On a conjecture for the dimension of the space of themultiple zeta values

Kaneko, Masanobu
Faculty of Mathematics, Kyushu University

Noro, Masayuki
Department of Mathematics, Graduate School of Science, Kobe University

Tsurumaki, Ken'ichi
Oracle Corporation Japan

出版情報：The IMA volumes in mathematics and its applications. 148, pp.47-58, 2008. Springer
バージョン：

ON A CONJECTURE FOR THE DIMENSION OF THE SPACE OF THE MULTIPLE ZETA VALUES

MASANOBU KANEKO *, MASAYUKI NORO †, AND KEN'ICHI TSURUMAKI‡

Abstract. Since Euler, values of various zeta functions have long attracted a lot of mathematicians. In computer algebra community, Apéry's proof of the irrationality of $\zeta(3)$ is well known. In this paper, we are concerned with the "multiple zeta value (MZV)". More than fifteen years ago, D. Zagier gave a conjecture on MZVs based on numerical computations on PARI. Since then there have been various derived conjectures and two kinds of efforts for attacking them: one is a mathematical proof and another one is a computational experiment to get more confidence to verify a conjecture. We have checked one of these conjectures up to weight $k = 20$, which will be explained later, with Risa/Asir function for non-commutative polynomials and special parallel programs of linear algebra designed for this purpose.

Key words. Multiple zeta value, double shuffle relation, symbolic computation, parallel computation.

AMS(MOS) subject classifications. Primary 14G10, 11M06, 11Y99, 68W30.

1. Introduction. The multiple zeta value (MZV) is a real number defined by the convergent series

$$\zeta(\mathbf{k}) = \zeta(k_1, k_2, \ldots, k_n) = \sum_{m_1 > m_2 > \cdots > m_n > 0} \frac{1}{m_1^{k_1} m_2^{k_2} \cdots m_n^{k_n}}, \qquad (1.1)$$

where $\mathbf{k} = (k_1, k_2, \ldots, k_n)$ is an index set of positive integers with $k_1 > 1$ (which ensures the convergence). In recent years, the MZVs have been appeared in various areas of mathematics and physics and aroused stimulating interest among researchers. In particular, it has become apparent that the structures of (linear or algebraic) relations over the rationals $\mathbf{Q}$ among MZVs reflect properties or structures of various, seemingly unrelated mathematical objects. We refer the readers to Zagier's pioneer work [13] and [2], [12], [8] together with the references therein for more on the subject. In the present paper, we discuss experiments concerning a certain conjecture on the linear relations among MZVs.

For each integer $k \geq 2$, let $\mathcal{Z}_k$ be the $\mathbf{Q}$-vector space spanned by the MZVs $\zeta(k_1, \ldots, k_n)$ whose weight $= k_1 + \cdots + k_n$ is equal to k. In [13], Don Zagier gave a remarkable conjectural formula for $\dim_{\mathbf{Q}} \mathcal{Z}_k$:

$$\dim_{\mathbf{Q}} \mathcal{Z}_k \stackrel{?}{=} d_k,$$

*Graduate School of Mathematics, Kyushu University, 33, Fukuoka 812-8581, Japan
†Department of Mathematics, Graduate School of Science, Kobe University, 1-1, Rokkodai, Nada-ku, Kobe 657-8501, Japan
‡Oracle Corporation Japan

where the number d_k is determined by the Fibonacci-like recurrence

$$d_2 = d_3 = d_4 = 1, \ \ d_k = d_{k-2} + d_{k-3} \quad (k \geq 5).$$

The total number of index sets of weight k is 2^{k-2}, which is much bigger than $d_k \approx 0.4115 \cdots \times (1.3247 \cdots)^k$. For instance, $2^{20-2} = 262144$ whereas $d_{20} = 114$. Hence we expect many linear relations among MZVs of given weight (note that any relations known so far are the relations among MZVs of *same* weight). One of the recent major progress in the theory is that Goncharov [5] and Terasoma [11] independently proved that the number d_k gives an upper bound of $\dim_{\mathbf{Q}} \mathcal{Z}_k$:

THEOREM 1.1 ([5],[11]). *The inequality* $\dim_{\mathbf{Q}} \mathcal{Z}_k \leq d_k$ *holds for all* k*.*

However, their proofs (both relying on the theory of mixed Tate motif) do not give any explicit set of linear relations to reduce the number of generators to the upper bound, and the question as to what sorts of relations are needed for that is still unanswered. Concerning to this question, there is a conjecture in which we are mainly interested:

CONJECTURE 1.1 ([7]). *The extended double shuffle relations suffice to give all linear relations among MZVs.*

