View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Dagstuhl Research Online Publication Server

Fast Multi-Subset Transform and Weighted Sums over Acyclic Digraphs

Mikko Koivisto
Department of Computer Science, University of Helsinki, Finland
mikko.koivisto@helsinki.fi

Antti Röyskö
Department of Computer Science, University of Helsinki, Finland
antti.roysko@helsinki.fi

Abstract

The zeta and Moebius transforms over the subset lattice of n elements and the so-called subset convolution are examples of unary and binary operations on set functions. While their direct computation requires $O(3^n)$ arithmetic operations, less naive algorithms only use $2^n\text{poly}(n)$ operations, nearly linear in the input size. Here, we investigate a related n-ary operation that takes n set functions as input and maps them to a new set function. This operation, we call multi-subset transform, is the core ingredient in the known inclusion–exclusion recurrence for weighted sums over acyclic digraphs, which extends Robinson's recurrence for the number of labelled acyclic digraphs. Prior to this work, the best known complexity bound for computing the multi-subset transform was the direct $O(3^n)$. By reducing the task to rectangular matrix multiplication, we improve the complexity to $O(2.985^n)$.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Bayesian networks, Moebius transform, Rectangular matrix multiplication, Subset convolution, Weighted counting of acyclic digraphs, Zeta transform

Digital Object Identifier 10.4230/LIPIcs.SWAT.2020.29

Funding This work was partially supported by the Academy of Finland, Grant 316771.

Acknowledgements We thank Petteri Kaski for valuable discussions about the topic of the paper.

1 Introduction

In this paper, we consider the following problem. We are given a finite set U and, for each element $i \in U$, a function f_i from the subsets of U to some ring $\mathcal{R}$. The task is to compute the function g given by

$$g(T) = \sum_{S \subseteq T} \prod_{i \in T} f_i(S)\,, \quad T \subseteq U\,. \tag{1}$$

We shall call g the *multi-subset transform* of $(f_i)_{i \in U}$. While the present study of this operation on set functions stems from a particular application to weighted counting of acyclic digraphs, which we will introduce later in this section, we believe the multi-subset transform could also have applications elsewhere.

A straightforward computation of the multi-subset transform requires $\Omega(3^n)$ arithmetic operations (i.e., additions and multiplications in the ring $\mathcal{R}$) when U has n elements. In the light of the input size $O(2^n n)$ and output size $O(2^n)$, one could hope for an algorithm that requires $2^n n^{O(1)}$ operations. Some support for optimism is provided by the close relation to two similar operations on set functions: the zeta transform of f and the subset convolution of f_1 and f_2, given respectively by

$$(f\zeta)(T) = \sum_{S \subseteq T} f(S) \quad \text{and} \quad (f_1 * f_2)(T) = \sum_{S \subseteq T} f_1(S) f_2(T \setminus S)\,, \quad T \subseteq U\,;$$

© Mikko Koivisto and Antti Röyskö;
licensed under Creative Commons License CC-BY

these unary and binary operations can be performed using $O(2^n n)$ [18, 12] and $O(2^n n^2)$ [2] arithmetic operations, thus significantly beating the naive $\Omega(3^n)$-computation. Indeed, consider the seemingly innocent replacement of "$i \in T$" by "$i \in S$" or "$i \in U$" in (1): either one would yield a variant that immediately (and efficiently) reduces to the zeta transform. Likewise, replacing the factor $\prod_{i \in T \setminus S} f_i(S)$ in the product by $\prod_{i \in T \setminus S} f_i(T \setminus S)$ would give us an instance of subset convolution. The present authors do not see how to fix these "broken reductions" – the multi-subset transform could be a substantially harder problem not admitting a nearly linear-time algorithm. One might even be tempted to hypothesize that one cannot reduce the base of the exponential complexity below the constant 3. We refute this hypothesis:

▶ **Theorem 1.** *The multi-subset transform can be computed using $O(2.985^n)$ arithmetic operations.*

