

Isomorphism testing for Lie algebras over finite fields and simple Lie algebras over the field with two elements

Bettina Eick

Abstract

We describe an algorithm for computing automorphism groups and testing isomorphisms of finite dimensional Lie algebras over finite fields. The algorithm is particularly effective for simple or almost simple Lie algebras. We show how it can be used in a computer search for new small-dimensional simple Lie algebras over the field with two elements.

1 Introduction

Algorithms for computing automorphism groups and for testing isomorphisms of Lie algebras over finite fields are available for certain classes of Lie algebras so far: Schneider [9] describes a method for nilpotent Lie algebras and in [3] a method for solvable Lie algebras is introduced. Both are available as part of the computer algebra system GAP [12] and have proved to be effective. For example, the method by Schneider [9] has been used to classify the nilpotent Lie algebras of dimension 9 over the field with two elements \mathbb{F}_2 .

The first aim of this paper is to describe an algorithm to test isomorphism and to compute automorphism groups for arbitrary finite dimensional Lie algebras over finite fields. This new method determines and uses 'nice' presentations for the considered Lie algebras. We describe two variations of our algorithm: a deterministic and a Monte-Carlo version. The Monte-Carlo version may fail to detect an isomorphism (with a small probability), but it is significantly more effective than the deterministic version. A report on its implementation in the computer algebra system GAP [12] is included below.

The classification of simple Lie algebras in positive characteristic has recently been completed for all algebraically closed fields of characteristic at least 5, see the book by Strade [10] for details. The classifications in characteristic 2 and 3 are still wide open. Vaughan-Lee [13] used computational methods to determine up to isomorphism all simple Lie algebras of dimension at most 9 over \mathbb{F}_2 . His approach yields a complete classification, but it seems unpractical for larger dimensions.

The second aim here is to show how the new isomorphism testing method can be used in a computer-based search for new simple Lie algebras over \mathbb{F}_2 . Our approach uses a randomized search for simple Lie algebras. It does not yield a complete classification, but found some new simple Lie algebras in the dimension 15 and 16. We include a report on this computation below.

As a result of our computation, we obtain the following table of the currently known simple Lie algebras of dimension at most 20 over \mathbb{F}_2 . The table exhibits for every listed Lie algebra its names (see below for details) as well as the dimension of its derivation Lie algebra (*der*), the order of its automorphism group (*aut*) and shows whether it is restricted (*res*). Lie algebras in different rows of this table are non-isomorphic.

dim	nr	names	der	aut	res
3	1	$W(2)$	5	6	-
6	1	$W(2) \otimes \mathbb{F}_4$	10	120	-
7	1	$W(3)$	10	16	-
7	2	$V_7, P(1, 2)$	10	4	-
8	1	$A_2, W(1, 1), Q(1, 1, 1)$	8	336	+
8	2	V_8	8	432	+
9	1	$W(2) \otimes \mathbb{F}_8, V_9$	15	1512	-
10	1	$Kap_3(5)$	14	720	-
12	1	$W(2) \otimes \mathbb{F}_{16}$	20	16320	-
14	1	$W(3) \otimes \mathbb{F}_4$	20	1536	-
14	2	$V_7 \otimes \mathbb{F}_4$	20	96	-
14	3	$S(2, 2)$	20	1536	-
14	4	$P(1, 1, 1, 1), Kap_1(4)$	20	1152	-
14	5	$A_3, B_3, C_3, G_2, S(1, 1, 1), H(1, 1, 1, 1)$	21	1451520	+
14	6	$Bro_2(1, 1)$	20	10752	-
15	1	$W(2) \otimes \mathbb{F}_{32}$	25	163680	-
15	2	$W(4)$	19	2048	-
15	3	$Kap_3(6), Kap_2(4),$	20	23040	-
15	4	$P(2, 1, 1)$	19	64	-
15	5	$P(3, 1)$	19	512	-
15	6	$P(2, 2)$	19	256	-
15	7	New	19	32	-
15	8	New	19	192	-
16	1	$W(1, 1) \otimes \mathbb{F}_4, V_8 \otimes \mathbb{F}_4$	16	241920	+
16	2	$W(2, 1), Q(2, 1, 1)$	17	2048	-
16	3	New	17	1536	-
16	4	New	17	384	-
16	5	New	17	512	-
16	6	New	17	360	-
18	1	$W(2) \otimes \mathbb{F}_{64}$	30	1572480	-
20	1	$Kap_3(5) \otimes \mathbb{F}_4$	28	1958400	-

The names for these Lie algebras arise as follows.