A stronger version of this conjecture was proposed by H. N. Minh, M. Petitot et al in [9] and they verified it (in the sense that their proposed relations suffice to reduce $\dim_{\mathbf{Q}} \mathcal{Z}_k$ to d_k) up to weight 16 (private communication). Also, Espie, Novelli and Racinet [3] verified the conjecture up to weight 19 (but modulo powers of π^2 at even weights), in a different context of certain Lie algebra closely related to the "Drinfel'd associator". Our main objective is to verify (a still stronger version of) this conjecture up to weight $k = 20$. In the next section we explain what this conjecture exactly means and introduce an algebraic setup to study the conjecture. Using this setup, we can implement tools for generating relations among MZVs systematically. This will be carried out in Section 3. In Section 4, we will verify the conjecture up to $k = 20$ by using the algebraic tools and special parallel programs of linear algebra. This work follows an experimental computation by Minh and Petitot. We also give a new conjecture in Section 4, which is a refined version of Conjecture 1.1.

Acknowledgements
Prof. Nobuki Takayama gave us sincere encouragements and valuable comments on this work. Discussion with Dr. Naoyuki Shinohara was useful to implement an efficient representation of a non-commutative monomial in Risa/Asir. Prof. Kinji Kimura and Prof. Kazuhiro Yokoyama were interested in our work and provided several computing environments for our experiments.

2. The algebraic formulation. The MZV can be given not only as a sum (1.1) but also as an integral

$$\zeta(k_1,k_2,\ldots,k_n) = \int\cdots\int_{1>t_1>t_2>\cdots>t_k>0} \omega_1(t_1)\omega_2(t_2)\cdots\omega_k(t_k), \tag{2.1}$$

where $k = k_1 + k_2 + \cdots + k_n$ is the weight and $\omega_i(t) = dt/(1-t)$ if $i \in \{k_1, k_1+k_2, \ldots, k_1+k_2+\cdots+k_n\}$ and $\omega_i(t) = dt/t$ otherwise. From each of these representations one finds that the product of two MZVs is a $\mathbf{Z}$-linear combination of MZVs. The first example is

$$\begin{aligned}\zeta(2)^2 &= \left(\sum_{m>0}\frac{1}{m^2}\right)\left(\sum_{n>0}\frac{1}{n^2}\right) = \sum_{m,n>0}\frac{1}{m^2n^2}\\ &= \left(\sum_{m>n>0} + \sum_{n>m>0} + \sum_{m=n>0}\right)\frac{1}{m^2n^2}\\ &= 2\zeta(2,2)+\zeta(4)\end{aligned}$$

and

$$\begin{aligned}\zeta(2)^2 &= \left(\iint_{1>t_1>t_2>0}\frac{dt_1dt_2}{t_1(1-t_2)}\right)\left(\iint_{1>t_1'>t_2'>0}\frac{dt_1'dt_2'}{t_1'(1-t_2')}\right)\\ &= \iiiint_{\substack{1>t_1>t_2>0\\1>t_1'>t_2'>0}}\frac{dt_1dt_2dt_1'dt_2'}{t_1(1-t_2)t_1'(1-t_2')}\\ &= 4\iiiint_{1>s_1>s_2>s_3>s_4>0}\frac{ds_1ds_2ds_3ds_4}{s_1s_2(1-s_3)(1-s_4)}\\ &\quad + 2\iiiint_{1>s_1>s_2>s_3>s_4>0}\frac{ds_1ds_2ds_3ds_4}{s_1(1-s_2)s_3(1-s_4)}\\ &= 4\zeta(3,1)+2\zeta(2,2),\end{aligned}$$

and this gives $\zeta(4) = 4\zeta(3,1)$. The point here is that the two expressions obtained are always different, and thus their equality gives a collection of linear relations among MZVs which we call the *finite double shuffle relations* (FDS). Moreover, one can extend the finite double shuffle relations by taking divergent sums and integrals into account together with a certain regularization procedure. We call these generalized relations the *extended double shuffle relations* (EDS), and the conjecture is that the EDS suffices to give all linear relations among MZVs (Conjecture 1.1).