We obtain our result by a reduction to *rectangular matrix multiplication (RMM)*. The basic idea is to split the ground set U into two halves U_1 and U_2 and divide the product over $i \in T$ into two smaller products accordingly. In this way we can view (1) as a matrix product of dimensions $2^{|U_1|} \times 2^{|U|} \times 2^{|U_2|}$. The two rectangular matrices are sparse, with at most $6^{n/2} = O(2.4495^n)$ non-zero elements out of the total $8^{n/2}$. The challenge is to exploit the sparsity. Known algorithms for general sparse matrix multiplication [19, 11] turn out to be insufficient for getting beyond the $O(3^n)$ bound (see Section 2.1 for details). Fortunately, in our case the sparsity occurs in a special, structured form that enables better control of zero-entries, and thereby a more efficient reduction to dense RMM. To get the best available constant base in the exponential bound, we call upon the recently improved fast RMM algorithms [7].

1.1 Application to weighted counting of acyclic digraphs

Let a_n be the number of labeled acyclic digraphs on n nodes. Robinson [14] and Harary and Palmer [10], independently discovered the following inclusion–exclusion recurrence:

$$a_n = \sum_{s=1}^{n} (-1)^{s-1} \binom{n}{s} 2^{s(n-s)} a_{n-s} \,.$$

To see why the formula holds, view s as the number of sinks (i.e., nodes with no out-neighbors), each of which can choose its in-neighbors freely form the remaining $n - s$ nodes.

Tian and He [16] generalized the recurrence to weighted counting of acyclic digraphs on a given set of n nodes V. Now every acyclic digraph D on V is assigned a *modular* weight, that is, a real-valued weight $w(D)$ that factorizes into node-wise weights $w_i(D_i)$, where $D_i \subseteq V \setminus \{i\}$ is the set of in-neighbors of node i in D. This counting problem has applications particularly in Bayesian learning of Bayesian networks from data; the weighted count is the partition function of a statistical model that associates each node of the graph with a random variable, and evaluating the partition function is the main computational bottleneck [6, 16, 15]. Letting a_V denote the weighted sum of acyclic digraphs on V, we have

$$a_V = \sum_{D} \prod_{i \in V} w_i(D_i) = \sum_{\emptyset \neq S \subseteq V} (-1)^{|S|-1} \left(\prod_{i \in S} \sum_{D_i \subseteq V \setminus S} w_i(D_i) \right) a_{V \setminus S} \,. \qquad (2)$$

The recurrence enables computing a_V using $O(3^n n)$ arithmetic operations [16].

We will apply Theorem 1 to lower the base of the exponential bound:

▶ **Theorem 2.** *The sum over acyclic digraphs with modular weights can be computed using $O(2.985^n)$ arithmetic operations.*

1.2 Related work

There are numerous previous applications of fast matrix multiplication algorithms to decision, optimization, and counting problems. Here we only mention a few that are most related to the present work.

Williams [17] employs fast *square* matrix multiplication to count all variable assignments that satisfy a given number of constraints, each involving at most two variables. By a simple reduction, this yields the fastest known algorithm for the Max-2-CSP problem. The present work is based on the same idea of viewing the product of a group of low-arity functions as a large matrix; this general idea is also studied in the doctoral thesis of the first author [13, Sects. 3.3 and 3.6], including reductions to *RMM*, however, without concrete applications.

Björklund, Kaski, and Kowalik [3] apply fast RMM to show the following: Given a nonnegative integer q and three mappings f, g, h from the subsets of an n-element set to some ring, one can sum up the products $f(A)g(B)h(C)$ over all pairwise disjoint triplets of q-sets A, B, C using $O\big(n^{3q\tau+c}\big)$ ring operations, where $\tau < \frac{1}{2}$ and $c \geq 0$ are constants independent of q and n. Consequently, one can count the occurrences of constant-size paths (or any other small-pathwidth patterns) faster than in the "meet-in-the-middle time" [3]. While the involvement of set functions and set relations bear a resemblance to those in multi-subset transform, the reduction of Björklund et al. is based on solving an appropriately constructed system of linear equations, and is thus very different from the combinatorial approach taken in the present work.

