space for the calculated structures). The illustrative example given below yields relatively compact output and has been studied in (Roelofs, 1993). This example arises in investigation of supersymmetries of $N=1$ superization of KdV equation (Manin and Radul, 1985). The relations contain two even generators x_1 and x_2 and odd generator y (prefixed by the sign "$-$" at the input description). We deform the original system by two parameters a and b to get a parametric ring and, thus, to illustrate the classification problem.

Note that the program asks for the output form of Lie monomials. In this example we choose the standard one. Otherwise, one can choose the *right-normed* arrangement for non-associative monomials. It means, for instance, that Lie monomials $[x,[x,[y,x]]$ and $[[x,y],[x,[x,y]]]$ are presented in the output as x^2yx and $(xy)x^2y$, respectively. Such notations are more compact and expressive especially for high degree Lie monomials and widely used by algebraists.

Input data can be entered from either a keyboard or a separate file.

```
Enter name of existing or new input file -> skdvab.in
Input data:
Generators: x_2 -y x_1;
Parameters: a b;
Relations:
[[[y,x_1],x_1],x_1];
[y,x_2];
[y,[[[y,x_1],x_1],y]];
[y,[[y,x_1],[y,x_1]]] - a [[y,[[y,x_1],y]],x_1];
[x_1,x_2] - [[y,[[y,[[y,x_1],y]],y]],y];
[x_1,[[y,x_1],y]] + [[y,x_1],[y,x_1]] + [[[y,x_1],x_1],y];
[x_1,[[y,x_1],[y,x_1]]] + b [x_1,[[[y,x_1],x_1],y]];
[x_1,[[y,[[y,x_1],y]],y]] - 3 [[y,x_1],y] - [[[y,[[y,x_1],y]],x_1],y];

Right-normed output for Lie monomials? (y/n) -> n
Standard grading assumes unit weight for every generator.
Do you want to use a different grading? (y/n) -> n
Enter limiting number for relations -> 20


Initial relations:

(1)  [x ,y] = 0
       2
```

(2) $[x_1,[x_1,[y,x_1]]] = 0$

(3) $[x_1,[y,[y,[y,x_1]]]] = 0$

(4) $[[y,x_1],[y,[y,x_1]]] = 0$

(5) $(2\,b - 2)\ [[y,x_1],[x_1,[y,x_1]]] + b\ [x_1,[x_1,[y,[y,x_1]]]] = 0$

(6) $[x_1,[y,[y,[y,[y,x_1]]]]] - 3\ [y,[y,x_1]] = 0$

(7) $[y,[y,[y,[y,[y,[y,x_1]]]]]] + [x_2,x_1] = 0$

Non-zero parametric coefficients:

(1) $a - 2$

(2) $b - 1$

(3) $b^2 + b - 2$

Reduced relations:

(1) $[x_2,y] = 0$

(2) $[x_2,x_1] = 0$

(3) $[y,[y,x_1]] = 0$

(4) $[x_1,[x_1,[y,x_1]]] = 0$

(5) $[[y,x_1],[x_1,[y,x_1]]] = 0$

(6) $[[x_1,[y,x_1]],[x_1,[y,x_1]]] = 0$

Basis elements:

(1) $E_1 = x_2$

(2) $O_2 = y$

(3) $E_3 = x_1$

(4) $E_4 = [y,y]$

(5) $O_5 = [y,x_1]$

(6) $O_6 = [x_1,[y,x_1]]$

(7) $E_7 = [[y,x_1],[y,x_1]]$

Non-zero commutators of basis elements:

(1) $[O_2,O_2] = E_4$

(2) $[O_2,E_3] = O_5$

(3) $[E_3,O_5] = O_6$

(4) $[O_5,O_5] = E_7$

(5) $[O_2,O_6] = E_7$

Dimensions of homogeneous components:

$\dim G_1 = 3$

$\dim G_2 = 2$

$\dim G_3 = 1$

```
dim G    = 1
     4


Time: 0.05 sec
Number of relations:    15  Relation space:         120 bytes
Number of ordinals:     40  Ordinal space:          480 bytes
Number of nodes:        50  Node space:             600 bytes
Total space: 1200 bytes
```

Here E_i and O_i are even and odd basis elements, respectively. In the case of infinite-dimensional algebra the program prints out only those commutators which can be expressed in terms of the basis elements have been computed.

In the above example the chosen ordering among generators $x_2 < y < x_1$ provides the minimal number of the reduced relations in the output. As well as for the commutative Gröbner basis method the final structure of the reduced relations and even their number essentially depends on the ordering chosen.