The two multiplication rules mentioned above are described in a purely algebraic manner, as given in Hoffman [6]. Let $\mathfrak{H} = \mathbf{Q}\langle x, y\rangle$ be the non-commutative polynomial algebra over the rationals in two indeterminates

x and y, and $\mathfrak{H}^1$ and $\mathfrak{H}^0$ its subalgebras $\mathbf{Q}+\mathfrak{H}y$ and $\mathbf{Q}+x\mathfrak{H}y$, respectively. Let $Z:\mathfrak{H}^0\to\mathbf{R}$ be the $\mathbf{Q}$-linear map ("evaluation map") which assigns to each word (monomial) $u_1u_2\cdots u_k$ ($u_i\in\{x,y\}$) in $\mathfrak{H}^0$ the multiple integral

$$\int\cdots\int_{1>t_1>t_2>\cdots>t_k>0} \omega_{u_1}(t_1)\omega_{u_2}(t_2)\cdots\omega_{u_k}(t_k)$$

where $\omega_x(t)=dt/t$, $\omega_y(t)=dt/(1-t)$. We set $Z(1)=1$. Since the word $u_1u_2\cdots u_k$ is in $\mathfrak{H}^0$, we always have $\omega_{u_1}(t) = dt/t$ and $\omega_{u_k}(t) = dt/(1-t)$, so the integral converges. By the integral representation (2.1), we have

$$Z(x^{k_1-1}yx^{k_2-1}y\cdots x^{k_n-1}y)=\zeta(k_1,k_2,\ldots,k_n).$$

The weight $k=k_1+k_2+\cdots+k_n$ of $\zeta(k_1,k_2,\ldots,k_n)$ is the total degree of the corresponding monomial $x^{k_1-1}yx^{k_2-1}y\cdots x^{k_n-1}y$.

Let $z_k := x^{k-1}y$, which corresponds under Z to the Riemann zeta value $\zeta(k)$. Then $\mathfrak{H}^1$ is freely generated by z_k ($k=1,2,3,\ldots$). We define the *harmonic product* $*$ on $\mathfrak{H}^1$ inductively by

$$1*w=w*1=w,\ z_kw_1*z_lw_2=z_k(w_1*z_lw_2)+z_l(z_kw_1*w_2)+z_{k+l}(w_1*w_2),$$

for all $k,\,l\geq 1$ and any words $w,\,w_1,\,w_2\in\mathfrak{H}^1$, and then extending by $\mathbf{Q}$-bilinearity. Equipped with this product, $\mathfrak{H}^1$ becomes a commutative algebra ([6]) and $\mathfrak{H}^0$ a subalgebra. The first multiplication law of MZVs asserts that the evaluation map $Z:\mathfrak{H}^0\to\mathbf{R}$ is an algebra homomorphism with respect to the multiplication $*$, i.e.,

$$Z(w_1*w_2)=Z(w_1)Z(w_2)\quad(w_1,w_2\in\mathfrak{H}^0). \tag{2.2}$$

For instance, the harmonic product $z_k*z_l=z_kz_l+z_lz_k+z_{k+l}$ corresponds to the identity $\zeta(k)\zeta(l)=\zeta(k,l)+\zeta(l,k)+\zeta(k+l)$.

The other commutative product ш, referred to as the *shuffle product*, corresponding to the product of two integrals in (2.1), is defined on all of $\mathfrak{H}$ inductively by setting

$$1ш w=wш 1=w,\ uw_1ш vw_2=u(w_1ш vw_2)+v(uw_1ш w_2),$$

for any words $w,\,w_1,\,w_2\in\mathfrak{H}$ and $u,\,v\in\{x,y\}$, and again extending by $\mathbf{Q}$-bilinearity. The character 'ш' is the Cyrillic *sha*, standing for *shuffle.* This product gives $\mathfrak{H}$ the structure of a commutative $\mathbf{Q}$-algebra ([10]) which we denote by $\mathfrak{H}_ш$. Obviously the subspaces $\mathfrak{H}^1$ and $\mathfrak{H}^0$ become subalgebras of $\mathfrak{H}_ш$, denoted by $\mathfrak{H}^1_ш$ and $\mathfrak{H}^0_ш$ respectively. By the standard shuffle product identity of iterated integrals, the evaluation map Z is again an algebra homomorphism for the multiplication ш:

$$Z(w_1ш w_2)=Z(w_1)Z(w_2)\quad(w_1,w_2\in\mathfrak{H}^0). \tag{2.3}$$

By equating (2.2) and (2.3), we get the *finite double shuffle relations* (FDS) of MZV:

$$Z(w_1 ш w_2 - w_1 * w_2) = 0 \quad (w_1, w_2 \in \mathfrak{H}^0). \tag{2.4}$$

These relations, however, do not suffice to obtain all relations. For instance, there are two MZVs of weight 3, $\zeta(3)$ and $\zeta(2,1)$. As Euler already showed in [4], these two values are actually equal, and therefore $\dim_{\mathbf{Q}} \mathcal{Z}_3 = 1$. But the least weight of FDS is 4 so that we cannot deduce the relation $\zeta(3) = \zeta(2,1)$ by the FDS. In order to obtain more relations, we introduce a "regularization" procedure which amounts to incorporate the divergent MZVs into the picture. In the following we only give a minimum of what we need to formulate the extended double shuffle relations, and the interested readers are invited to consult the paper [7].