2 Fast multi-subset transform: proof of Theorem 1

We will develop an algorithm for multi-subset transform in several steps. In Section 2.1 we give the basic reduction to RMM and the idea of splitting the sum over into several smaller sums. Then, in Section 2.2 we present a simple implementation of the splitting idea, and get our first below-3 algorithm. This algorithm is improved upon in Section 2.3, yielding the claimed complexity bound. We end this section by presenting a more sophisticated splitting scheme in Section 2.4. We have not succeeded to give a satisfactory analysis of its complexity. Yet, our numerical calculations suggest the bound $O(2.930^n)$.

We will denote by $\omega(k)$, for $k \geq 0$, the smallest value such that the product of an $N \times \lceil N^k \rceil$ matrix by an $\lceil N^k \rceil \times N$ can be computed using $O\big(N^{\omega(k)+\epsilon}\big)$ arithmetic operations for any constant $\epsilon > 0$; for a formal definition of $\omega(k)$, see Gall and Urrutia [7]. Thus, the exponent of square matrix multiplication is $\omega := \omega(1)$.

We will make repeated use of the following facts about binomial coefficients:

▶ **Fact 3.** *For integers $k \geq 1$ and $n \geq 2k$ we have*

$$(2n)^{-1/2} b\Big(\frac{k}{n}\Big)^n \leq \binom{n}{k} \leq \sum_{j=0}^{k} \binom{n}{j} \leq b\Big(\frac{k}{n}\Big)^n = 2^{nH(k/n)},$$

where

$$b(x) := x^{-x}(1-x)^{x-1} \quad \text{and} \quad H(x) := \log_2 b(x), \qquad x \in [0,1].$$

This can be proven using Stirling's approximation to factorials.

▶ **Fact 4.** *Let n be a positive integer. The function $k \mapsto \binom{n}{k}2^k$ is increasing in $[0, \frac{2}{3}n)$ and strictly decreasing in $[\frac{2}{3}n, n)$.*

This can be proven by observing that the ratio $\binom{n}{k+1}2^{k+1}/\binom{n}{k}2^k$ equals $2(n-k)/(k+1)$, and is thus decreasing in k, and is greater or equal to 1 exactly when $k \leq \frac{2}{3}n - \frac{1}{3}$.

2.1 Basic reduction to rectangular matrix multiplication

Assume without loss of generality that n is even. Let us arbitrarily partition U into two disjoint sets U_1 and U_2, both of size $h := n/2$. If $T \subseteq U$, denote by T_1 and T_2 respectively the intersections $T \cap U_1$ and $T \cap U_2$. Furthermore, write $N := 2^h$ so that $2^n = N^2$.

Armed with this notation, we write the multi-subset transform of set functions $(f_i)_{i \in U}$ as

$$g(T) = G(T_1, T_2) := \sum_{S \subseteq U} F_1(T_1, S)\, F_2(T_2, S)\,, \qquad T \subseteq U\,, \tag{3}$$

where we define

$$F_p(T_p, S) := [S \cap U_p \subseteq T_p] \prod_{i \in T_p} f_i(S)\,, \qquad p = 1, 2\,.$$

Here the Iverson's bracket notation $[Q]$ evaluates to 1 if Q is true, and to 0 otherwise.

We can write the representation (3) in terms of a matrix product as

$$G = F_1 F_2^\top\,,$$

where G is an $N \times N$ matrix indexed in $2^{U_1} \times 2^{U_2}$ and F_p is an $N \times N^2$ matrix indexed in $2^{U_p} \times 2^U$. As above, we will write the index pair in parentheses (not as subscripts).

Applying fast RMM without any further tricks already yields a somewhat competitive asymptotic complexity bound. To see this, recall that $\omega(k)$ denotes the exponent of RMM of dimensions $N \times \lceil N^k \rceil \times N$. Since $\omega(2) < 3.252$ [7], we get that G, and thus g, can be computed using $O\big(N^{3.252}\big) = O(3.087^n)$ arithmetic operations. If the lower bound $\omega(2) \geq 3$ was tight, we would achieve the bound $O(2.829^n)$.