It can be easily seen that for the generic values of parameters a and b we have seven-dimensional nilpotent Lie superalgebra. The branching of the algebra structure is possible at the values of parameters $a = 2$, $b = 1$ and $b = -2$. The computations with these particular values show that the choice $b = 1$ or $b = -2$ leads to the same algebra structure, whereas at $a = 2$ the algebra becomes to be infinite-dimensional one. In (Roelofs, 1993) this algebra at $a = 2$ and $b = 1$ was identified with the product of the seven-dimensional nilpotent algebra and the positive subalgebra of twisted Kac-Moody superalgebra $C^{(2)}(2)$.

4 Standard Relations for Simple Lie Algebras

In Table 1 we present the results of applying the program to the standard relations for all simple Lie algebras up to rank 10. The timings are presented for the above mentioned personal computer.

Any (semi)simple complex Lie algebra L possesses *Gauss decomposition* $L = E \oplus H \oplus F$, where H is commutative *Cartan subalgebra*, E and F are *positive* and *negative* nilpotent subalgebras. This decomposition is compatible with the following relations containing *Cartan elements* h_i and *Chevalley generators* e_i, f_i corresponding to positive and negative simple roots of the algebra (Jacobson, 1962):

$$[h_i, h_j] = 0, \tag{5}$$
$$[e_i, f_j] = \delta_{ij} h_j, \tag{6}$$
$$[h_i, e_j] = a_{ji} e_j, \tag{7}$$
$$[h_i, f_j] = -a_{ji} f_j, \tag{8}$$
$$(ad\, e_i)^{1-a_{ji}} e_j = 0, \tag{9}$$
$$(ad\, f_i)^{1-a_{ji}} f_j = 0, \tag{10}$$

where a_{ij} is *Cartan matrix*, $i, j = 1, \ldots, r = rank\ L$. Note that for Kac-Moody algebras just the same relations hold with slightly more general Cartan matrix.

Relations (4.5-4.6) are called *Serre relations*. One can see that these relations include only Chevalley generators corresponding to positive and negative subalgebras E and F carrying the principal part of information about algebra. Serre relations, being taken separately, define subalgebras E and F.

One can see that the calculation of exceptional algebra E_8 is the most difficult task among those included in Table 1. The number of initial relations here is 290. The program generates the Gröbner basis which contains 23074 relations involving Lie monomials up to degree 58 while Lie algebra basis elements go up to 29th degree. The task requires 15 min 36 sec of computing time and 815516 bytes of memory. Note that a separate processing of Serre relations for this algebra takes 1 min 13 sec and 186096 bytes.

Content of columns in Table 1 is as follows:
- *Dim* is dimension of the algebra,
- N_{in} is a number of the input relations,
- N_{GB} is a number of the relations in Gröbner basis,
- N_{comm} is a number of the non-zero commutators,
- D_{GB} is a maximum degree of Lie monomial in Gröbner basis,
- Space is maximum memory occupied by computed structures,
- Time is the running time excepting input-output operations.

5 Conclusion

Unlike commutative algebra, where such an universal tool for analysis of polynomial ideals as Buchberger's algorithm for computing of the Gröbner basis has been developed (Buchberger, 1985; Becker, Weispfenning and Kredel, 1993), its generalizations to non-commutative (Mora, 1988; Kandri-Rody and Weispfenning, 1990; Ufnarovsky, 1990) and especially to non-associative algebras are still far from being practically useful. Moreover, because of very serious mathematical and algorithmic problems are still to be solved, there are only a few packages implementing the non-commutative Gröbner basis technique, and no one of them so far is able to deal with non-associative algebras.

It justifies the practical use of other algorithmic methods. Among them there is one based on the straightforward verification of Jacobi identities (Gragert, 1989; Roelofs, 1991). Its implementation in the form of the *Reduce* package was successfully applied to a number of problems in mathematical physics.

Our algorithm reveals some common features with the involutive techniques in commutative algebra (Gerdt and Blinkov, 1995), namely, combining prolongations by the generators with subsequent reductions. The involutive approach can be considered as another efficient algorithmic method to the Gröbner basis construction, different from Buchberger's algorithm. By this analogy we hope that the further analysis of our

Table 1.