It is known (cf. [10]) that the commutative algebra $\mathfrak{H}^1_ш$ is isomorphic to the polynomial algebra over $\mathfrak{H}^0_ш$ in y:

$$\mathfrak{H}^1_ш \simeq \mathfrak{H}^0_ш[y]. \tag{2.5}$$

Let $\mathrm{reg}_ш$ ("regularization map") be the map from $\mathfrak{H}^1_ш$ to $\mathfrak{H}^0_ш$ obtained by taking the "constant term" with respect to y under the isomorphism (2.5):

$$\mathrm{reg}_ш : \mathfrak{H}^1_ш \ni w = \sum_{i=0}^{n} w_i ш y^{ш\, i} \mapsto w_0 \in \mathfrak{H}^0_ш.$$

Note that the map $\mathrm{reg}_ш$ is the identity if restricted to the subspace $\mathfrak{H}^0_ш$. We then have the following theorem. For a proof, we refer the reader to [7].

Theorem 2.1 (extended double shuffle relations (EDS), [7]). *For any $w_1 \in \mathfrak{H}^1$ and $w_0 \in \mathfrak{H}^0$, we have*

$$Z\big(\mathrm{reg}_ш(w_1 ш w_0 - w_1 * w_0)\big) = 0.$$

When $w_1 \in \mathfrak{H}^0$, the above relation reduces to the finite double shuffle relation and so the EDS contains the FDS. If we take $w_1 = y$ as a particular case, it can be shown that the element $y ш w_0 - y * w_0$ is always in $\mathfrak{H}^0$ and hence we obtain the relation (without the regularization)

$$Z(y ш w_0 - y * w_0) = 0 \quad (w_0 \in \mathfrak{H}^0). \tag{2.6}$$

If we substitute $x^{k_1-1}yx^{k_2-1}y\cdots x^{k_n-1}y$ for w_0 in this relation and expand it by using the definitions of ш and $*$, we readily obtain the relation known as Hoffman's relation:

$$\sum_{i=1}^{n} \zeta(k_1,\ldots,k_{i-1},k_i+1,k_{i+1},\ldots,k_n) = \sum_{1\le l\le n, k_l\ge 2} \sum_{j=0}^{k_l-2} \zeta(k_1,\ldots,k_{l-1},k_l-j,j+1,k_{l+1},\ldots,k_n). \tag{2.7}$$

3. Implementation of the ring of bivariate non-commutative polynomials.

3.1. Prototyping. The algebra $\mathbf{Q}\langle x, y\rangle$ is a non-commutative polynomial ring. There are several computer algebra systems supporting non-commutative polynomial ring and we have tried some of them. For example Mathematica can calculate non-commutative polynomials, however it did not behave as we hoped and it was not efficient. Next we tried representing a non-commutative monomial as a list and wrote a short program to implement it. But it was not efficient because a list representation needs many links by pointers and even a simple monomial operation requires a considerable number of memory operations. Then we tried representing a monomial by a character string. It may seem more naive than the list representation, but in fact it can be efficiently realized because the product of two monomials can be computed by the concatenation of two character strings, and the comparison can be done by comparing the character strings according to a specific ordering, say, the lexicographic ordering. By using this method, we implemented the operations in $\mathbf{Q}\langle x, y\rangle$ by the user language of Risa/Asir. The program amounts to only 160 lines.

3.2. Implementation as a built-in data type. Based on the prototyping, we decided to implement $\mathbf{Q}\langle x, y\rangle$ as a built-in data type in Risa/Asir for our large scale experiments. In this implementation, a monomial is represented as a bit sequence. All the fundamental operations in $\mathbf{Q}\langle x, y\rangle$, including the shuffle and harmonic products are implemented as built-in functions written in C. Preliminary experiments show that they are more efficient than the prototype, nevertheless the speed-up is about a factor of 10 and the prototype by general facilities is proved to be efficient enough for small experiments.