So far, we have ignored the sparsity of the matrices F_p. An entry $F_p(T_p, S)$ is zero whenever the intersection $S_p = S \cap U_p$ is not contained in T_p. Thus, out of the $8^{n/2}$ entries of F_p, at most $3^h 2^h = 6^{n/2}$ are nonzero. In general, one can compute a matrix product of dimensions $r \times r^k \times r$ using $O\big(mr^{(\omega-1)/2+\epsilon}\big)$ operations, provided that the matrices have at most $m \geq r^{(\omega+1)/2}$ non-zero entries, irrespective of k [11]. This result applies to our case, but with the best known upper bound for ω [8], it only yields a bound $O(3.108^n)$. A direct reduction to multiple multiplications of sparse square matrices [19] yields an even worse bound, $O(3.142^n)$ (calculations omitted). Output-sensitive sparse matrix multiplication algorithms [1] will not work either, as our output matrix is dense in general.

Luckily, in our case, we can make more efficient use of the sparsity. We will decompose the matrix product into a sum of smaller matrix products, as formulated by the following representation (the proof is trivial and omitted):

▶ **Lemma 5.** *Let $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M\}$ be a set partition of 2^U. Let F_{pq} be the submatrix of F_p obtained by removing all columns but those in $\mathcal{S}_q$, for $p = 1, 2$ and $q = 1, 2, \ldots, M$. Then*

$$G = \sum_{q=1}^{M} G_q\,, \qquad \textit{where } G_q = F_{1q} F_{2q}^\top\,.$$

We will also apply this decomposition after removing some rows from the matrices F_{pq}. Then the index sets may be different for different G_q. To properly define the entry-wise addition in these cases, we simply make the convention that the missing entries equal zero.

To employ a fast RMM algorithm we will call a function Fast-RMM$(\mathcal{T}_1, \mathcal{S}, \mathcal{T}_2)$. The function returns the product $E_1 E_2^\top$, where each E_p is obtained from F_p by only keeping the rows $\mathcal{T}_p$ and the columns $\mathcal{S}$. Note that we do not show the input matrices explicitly in the function call, as the submatrices will always be extracted from F_1 and F_2.

Algorithm 1 The Columns algorithm for the multi-subset transform.

function Columns-Directly($\mathcal{S}$)
$G[T] \leftarrow 0$ for all $T \subseteq U$
for $S \in \mathcal{S}$
  for $T \subseteq U$ s.t. $S \subseteq T$
    $G[T] \leftarrow G[T] + F_1(T_1, S)F_2(T_2, S)$
return G

Algorithm Columns$\big((f_i)_{i\in U}\big)$
$G[T] \leftarrow 0$ for all $T \subseteq U$
select $\sigma \in (\frac{1}{3}, \frac{1}{2})$
$\mathcal{S}_1 \leftarrow \{S \subseteq U : |S| \leq \sigma n\}$
$G \leftarrow G +$ Fast-RMM$\big(2^{U_1}, \mathcal{S}_1, 2^{U_2}\big)$
$G \leftarrow G +$ Columns-Directly$\big(2^U \setminus \mathcal{S}_1\big)$
return G

2.2 A simple below-3 algorithm

We apply Lemma 5 with $M = 2$ and split the columns to those that are smaller than a threshold σn and to those that are at least as large:

$$\mathcal{S}_1 = \{S \subseteq U : |S| < \sigma n\} \quad \text{and} \quad \mathcal{S}_2 = \{S \subseteq U : |S| \geq \sigma n\}\,.$$

We assume σn is an integer and that $\frac{1}{3} < \sigma < \frac{1}{2}$. We will optimize the parameter σ later. The idea is to call fast RMM only for summing over the columns $\mathcal{S}_1$ and to handle the remaining columns in a brute-force manner. The algorithm Column is given in Algorithm 1.

Consider first the computation of the matrix G_1. We compute G_1 using fast RMM. The computational complexity depends on the number of columns in the matrices F_{11} and F_{21}. Letting C be the number of columns, the required number of operations for the matrix multiplication of dimensions $N \times C \times N$ is $O\big(N^{\omega(k)}\big)$, where $k = \log_N C$. We have

$$C = |\mathcal{S}_1| = \sum_{s=0}^{\sigma n} \binom{n}{s} \leq b(\sigma)^n\,, \tag{4}$$

where the inequality follows by Fact 3.