- $L \otimes \mathbb{F}_m$ is the tensor of the Lie algebra L with the finite field \mathbb{F}_m with m elements; see Section 7.
- A, B, C, D, E, F, G describe the simple constituents of the classical Lie algebras as determined by Hiss [4] and Hogeweij [5].
- W, S, H, K describe the simple constituents of the Lie algebras of Cartan type; these have been determined computationally from the Lie algebras of Cartan type.
- P describes the Hamiltonian-type Lie algebras as determined by Lin [8];
- Q describes the Contact-type Lie algebras as described by Zhang and Lin [14];
- Kap_i describes the i th series of Lie algebras determined by Kaplansky [7] for $1 \leq i \leq 4$.
- Bor_i describes the i th series of Lie algebras determined by Brown [1] for $1 \leq i \leq 3$.
- V_7, V_8 and V_9 are simple Lie algebras determined by Vaughan-Lee [13].

2 Preliminaries

Many algorithms for Lie algebras are described in the book by de Graaf [2]. The book focusses particularly on Lie algebras in characteristic 0. The characteristic $p > 0$ case, in particular the case of finite fields, is less well developed in the book and in implementations as well.

The aim of this section is to describe some basic algorithms for Lie algebras over a finite field \mathbb{F} . We assume throughout that all arising Lie algebras are given by structure constants tables; that is, for a Lie algebra L we are given a basis b_1, \dots, b_n and the elements $\alpha_{i,j,k} \in \mathbb{F}$ with

$$b_i b_j = \sum_{k=1}^n \alpha_{i,j,k} b_k.$$

In this setting it is straightforward to compute a basis for a subalgebra of L given by generators, to check whether a subalgebra is an ideal and to compute structure constants tables for sub- and quotient algebras. Further, we can readily determine its derivation Lie algebra $Der(L)$ as a subalgebra of the matrix algebra $M_{n \times n}(\mathbb{F})$, see the book by de Graaf [2] for details.

2.1 Ideals and constituents of Lie algebras

There exists a highly effective algorithm to check whether a given Lie algebra L over a finite field \mathbb{F} is simple or, if not, then to compute the simple constituents of L . For this purpose let m_j denote the right adjoint of the basis element b_j for $1 \leq j \leq n$. Then m_j is a $n \times n$ -matrix over \mathbb{F} and it can be read off from the structure constants table of L . Let M be the associative algebra generated by the matrices m_1, \dots, m_n . Then M acts by multiplication from the right on $V = \mathbb{F}^n$.

1 Lemma: *The ideals of L correspond one-to-one to the M -submodules of V .*

Proof: By construction, the M -submodules of V correspond one-to-one to the right ideals of L . As L is a Lie-algebra, right ideals are two-sided ideals and thus the result follows. •

Using the so-called ‘MeatAxe’, see [6] for background, we can readily check whether the associative algebra M acts irreducibly (and thus L is simple) or determine a non-trivial M -submodule of V (and thus a non-trivial ideal I of L). In the later case we can then apply the method recursively to L/I and I and thus finally obtain a series

$$L = L_1 > L_2 > \dots > L_r > L_{r+1} = \{0\},$$

with $L_{i+1} \trianglelefteq L_i$ and L_i/L_{i+1} simple. This allows to read off the simple constituents of L . We note that the Meataxe can also effectively compute the minimal and maximal submodules and the radical and socle series of a module. Hence the corresponding ideals of L can also be computed effectively.

2.2 Envelopes of simple Lie algebras

Let L be a simple Lie algebra over a field \mathbb{F} of characteristic $p > 0$ and let $L \rightarrow \bar{L} : g \mapsto \bar{g}$ denote its embedding into $Der(L)$ via the adjoint action. The Leibniz formula implies that $d^p \in Der(L)$ for every $d \in Der(L)$. Hence $Der(L)$ is a restricted Lie algebra and $\bar{g}^p \in Der(L)$ for every $g \in L$.