An element in $\mathbf{Q}\langle x, y\rangle$ is converted from a `QUOTE` object, which is also a built-in data type for expressing general non-commutative objects. A `QUOTE` object holds a tree structure converted from an input expression. It preserves the order of products and we can convert it to an element in $\mathbf{Q}\langle x, y\rangle$ by calling a built-in function `qt_to_nbp()`. After computing polynomials giving EDS relations of a fixed weight k, we convert them to integer vectors via the monomial basis consisting of all weight k monomials in $\mathfrak{H}^0$. In order to make the conversion efficient, we provide a function to compute the index of a monomial in the monomial basis. Note that the index can be computed easily if we apply the lexicographic ordering. `http://www.math.kobe-u.ac.jp/OpenXM/Math/MZV/mzv_dsr.rr` is a simple Risa/Asir program to compute a set of generators of $\mathcal{Z}_k$ and to represent remaining MZVs of weight k as linear combinations of the generators. The function `dsr_matrix(k)` returns a matrix constructed from a specific subset of EDS relations of weight k, which will be explained in Section 4.1. Each row of the matrix gives the coefficients of MZVs in an EDS relation, where MZVs are indexed according to the lexicographic or-

$$\left(\begin{array}{cccccccccccccccccccccccccccccccc}
1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & -1 \\
\hline
-1 & 10 & 4 & 0 & 3 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 12 & -1 & 4 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 6 & 6 & 0 & -1 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & 12 & 0 & -1 & 0 & 4 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 4 & 2 & 0 & 4 & 0 & 0 & 0 & -1 & 6 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 4 & 0 & 0 & 0 & -1 & 0 & 6 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 6 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 1 & 0 & -1 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 3 & 0 & -1 & 0 & 2 & -1 & 1 & 0 & 0 \\
-1 & 20 & 10 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 18 & 0 & 9 & 3 & 0 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 6 & 6 & 0 & 0 & 8 & 4 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & 12 & 0 & 0 & 0 & 6 & 0 & 4 & 2 & 0 & 0 & -1 & 0 & 2 & 0 & 2 & 1 & 0 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 12 & 0 & 6 & 3 & 0 & 0 & 4 & 2 & 0 & 2 & 0 & 0 & 0 & -1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & -2 & 0 & 0 & 0 & 18 & 0 & -1 & -1 & 7 & 0 & 2 & 0 & 0 & 0 & -1 & -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 6 & 0 & 8 & 6 & 0 & 0 & -1 & -1 & 6 & -2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & -1 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & -1 & -3 & 0 & -2 & -2 & 4 & 0 & -1 & -1 & 1 & 0 & 0 & 0 & 0
\end{array}\right)$$

FIG. 1. *The output of* `dsr_matrix(7)`

dering. FIG. 1 is the output of `dsr_matrix(7)`. For example, the first row in FIG. 1 shows a relation

$$\zeta(7) - \zeta(6,1) - \zeta(5,2) - \zeta(4,3) - \zeta(3,4) - \zeta(2,5) = 0.$$

The function `dsr_basis(k)` returns a list $[[M_1, P_1], \ldots, [M_i, P_i], \ldots]$. In this output M_i and P_i are given as elements of $\mathbf{Q}\langle x, y\rangle$. Each pair $[M_i, P_i]$ means $Z(M_i) = Z(P_i)$, which represents an MZV $Z(M_i)$ as a linear combination of the generators of $\mathcal{Z}_k$ computed from the input matrix given by `dsr_matrix(k)`. The function `nbp_to_mzv(F)` converts a polynomial F in $\mathbf{Q}\langle x, y\rangle$ to a linear combination of MZVs. In the output of this function, an MZV is given as a list $[k_1, k_2, \ldots]$ which represents $\zeta(k_1, k_2, \ldots)$. The following example shows that $\mathcal{Z}_6$ is generated by

$$\{Z(xyyyxy), Z(xyyyyy)\} = \{\zeta(2,1,1,2), \zeta(2,1,1,1,1)\}.$$

For example, the first element of the output means

$$\zeta(2,1,2,1) = -\frac{1}{2}\zeta(2,1,1,2) + \frac{13}{24}\zeta(2,1,1,1,1).$$

Note that the number of generators coincides with $d_6 = 2$.

```
[0] load("./mzv_dsr.rr")$
[7] map(print,dsr_basis(6))$
[(1)*xyyxyy,(-1/2)*xyyyxy+(13/24)*xyyyyy]
[(1)*xyyxxy,(-1)*xyyyxy+(61/48)*xyyyyy]
```

```
[(1)*xyxyyy,(-1)*xyyyxy+(3/4)*xyyyyy]
[(1)*xyxyxy,(3/16)*xyyyyy]
[(1)*xyxxyy,(3/2)*xyyyxy+(-11/12)*xyyyyy]
[(1)*xyxxxy,(1)*xyyyxy]
[(1)*xxyyyy,(1/2)*xyyyxy+(-7/24)*xyyyyy]
[(1)*xxyyxy,(3/2)*xyyyxy+(-11/12)*xyyyyy]
[(1)*xxyxyy,(-3)*xyyyxy+(97/48)*xyyyyy]
[(1)*xxyxxy,(-1/2)*xyyyxy+(13/24)*xyyyyy]
[(1)*xxxyyy,(1)*xyyyxy+(-31/48)*xyyyyy]
[(1)*xxxyxy,(-1)*xyyyxy+(3/4)*xyyyyy]
[(1)*xxxxyy,(1/2)*xyyyxy+(-7/24)*xyyyyy]
[(1)*xxxxxy,(1)*xyyyyy]
```