Algebra	Dim	N_{in}	N_{GB}	N_{comm}	D_{GB}	Space, *bytes*	Time, *seconds*
A_2	8	17	24	21	4	1188	< 1
A_3	15	40	84	60	6	3612	< 1
A_4	24	72	218	126	8	8716	1
A_5	35	113	473	225	10	18088	2
A_6	48	163	908	363	12	33700	6
A_7	63	222	1594	546	14	57908	13
A_8	80	290	2614	780	16	93452	27
A_9	99	367	4063	1071	18	143456	51
A_{10}	120	453	6048	1425	20	211428	89
B_2	10	17	35	28	6	1672	< 1
B_3	21	40	149	106	10	6160	1
B_4	36	72	441	263	14	17148	3
B_5	55	113	1047	522	18	39180	9
B_6	78	163	2153	906	22	78544	25
B_7	105	222	3981	1441	26	142620	59
B_8	136	290	6792	2150	30	240024	124
B_9	171	367	10904	3057	34	381256	241
B_{10}	210	453	16683	4185	38	578364	444
C_3	21	40	138	106	10	5772	1
C_4	36	72	411	263	14	16032	3
C_5	55	113	968	522	18	36304	9
C_6	78	163	2007	906	22	73320	25
C_7	105	222	3756	1441	26	134652	62
C_8	136	290	6439	2150	30	227672	130
C_9	171	367	10398	3057	34	363720	248
C_{10}	210	453	15999	4185	38	554832	451
D_4	28	72	283	179	10	11336	1
D_5	45	113	726	389	14	27768	5
D_6	66	163	1573	713	18	58292	15
D_7	91	222	3034	1174	22	109964	39
D_8	120	290	5355	1798	26	190932	87
D_9	153	367	8817	2608	30	310452	174
D_{10}	190	453	13762	3628	34	479792	328
G_2	14	17	73	56	10	3200	< 1
F_4	52	72	858	544	22	32832	10
E_6	78	163	2186	1003	22	80740	27
E_7	133	222	6389	2527	34	230140	132
E_8	248	290	23074	7710	58	815516	936

method could give a new insight to generalization of the Gröbner basis approach to Lie (super)algebras.

The program can be easily modified for working over finite fields and generalized to handle colour Lie superalgebras with finite grading groups.

6 Acknowledgments

We are grateful to J. Backelin, P. Gragert, D. Leites, V.Robuk, M.Roelofs, and especially to V. Ufnarovsky for fruitful discussions. This work was supported in part by the INTAS project No. 93-0030.

References

[1] Bahturin, Yu.A., Mikhalev, A.A., Petrogradsky, V.M., Zaicev, M.V. (1992). *Infinite dimensional Lie superalgebras.* Walter de Gruyter. Berlin- New York.

[2] Becker, T., Weispfenning, V., Kredel, H. (1993). *Gröbner Bases. A Computational Approach to Commutative Algebra.* Graduate Texts in Mathematics **141**, Springer-Verlag, New York.

[3] Buchberger, B. (1985). Gröbner bases: an algorithmic method in polynomial ideal theory, in *Recent Trends in Multidimensional System Theory*, ed. N.K.Bose, (D.Reidel), pp.184-232.

[4] Gebert, R. W. (1994). *Beyond Affine Kac-Moody Algebras in String Theory,* DESY 94-209, Hamburg.

[5] Gerdt, V.P., Blinkov, Yu.A. (1995). Involutive Polynomial Bases, Publication IT-95-271, LIFL USTL, Lille, Submitted in J. Symb. Comp.

[6] Gragert, P.K.H. (1989). Lie Algebra Computations. *Acta Applicandae Mathematicae* **16**, 231-242.

[7] Grozman, P., Leites, D. (1995a). Defining Relations Associated with the Principal sl(2)-subalgebras, to appear.

[8] Grozman, P., Leites, D. (1995b). Lie Superalgebras of Supermatrices of Complex Size, to appear.

[9] Jacobson, N. (1962). *Lie Algebras,* Interscience Publishers, New York-London.

[10] Kac, V.G. (1990). *Infinite dimensional Lie algebras.* Cambridge University Press, Cambridge, third edition.

[11] Kandri-Rody, A., Weispfenning, V. (1990). Non-commutative Gröbner bases in Algebras of Solvable Type. *J. Symb. Comp.* **9**, 1-26.

[12] Leites, D. (1984). *Lie superalgebras.* VINITI. Itogi Nauki i Tekhniki. Modern Problems in Mathematics. Recent Progress, **25**, Moscow, pp.3-50 (in Russian).

[13] Manin, Yu.I., Radul, A.O. (1985). A supersymmetric extension of the Kadomtsev-Petviashvili hierarchy, *Commun. Math. Phys.* **98**, 65-77.

[14] Mora, T. (1988). Seven Variations on Standard Bases. Preprint No.45, Dip. di Mathematica, Univ. di Genova.

[15] Roelofs, G.H.M. (1991). *The LIESUPER Package for REDUCE,* Memorandum 943, Univ. of Twente, Netherlands.

[16] Roelofs, G.H.M. (1993). *Prolongation structures of supersymmetric systems.* Ph.D. Thesis, Univ. of Twente, Enschede, Netherlands.

[17] Ufnarovsky, V.A. (1990). *Combinatorial and asymptotic methods in algebra.* VINITI. Itogi Nauki i Tekhniki. Modern Problems in Mathematics. Fundamental Branches, **57**, Moscow, pp.5-177 (in Russian).

Received by Publishing Department
on August 3, 1995.