The p -envelope $Env(L)$ of L is the smallest Lie subalgebra of $Der(L)$ which contains \bar{L} and is restricted. It is not difficult to observe that \bar{L} is an ideal in $Env(L)$ with abelian quotient $Env(L)/\bar{L}$.

The dimension of $Env(L)$ is an isomorphism invariant of L which can be computed readily from $Der(L)$ by its definition. It exhibits how far L is away from being restricted.

3 Presentations and isomorphisms

In this section we recall some basic features of presentations for Lie algebras. The results of this section apply to finite dimensional Lie algebras over an arbitrary field \mathbb{F} .

2 Theorem: *Let L and \bar{L} be two Lie algebras over the field \mathbb{F} . Let $\langle g_1, \dots, g_d \mid z_1, \dots, z_k \rangle$ be an arbitrary finite presentation for L and let $\{h_1, \dots, h_d\}$ be an arbitrary generating set with d elements for \bar{L} . If $z_i(h_1, \dots, h_d) = 0$ for $1 \leq i \leq k$, then the map $g_i \mapsto h_i$ for $1 \leq i \leq d$ extends to a Lie algebra epimorphism $L \rightarrow \bar{L}$.*

Proof: Let F be the free Lie algebra on d generators x_1, \dots, x_d over the field \mathbb{F} . Let $\alpha : F \rightarrow L : x_i \mapsto g_i$ and $\beta : F \rightarrow \bar{L} : x_i \mapsto h_i$. Then α extends to an epimorphism whose kernel K is generated as ideal by z_1, \dots, z_k . Further, β extends to epimorphism with kernel \bar{K} , say. As $z_i(h_1, \dots, h_d) = 0$ for all i , it follows that $K \leq \bar{K}$. Thus $L \rightarrow \bar{L} : g_i \mapsto h_i$ extends to a Lie algebra epimorphism. •

Next, we recall how a presentation for a Lie algebra on a given generating set can be obtained.

3 Theorem: *Let L be a Lie algebra over a field \mathbb{F} and let g_1, \dots, g_d be an arbitrary generating set for L . Then there exists a finite presentation on the generators g_1, \dots, g_d for L .*

Proof: The proof is constructive and shows how to determine relators for the desired presentation. First, let $\langle b_1, \dots, b_n \mid r_1, \dots, r_k \rangle$ be an arbitrary presentation for L . For example, the structure constants table for L can be used to obtain such a presentation readily.

We define expressions v_i with $g_i = v_i(b_1, \dots, b_n)$ for $1 \leq i \leq d$ and expressions w_i with $b_i = w_i(g_1, \dots, g_d)$ for $1 \leq i \leq n$. Then we replace each b_j in a relator r_i by $w_i(g_1, \dots, g_d)$ to obtain

$$s_i(g_1, \dots, g_d) := r_i(w_1(g_1, \dots, g_d), \dots, w_n(g_1, \dots, g_d))$$

for $1 \leq i \leq k$. Similarly, we define

$$t_i(g_1, \dots, g_d) := v_i(w_1(g_1, \dots, g_d), \dots, w_n(g_1, \dots, g_d)) - g_i$$

for $1 \leq i \leq d$. Let \bar{L} denote the Lie algebra defined by the presentation $\langle g_1, \dots, g_d \mid s_1, \dots, s_k, t_1, \dots, t_d \rangle$. We show that L is isomorphic to \bar{L} .

For this purpose we define the linear map $\alpha : L \rightarrow \bar{L}$ via $\alpha(b_i) = w_i(g_1, \dots, g_d)$. Then α is a Lie algebra homomorphism, since the images of α satisfy the relators of L due to s_1, \dots, s_k . Further, α is surjective due to t_1, \dots, t_d . This also yields that $\bar{L} = \langle w_1, \dots, w_n \rangle$. Next, we define the linear map $\beta : \bar{L} \rightarrow L$ via $\beta(w_i) = b_i$. Again, β is a Lie algebra homomorphism, since the images of β satisfy the relators of \bar{L} . Further, α and β are mutually inverse homomorphism and hence both are isomorphisms. •

4 Isomorphism testing

In this section we describe an approach for testing whether two given Lie algebras L and \bar{L} over a finite field \mathbb{F} are isomorphic.