4. Experimental results and a new conjecture. In this section, we numerically verify Conjecture 1.1 up to $k = 20$, which seems to be a world record. Exactly speaking we show that the set of EDS relations reduces the upper bound of $\dim_{\mathbf{Q}} \mathcal{Z}_k$ to d_k. That is, the verification is to check $\dim_{\mathbf{Q}} DS^{(k)} \geq 2^{k-2} - d_k$, where $DS^{(k)}$ denotes the $\mathbf{Q}$-vector space spanned by all EDS relations of weight k. There are several conjectures on the set of relations sufficient for reducing the dimension to d_k. Minh, Jacob, Oussous and Petitot [9] conjecture that Hoffman's relations and the FDS suffice and they checked it up to $k = 16$. We shall check (a stronger version of) their conjecture up to $k = 20$. We choose the weight $k = 20$ as our final target because the verification of the case is hard with respect to both the time and space complexity but it is expected that it is executable on an ordinary computing environment, if we apply well-known techniques such as elimination by sparse rows to reduce the size of a matrix, computation over a finite field and parallel Gaussian elimination. Detailed explanation of these techniques will be given in Section 4.2, 4.3 and 4.4 respectively.

4.1. Generation of the double shuffle relations. We denote the sets of polynomials giving all FDS and Hoffman's relations of weight k by $F^{(k)}$ and $E^{(k)}$ respectively. $F^{(k)}$ is given by $F^{(k)} = \bigcup_{i=2}^{\lfloor k/2 \rfloor} F_i^{(k)}$, where

$$F_i^{(k)} = \{w_1 ш w_2 - w_1 * w_2 \mid w_1 \in \mathfrak{M}_i^0, w_2 \in \mathfrak{M}_{k-i}^0\}$$

and $\mathfrak{M}_i^0$ denotes the set of all monomials of weight i in $\mathfrak{H}^0$. Denoting the cardinality of a set S by $|S|$, $|\mathfrak{M}_i^0| = 2^{i-2}$ implies $|F_i^{(k)}| = 2^{k-4}$ for $i < k/2$. If k is even, $|F_{k/2}^{(k)}| = \frac{2^{k/2-2}(2^{k/2-2}+1)}{2}$, and $|E^{(k)}| = 2^{k-3}$. Thus we have

$$|F^{(k)}| + |E^{(k)}| > (\lfloor \frac{k}{2} \rfloor - 2)2^{k-4} + 2^{k-3}. \tag{4.1}$$

In particular we have $|F^{(k)}| + |E^{(k)}| > 2^{k-2}$ for $k \geq 8$. The verification of the conjecture is reduced to the rank computation of a matrix $M^{(k)}$ constructed from the coefficients of the relations. The inequality (4.1) means that the number of rows of $M^{(k)}$ is greater than that of columns and their ratio increases if k becomes large. Our purpose is to show $\mathrm{rank}(M^{(k)}) \geq 2^{k-2} - d_k$ and it is sufficient to show this inequality for a sub-matrix of $M^{(k)}$. Our experiments for small k indicate that

$$R^{(k)} = E^{(k)} \cup F_2^{(k)} \cup F_3^{(k)}$$

suffices to attain the lower bound of the rank and we are led to a new conjecture:

CONJECTURE 4.1. *$R^{(k)}$ suffices to reduce the upper bound of* $\dim_{\mathbf{Q}} \mathcal{Z}_k$ *to d_k.*

If $k \geq 7$, $|R^{(k)}|$ is equal to 2^{k-2} and the matrix constructed from $R^{(k)}$ is a square matrix. As d_k is much smaller than 2^{k-2}, $R^{(k)}$ is practically optimal for our experimental verification. The FDS relations are generated by shuffle and harmonic products implemented in Risa/Asir. Hoffman's relations can be generated either by (2.6) or by the explicit representation (2.7). FIG. 1 actually shows the matrix constructed from $R^{(7)}$. Its upper-half part comes from Hoffman's relations and the lower-half part comes from the FDS relations. The matrix data for $k \leq 20$ are available from `http://www.math.kobe-u.ac.jp/OpenXM/Math/MZV/matdata`. For each k, $E^{(k)}$, $F_2^{(k)}$ and $F_3^{(k)}$ are converted to matrices and stored as files `mh`k, `mfds`k`_2` and `mfds`k`_3` respectively. These files are written according to the following format:

$$\boxed{(r\ c)\ (l_1\ \cdots\ l_r)\ (j_{1,1} a_1)\ \cdots\ (j_{1,l_1} a_{l_1})\ \cdots\ (j_{r,1} a_{s+1})\ \cdots\ (j_{r,l_r} a_{s+l_r})}$$

where (r, c) denotes the size of the matrix, l_i denotes the number of non-zero elements in the i-th row and $(j_{i,k}, a_t)$ denotes a non-zero element a_t at $(i, j_{i,k})$. That is, non-zero elements are stored in row-major order. In the file, all numbers are four-byte integers and they are represented according to the network byte order.

4.2. Preprocessing by Hoffman's relations. If we convert $R^{(k)}$ itself to a dense matrix, then we need huge memory for a large k. Fortunately the matrix contains a large number of sparse rows coming from $E^{(k)}$ and we can apply preprocessing to eliminate many matrix entries with a small cost. The coefficients of $f \in E^{(k)}$ are 1 or -1 and the number of terms in f is $k-1$. Furthermore, under the lexicographic ordering we have the following (cf. FIG. 1):

PROPOSITION 4.1. *For any $xxw \in \mathfrak{M}_k^0$ there uniquely exists $f \in E^{(k)}$ such that the leading term of f is xxw.*

That is, the left-half of the $2^{k-3} \times 2^{k-2}$ sub-matrix obtained from $E^{(k)}$ is already upper triangular with unit diagonals. Thus we can easily eliminate the left-half of the sub-matrix obtained from $F_2^{(k)}$ and $F_3^{(k)}$ by using $E^{(k)}$. Note that this is a sparse elimination and it can be done efficiently. After this operation, we have the lower-right sub-matrix which

k	16	17	18	19	20
size	128MB	512MB	2GB	8GB	32GB

TABLE 1
Required size of memory for storing $S^{(k)} \bmod p$

is denoted by $S^{(k)}$. We set $N_k = 2^{k-3}$. $S^{(k)}$ is an $N_k \times N_k$ matrix and what we have to show is $\mathrm{rank}(S^{(k)}) \geq N_k - d_k$.

4.3. Rank computation over finite fields. The coefficients of the polynomials in $F_2^{(k)}$ and $F_3^{(k)}$ are within one machine word for $k \leq 20$. However, the result of the whole Gaussian elimination will have larger entries and it will be hard to execute the rank computation over the rationals. For any prime p we have $\mathrm{rank}(S^{(k)}) \geq \mathrm{rank}(S^{(k)} \bmod p)$. Therefore it is sufficient for our purpose to show $\mathrm{rank}(S^{(k)} \bmod p) \geq N_k - d_k$ for some prime p. In our experiments we use a two-byte prime $p = 31991$ for computing $\mathrm{rank}(S^{(k)} \bmod p)$. TABLE 1 shows the sizes of memory required for storing $S^{(k)} \bmod p$. If a machine is not equipped with sufficient amount of memory, then a further preprocessing may be necessary. By examining $S^{(k)}$ for $k \leq 20$, we find that there are $N_k/4 \times N_k$ sparse sub-matrix whose left-quarter part is upper triangular. We can apply a sparse elimination by this sub-matrix, which results in a $3/4 \cdot N_k \times 3/4 \cdot N_k$ matrix. For example, 32GB of memory is required to store $S^{(20)} \bmod p$. If the second preprocessing is not applied then it is practically hard to execute the computation on a machine with just 32GB of memory, which is one of our computing environments. If we apply the preprocessing, then the required size is reduced to 18GB and we can efficiently compute the rank on that machine. We note that there are probabilistic Wiedemann-Krylov type algorithms to compute the rank of a matrix over finite fields. In general these are more efficient than Gaussian elimination in sparse cases and it is interesting to compare their performances in our cases.