4.1 Types, bins and efficient generating sets

We first introduce some further notation. For this purpose let L be a Lie algebra over the field \mathbb{F} . Recall that $L \rightarrow \text{Der}(L) : g \mapsto \bar{g}$ is the homomorphism from L to $\text{Der}(L)$ via the adjoint action.

Types:

- The type $t(g)$ for $g \in L$ is the minimal polynomial of its adjoint action \bar{g} .
- The type of (g_1, \dots, g_d) is defined via $t(g_1, \dots, g_d) = (t(g_1), \dots, t(g_d))$.

Bins: Let $U \subseteq L$.

- The bin of $t \in \mathbb{F}[x]$ in U is $b_U(t) = \{g \in U \mid t(g) = t\}$.
- $b_U(t_1, \dots, t_d) = b_U(t_1) \times \dots \times b_U(t_d)$ the cartesian products of sets.
- The bin of $g \in L$ in U is $b_U(g) = b_U(t(g))$.
- $b_U(g_1, \dots, g_d) = b_U(t(g_1), \dots, t(g_d))$.

We say that a generating set g_1, \dots, g_d for L is *d-efficient* there exists no d -element generating set h_1, \dots, h_d for L with $|b_L(h_1, \dots, h_d)| < |b_L(g_1, \dots, g_d)|$. Further, a generating set for L is *efficient*, if it is d -efficient and d is the minimal cardinality for a generating set for L .

The following outlines an algorithm to determine or approximate a d -efficient generating set for a Lie algebra L over a finite field \mathbb{F} . It takes as input the Lie algebra L , the number d and a function $f : \mathbb{N} \rightarrow \mathbb{N}$.

Algorithm: DEfficientGeneratingSet(L, d, f)

- Determine the elements of L and their types.
- Let t_1, \dots, t_r denote the different types of elements in L .
- Sort the elements of L into their bins $b_L(t_1), \dots, b_L(t_r)$.
- Let s_i denote the number of elements in $b_L(t_i)$ for $1 \leq i \leq r$.
- Let $v_1 < \dots < v_s$ be the different values of d -fold products $s_{i_1} \cdots s_{i_d}$.
- For i from 1 to s do:
 - Loop over all d -tuples $(s_{i_1}, \dots, s_{i_d})$ with $i_1 \leq \dots \leq i_d$ and $s_{i_1} \cdots s_{i_d} = v_i$.
 - For each such $(s_{i_1}, \dots, s_{i_d})$ loop over $f(v_i)$ different, randomly chosen elements (g_1, \dots, g_d) in the corresponding bin $b_L(t_{i_1}, \dots, t_{i_d})$.
 - For each (g_1, \dots, g_d) check whether it generates L ;
 - If so, then return (g_1, \dots, g_d) and stop.
- Return fail.

The algorithm requires to loop over all elements of L and to determine types and bins of elements. This limits the range of application of this method. We note that this method performed well on a standard PC for Lie algebras with up to 2^{20} elements.

If f is the identity function, then this algorithm is a deterministic routine to compute a d -efficient generating set for L . The algorithm returns such a generating set or fail; in the latter case there exists no d -element generating set for L . The algorithm requires to loop over various d -tuples of elements in L in this case and hence is fairly time- and space-consuming.

The smaller $f(m)$ is, the more effective is the algorithm, as it has to run over less d -tuples of elements. However, if $f(m) < m$, then the algorithm may return a generating set which is not d -efficient or it may return fail even if there exists a d -element generating set in L . Our experiments suggest that the algorithm is very effective and has a high chance of success if $f(m) = \lfloor \sqrt{m} \rfloor$ is used.

The algorithm can be improved further if a subgroup of $\text{Aut}(L)$ is known. The group $\text{Aut}(L)$ acts on the elements of L and this action leaves the bins invariant. Hence instead of determining and looping over all elements in a bin, it is sufficient to determine representatives for the action of $\text{Aut}(L)$ or a subgroup of this group.

4 Remark: The algorithm ‘DEfficientGeneratingSet’ can be used to determine or approximate an efficient generating set by applying it with increasing d .

4.2 The isomorphism test algorithm

Assume that we are given two Lie algebras L_1 and L_2 over a finite field \mathbb{F} . The aim is to check whether L_1 and L_2 are isomorphic. Our approach to this problem splits into 3 steps. The third step depends on a predefined number $m \in \{d, \dots, |L_2|\}$.

Step 1: Elementary checks

- (a) Check whether $\dim(L_1) = \dim(L_2)$ holds.
- (b) Check whether $\dim(\text{Der}(L_1)) = \dim(\text{Der}(L_2))$ holds.
- (c) Check whether $\dim(\text{Env}(L_1)) = \dim(\text{Env}(L_2))$ holds.

If either of these equalities is not satisfied, then the Lie algebras are not isomorphic and false is returned.

Step 2: Precomputations in L_1

- (a) Determine (or approximate) an efficient generating set (g_1, \dots, g_d) for L_1 .
- (b) Determine a presentation P for L_1 on g_1, \dots, g_d using Lemma 3.

Step 3: Computations in L_2

- (a) Choose a random subset U of L_2 of size m .
- (b) Determine the bin $b := b_U(g_1, \dots, g_d)$.
- (c) Loop over the elements (h_1, \dots, h_d) in b :
 - Check if (h_1, \dots, h_d) generates L_2 .
 - Check if (h_1, \dots, h_d) satisfies the relators of P .
 - If both is satisfied, then return the homomorphism induced by $L_1 \rightarrow L_2 : g_i \mapsto h_i$.
- (d) Return false.

This algorithm either returns an isomorphism from L_1 onto L_2 or false. It is a Monte-Carlo algorithm: it is possible that the algorithm returns false even though L_1 and L_2 are isomorphic. The number m has an impact on the time- and space requirements of the algorithm as well as on the probability of an incorrect result as noted in the following.

5 Remark:

- a) The probability of an incorrect result depends on m . The larger m is, the smaller is this probability.
- b) The time- and space requirements of the algorithm depend on m . The larger m is, the more time and space is needed.
- c) If $m = |L_1|$, then the algorithm always returns a correct result. Hence choosing $m = |L_1|$ yields a deterministic isomorphism test.

We also note that in Step 1 one could use further isomorphism invariants if desired. For example, instead of just using the dimension of the derivations one can use the dimensions of the derived series subalgebras as well as the dimensions of the lower central series subalgebras of the derivations.

Step 2 of the algorithm is used to determine a presentation of L_1 on a (nearly) efficient generating set of L . In our applications we often allow for a full determination of an efficient generating set and the construction of a possibly short presentation in this Step, as this improves the efficiency of Step 3 of the algorithm.

Step 3 tries to identify an isomorphism $L_1 \rightarrow L_2$ using Theorem 2. In principle, this step works with any presentation P for L_1 . However, the fact that P is based on a (nearly) efficient generating set of L_1 implies that Step 3 is effective, since the bin b will be comparatively small.

5 Exponential automorphisms $Exp(L)$

In this section we describe a subgroup $Exp(L)$ of $Aut(L)$ which can be obtained from the derivations of L . We consider $Der(L)$ as Lie subalgebra of $M_{n \times n}(\mathbb{F})$ and we assume that \mathbb{F} is a field of characteristic $p > 0$.

We say that $d \in Der(L)$ is p -nilpotent if $d^p = 0$ holds. Further, we call a derivation d an *annihilator* if d is p -nilpotent and for all $x, y \in L$ and for all i, j with $i, j \geq 0$ and $i + j \geq p$ it satisfies

$$d^i(x)d^j(y) = 0.$$

Let $Ann(L) \subseteq Der(L)$ be the subset of annihilator derivations in $Der(L)$. For a p -nilpotent derivation d let

$$exp(d) = \sum_{i=0}^{p-1} \frac{1}{i!} d^i.$$

6 Lemma: *If $d \in Ann(L)$, then $exp(d)$ is an automorphism of L .*

Proof: We have to show that $exp(d)(xy) = exp(d)(x)exp(d)(y)$ for all $x, y \in L$. This follows from expanding both sides of the equation using the definition of $exp(d)$ and the fact that d is an annihilator derivation. •

We define $Exp(L)$ as the subgroup of $Aut(L)$ generated by $\{exp(d) \mid d \in Ann(L)\}$. Note that $exp(d)$ has order p (unless d and thus $exp(d)$ are trivial) and thus $Exp(L)$ is a subgroup of $Aut(L)$ generated by automorphisms of order p .