4.4. Parallel computation by MPI. Even if we apply the preprocessing to reduce the size of the matrices, they are still huge and it is necessary to apply parallel computation from a practical point of view. We wrote two parallel programs by MPI to execute Gaussian elimination over finite fields. The implemented method is a simplification of the one used in ScaLAPACK [1]. That is, the whole matrix is decomposed according to 1- or 2-dimensional cyclic distribution algorithm and a non-blocking Gaussian elimination is executed in parallel.
`http://www.math.kobe-u.ac.jp/OpenXM/Math/MZV/gauss.c` is a C code for computing the rank of a matrix over a small finite field by using MPI. Each processor element reads the same input files containing fragments of the whole matrix, and stores a subset of rows of the input matrix into its local memory. At each step of eliminating a column, informations of the rows in the local matrices are shared among all processor elements to de-

k	16	17	18	19	20
d_k	37	49	65	86	114
N_k	8192	16384	32768	65536	131072
$\mathrm{rank}(S^{(k)} \bmod p)$	8155	16335	32703	65450	130958
$N_k - \mathrm{rank}(S^{(k)} \bmod p)$	37	49	65	86	114
Generation (1CPU)	22sec	85sec	4.5min	11min	30min
Preprocessing (1CPU)	5sec	19sec	1.5min	9min	57min
Elimination (8CPU)	2min	13min	1.3hour	9hour	67hour
Total memory	72MB	288MB	1.2GB	4.6GB	18GB

TABLE 2
Statistics of the rank computation in the environment (3)

termine the pivot row. Then the selected pivot row is broadcasted to all processor elements and the elimination is done locally in each processor element. Both the algorithm and the implementation are not so optimized but the performance is satisfactory. We used several computing environments: (1) a cluster of heterogeneous linux PCs, (2) a cluster of large SMP Sparc/Solaris machines, and (3) an SMP linux PC. The last one is an 8 CPU SMP machine with two quad-core Intel X5355/2.66GHz CPUs and 32GB of memory. TABLE 2 shows various statistics in the last environment up to $k = 20$. In the table, "Generation", "Preprocessing" and "Elimination" show the elapsed time for generating all relations, preprocessing by the sparse elimination and Gaussian elimination respectively. "Total memory" shows the size to store the $3/4 \cdot N_k \times 3/4 \cdot N_k$ matrix. The table shows that $\mathrm{rank}(S^{(k)} \bmod 31991) = N_k - d_k$ up to $k = 20$, thus the conjecture is verified up to $k = 20$. The table also tells us that the preprocessing is almost negligible compared with the final Gaussian elimination. We note that we also tried the same computations in the second environment with another parallel program and $p = 16381$, and that we obtained the same results of the ranks.

REFERENCES

[1] L.S. Blackford et al., *ScaLAPACK Users' Guide*, SIAM (1997), `http://www.netlib.org/scalapack/slug/`.
[2] P. Cartier, *Fonctions polylogarithmes, nombres polyzetas et groupes pro-unipotents*, Séminaire Bourbaki, 53éme année, 2000–2001, **885** (2001).
[3] M. Espie, J-C. Novelli, and G. Racinet, *Formal computations about multiple zeta values*, in "From Combinatorics to Dynamical Systems" (Strasbourg, 2002), IRMA Lect. Math. Theor. Phys. 3, F. Fauvet and C. Mitschi (eds.), de Gruyter, Berlin, (2003), 1–16.
[4] L. Euler, *Meditationes circa singulare serierum genus*, Novi Comm. Acad. Sci. Petropol **20** (1775), 140–186, reprinted in Opera Omnia ser. I, vol. 15, B. G. Teubner, Berlin (1927), 217–267.
[5] A. B. Goncharov, *Periods and mixed motives*, preprint (2002).
[6] M. Hoffman, M., *The algebra of multiple harmonic series*, J. Algebra **194** (1997),

477–495.

[7] K. Ihara, M. Kaneko and D. Zagier, *Derivation and double shuffle relations for multiple zeta values*, Compositio Math. **142-02** (2006), 307–338.

[8] M. Kaneko, *Multiple zeta values* (translation of Sugaku 54 (2002), no. 4, 404–415), Sugaku Expositions **18** (2005), no. 2, 221–232.

[9] H. N. Minh, G. Jacob, N. E. Oussous and M. Petitot, *Aspects combinatoires des polylogarithmes et des sommes d'Euler-Zagier*, J. Électr. Sém. Lothar. Combin. **43** (2000), Art. B43e, 29 pp.

[10] C. Reutenauer, *Free Lie Algebras*, Oxford Science Publications,1993.

[11] T .Terasoma, *Mixed Tate motives and multiple zeta values*, Invent. Math. **149** (2002), 339–369.

[12] M. Waldschmidt, *Valeurs zeta multiples: une introduction*, J. Theor. Nombres Bordeaux, **12** (2000), 581–595.

[13] D. Zagier, *Values of zeta functions and their applications*, in ECM volume, Progress in Math., **120** (1994), 497–512.