The full determination of $Exp(L)$ is often an expensive computation, as it is not straightforward to determine the subset $Ann(L)$ of $Der(L)$. However, by looping over randomly chosen elements of $Der(L)$ one can usually easily find nilpotent elements. A suitable powers of a nilpotent element is then p -nilpotent and thus a candidate for an annihilator. Hence a subgroup of $Exp(L)$ can often be determined readily using random methods.

6 Computing the automorphism group

Every automorphism of a Lie algebra L is also an isomorphism $L \rightarrow L$. Hence we can determine or approximate $Aut(L)$ using the following variation of the method in Section 4.2. Note that we can always skip Step 1 of the algorithm in Section 4.2, since we know *a priori* that our given Lie algebras are isomorphic in this case.

Step 1: Skipped

Step 2: Precomputations in L (similar to Section 4.2)

- (a) Determine (or approximate) an efficient generating set (g_1, \dots, g_d) for L .
- (b) Determine a presentation P for L on g_1, \dots, g_d using Lemma 3.

Step 3: Computations in L (variation on Section 4.2)

- (a) Initialize A as the trivial subgroup of $Aut(L)$.
- (b) Determine the bin $b := b_L(g_1, \dots, g_d)$.
- (c) Loop over m randomly chosen elements (h_1, \dots, h_d) in b :
 - Check if (h_1, \dots, h_d) generates L .
 - Check if (h_1, \dots, h_d) satisfies the relators of P .
 - If both is satisfied, then
 - Let $\alpha : L \rightarrow L : g_i \mapsto h_i$.
 - Reset A to $\langle A, \alpha \rangle$.
- (d) Return A .

If $m = |L|$ is chosen, then $U = L$ follows and the algorithm is a deterministic method to compute $Aut(L)$. If $m < |L|$, then the algorithm determines a subgroup of $Aut(L)$ as approximation of $Aut(L)$.

Improvements to this general method are possible if a subgroup \bar{A} of $Aut(L)$ is known *a priori*. First, such a subgroup can be used in Step (3c) of the algorithm via initializing A with \bar{A} . Secondly, instead of looping over all elements in the bin b , it is sufficient to loop over orbits under the action of \bar{A} . Hence \bar{A} can be used to reduce the amount of work necessary in the algorithm.

A subgroup of the of the exponential automorphisms $Exp(L)$ yields a suitable subgroup of $Aut(L)$ which can often be determined easily.

7 Tensor Lie algebras

Let L be a Lie algebra over a field \mathbb{F} and let \mathbb{E} be an extension field of \mathbb{F} . Then both L and \mathbb{E} are vector spaces over \mathbb{F} and thus we can form the tensor product

$$L \otimes_{\mathbb{F}} \mathbb{E}.$$

This is a vector space of dimension $\dim(L)[\mathbb{E} : \mathbb{F}]$ over \mathbb{F} . It is a Lie algebra via

$$(l_1 \otimes e_1)(l_2 \otimes e_2) = l_1 l_2 \otimes e_1 e_2.$$

We note that if L is centrally simple over \mathbb{F} , then $L \otimes \mathbb{E}$ is simple over \mathbb{F} . The next lemma yields some elementary observations on the derivations of a tensor Lie algebra.

7 Lemma:

- a) The linear map $\alpha : Der(L \otimes \mathbb{E}) \rightarrow Der(L) \otimes \mathbb{E}$ defined by $\alpha(d \otimes e)(x \otimes y) = d(x) \otimes ye$ is an isomorphism.
 b) $\alpha(Env(L \otimes \mathbb{E})) = Env(L) \otimes \mathbb{E}$.
 c) $\alpha(Ann(L \otimes \mathbb{E})) = Ann(L) \otimes \mathbb{E}$.

Proof: a) It is not difficult to verify that $\alpha(d \otimes e)$ is a derivation of $L \otimes \mathbb{E}$ and thus α is well-defined. The image of $d \otimes e$ under α is represented by the Kronecker product of d with the regular representation of e . This also implies that α is injective. To observe that α is surjective, use that $Der(L)$ can be obtained as the nullspace of a matrix as outlined in [2], page 23.

b+c) The map α satisfies that $(\alpha(d \otimes e))^i = \alpha(d^i \otimes e^i)$ for every $i \in \mathbb{N}$. In turn, this implies that α maps annihilators onto tensors with annihilators and the envelope onto the tensor with the envelope. •

We also recall that $Aut(L)$ as well as $Gal(\mathbb{E}/\mathbb{F})$ embeds into $Aut(L \otimes \mathbb{E})$. We omit the straightforward proof.

8 Lemma: $Aut(L) \times Gal(\mathbb{E}/\mathbb{F})$ embeds into $Aut(L \otimes \mathbb{E})$ via $(\alpha, \beta)(l \otimes e) = \alpha(l) \otimes \beta(e)$ for all $l \otimes e \in L \otimes \mathbb{E}$.

The following table contains a list of all currently known simple tensor Lie algebras of dimension at most 20. The list exhibits that $L_1 \otimes \mathbb{E}$ can be isomorphic to $L_2 \otimes \mathbb{E}$ even if L_1 and L_2 are not isomorphic. The table includes the dimension of $Der(L)$ and the orders of $Exp(L)$ and $Aut(L)$ for all Lie algebras.

dim	names	der	aut	exp
3	$W(2)$	5	6	6
6	$W(2) \otimes \mathbb{F}_4$	10	120	60
9	$W(2) \otimes \mathbb{F}_8,$	15	1512	504
12	$W(2) \otimes \mathbb{F}_{16}$	20	16320	4080
15	$W(2) \otimes \mathbb{F}_{32}$	25	163680	32736
18	$W(2) \otimes \mathbb{F}_{64}$	30	1572480	262080
7	$W(3)$	10	16	16
14	$W(3) \otimes \mathbb{F}_4$	20	1536	256
7	V_7	10	4	2
14	$V_7 \otimes \mathbb{F}_4$	20	96	4
8	$W(1, 1)$	8	336	1
8	V_8	8	432	1
16	$W(1, 1) \otimes \mathbb{F}_4 \cong V_8 \otimes \mathbb{F}_4$	16	241920	1
10	$Kap_3(5)$	14	720	720
20	$Kap_3(5) \otimes \mathbb{F}_4$	28	1958400	979200

8 Searching for simple Lie algebras over \mathbb{F}_2

Let L be a given Lie algebra over \mathbb{F}_2 . Then we can readily construct random subquotients of L by choosing two (or more) random elements in L , taking the subalgebra U of L they generate and determining the simple constituents of U . This method is highly effective and usually produces many simple Lie algebras in short time.

It now remains to check whether a newly computed simple Lie algebra L is isomorphic to one of the known Lie algebras. For this purpose we use a datalibrary of the known simple Lie algebras over \mathbb{F}_2 and check an new Lie algebra against this. This can effectively been achieved using the method of Section 4.2.

Within the algorithm of Section 4.2 we assume that L is a Lie algebra from our datalibrary. We also assume that an presentation on an efficient generating set of this Lie algebra is available, so that Step 2 of the algorithm can be skipped. Thus it remains to perform Step 1 (which is usually very fast) and Step 3.

As a result we obtain the following table of currently known simple Lie algebras over \mathbb{F}_2 up to dimension 20. The table exhibits the names of the Lie algebras (and thus also isomorphisms among the known simple Lie algebras) as well as the dimension $Der(L)$, the dimension of $Env(L)$, the order of $Aut(L)$, the number of elements of $Ann(L)$ and the order of $Exp(L)$. For tensor Lie algebras we omit this information and refer to Section 7 instead.

dim	nr	names	der	env	aut	ann	exp
3	1	$W(2)$	5	5	6	4	6
6	1	$W(2) \otimes \mathbb{F}_4$					
7	1	$W(3)$	10	9	16	16	16
7	2	$V_7, P(1, 2)$	10	10	4	2	2
8	1	$A_2, W(1, 1), Q(1, 1, 1)$	8	8	336	1	1
8	2	V_8	8	8	432	1	1
9	1	$W(2) \otimes \mathbb{F}_8, V_9$					
10	1	$Kap_3(5)$	14	14	720	16	720
12	1	$W(2) \otimes \mathbb{F}_{16}$					
14	1	$S(2, 2)$	20	17	1536	80	256
14	2	$W(3) \otimes \mathbb{F}_4$					
14	3	$V_7 \otimes \mathbb{F}_4$					
14	4	$P(1, 1, 1, 1), Kap_1(4)$	20	17	1152	14	1152
14	5	$A_3, B_3, C_3, G_2, S(1, 1, 1), H(1, 1, 1, 1)$	21	14	1451520	64	1451520
14	6	$Bro_2(1, 1)$	20	17	10752	8	64
15	1	$W(2) \otimes \mathbb{F}_{32}$					
15	2	$W(4)$	19	17	2048	1152	2048
15	3	$Kap_3(6), Kap_2(4),$	20	19	23040	32	23040
15	4	$P(2, 1, 1)$	19	19	64	6	16
15	5	$P(3, 1)$	19	18	512	48	64
15	6	$P(2, 2)$	19	19	256	20	64
15	7		19	19	32	4	8
15	8		19	19	192	10	192
16	1	$W(1, 1) \otimes \mathbb{F}_4, A_2 \otimes \mathbb{F}_4, V_8 \otimes \mathbb{F}_4$					
16	2	$W(2, 1), Q(2, 1, 1)$	17	17	2048	24	32
16	3		17	17	1536	4	8
16	4		17	17	384	4	8
16	5		17	17	512	8	8
16	6		17	17	360	1	1
18	1	$W(2) \otimes \mathbb{F}_{64}$					
20	1	$Kap_3(5) \otimes \mathbb{F}_4$					

The new Lie algebras in dimension 15 all arose as subalgebras of $Kap_1(5)$ which has dimension 30. The new Lie algebras in dimension 16 all arose as subalgebras of the Lie algebra of 8×8 -matrices with trace 0 over \mathbb{F}_2 . Explicit generators for the new Lie algebras are exhibited in the following.

- [2] W. DeGraaf. *Lie Algebras: Theory and Algorithms*. North Holland, 2000.
- [3] B. Eick. Computing the automorphism group of a solvable Lie algebra. *Lin. Alg. and Appl.*, 382:195 – 209, 2004.
- [4] G. Hiss. Die adjungierten Darstellungen der Chevalley-Gruppen. *Arch. Math.*, 42:408 – 416, 1984.
- [5] G. M. D. Hogewij. Almost-classical Lie algebras. I, II. *Nederl. Akad. Wetensch. Indag. Math.*, 44(4):441–452, 453–460, 1982.
- [6] D. F. Holt, B. Eick, and E. A. O’Brien. *Handbook of Computational Group Theory*. Discrete Mathematics and its Applications. CRC Press, 2005.
- [7] I. Kaplansky. Some simple Lie algebras of characteristic 2. In *Lie algebras and related topics (New Brunswick, N.J., 1981)*, volume 933 of *Lecture Notes in Math.*, pages 127–129. Springer, Berlin, 1982.
- [8] L. Lin. Nonalternating Hamiltonian algebra $P(n, m)$ of characteristic two. *Comm. Algebra*, 21(2):399–411, 1993.
- [9] C. Schneider. A computer-based approach to the classification of nilpotent Lie algebras. *Experiment. Math.*, 14:153 – 160, 2005.
- [10] H. Strade. *Simple Lie algebras over fields of positive characteristic. I*, volume 38 of *de Gruyter Expositions in Mathematics*. Walter de Gruyter & Co., Berlin, 2004. Structure theory.
- [11] H. Strade and R. Farnsteirner. *Modular Lie Algebras and Their Representations*. Pure and Applied Mathematics. Marcel Dekker, New York, Basel, 1988.
- [12] The GAP Group. *GAP – Groups, Algorithms and Programming, Version 4.4*. Available from <http://www.gap-system.org>, 2005.
- [13] M. Vaughan-Lee. Low dimensional simple lie algebras over $gf(2)$. *Submitted*, 2006.
- [14] Y. Z. Zhang and L. Lin. Lie algebra $K(n, \mu_j, m)$ of Cartan type of characteristic $p = 2$. *Chinese Ann. Math. Ser. B*, 13(3):315–326, 1992. A Chinese summary appears in *Chinese Ann. Math. Ser. A* **13** (1992), no. 4, 516–517.

Bettina Eick

Dec 22, 2009

Institut Computational Mathematics
 TU Braunschweig
 Pockelsstrasse 14
 38106 Braunschweig
 Germany

beick@tu-bs.de