Consider then the computation of the matrix G_2. To compute $G_2(T)$, for $T \subseteq U$, it suffices to compute the sum of the products $F_1(T_1, S)F_2(T_2, S)$ over all columns $S \subseteq T$ whose size is at least σn. Thus, the required number pairs (S, T) to be considered is at most

$$B := \sum_{s=\sigma n}^{n} \binom{n}{s} 2^{n-s} \leq n \binom{n}{\sigma n} 2^{n(1-\sigma)} \leq n\big(2^{1-\sigma} b(\sigma)\big)^n \tag{5}$$

where the penultimate inequality follows by Fact 4 (since $1 - \sigma < \frac{2}{3}$) and the last by Fact 3.

Let us finally combine the bounds in (4) and (5).

▶ **Proposition 6.** *For any $\epsilon > 0$, the number of operations required by* Columns *is*

$$O\left(2^{n(\omega(2H(\sigma))+\epsilon)/2} + n2^{n(1-\sigma+H(\sigma))}\right)\,.$$

It remains to choose σ so as to optimize the bound. Clearly the first term is increasing and the second term is decreasing in σ. Thus, the bound is (asymptotically) minimized by choosing a σ that makes $\omega\big(2H(\sigma)\big)$ equal to $2(1-\sigma+H(\sigma))$. There are two obstacles to implement this idea: first, we only know upper bounds for $\omega(k)$, for various k; second, no closed-form expression is known for the best upper bounds – upper bounds for $\omega(k)$ have been computed and reported only at some points k [7].

Due to these complications, we resort to the following facts:

▶ **Fact 7** ([7]). *The exponent of RMM satisfies* $\omega(1.75) \le 3.021591$.

▶ **Fact 8.** *Let* $k > 0$ *and* $r \ge 0$. *The exponent of RMM satisfies* $\omega(k+r) \le \omega(k) + r$.

(This follows by reducing the larger RMM instance trivially to multiple smaller instances.)

Combining these two facts yields an upper bound:

$$\omega(2H(\sigma)) \le \omega(1.75) + 2H(\sigma) - 1.75 \le 1.271591 + 2H(\sigma)\,.$$

Now, solving $1.271591 + 2H(\sigma) = 2(1-\sigma+H(\sigma))$ gives

$$\sigma = 1 - 1.271591/2 = 0.3642045\,.$$

With this choice of σ the complexity bound becomes $O(2.994^n)$.

2.3 A faster below-3 algorithm

Next we give a slightly faster algorithm to compute G_1. This will allow us to choose a larger threshold σ, thus also rendering the computation of G_2 faster.

Instead of computing G_1 directly using fast RMM, we now compute some rows and columns of G_1 in a brute-force manner and only apply fast RMM to the remaining smaller matrix. Specifically, the algorithm only calls fast RMM to compute the entries $G_1(T_1,T_2)$ where the sizes of T_1 and T_2 exceed τh. We assume that τh is an integer and that $\tau \in (\frac{1}{2}, \frac{2}{3})$. We will optimize the parameter τ together with σ later. The algorithm ROWS&COLUMNS is given in Algorithm 2. The correctness of the algorithm being clear, we proceed to analysing the complexity in terms of the required number of arithmetic operations.

Consider first the computation of an entry $G_1(T_1,T_2)$ where $|T_1| \le \tau h$. The number of pairs (S,T) satisfying $S \subseteq T \subseteq U$ and $|T_1| \le \tau h$ is given by

$$B' := 3^h \sum_{t=0}^{\tau h} \binom{h}{t} 2^t \;\le\; 3^h h \binom{h}{\tau h} 2^{\tau h} \;\le\; h\big(3 \cdot 2^{\tau} b(\tau)\big)^h\,; \tag{6}$$

the penultimate inequality follows by Fact 4 (since $\tau < \frac{2}{3}$) and the last inequality by Fact 3.

Similarly, computing the entries $G_1(T_1,T_2)$ for all $T_1 \subseteq U_1$ and $T_2 \subseteq U_2$ such that $|T_2| \le \tau h$ requires at most B' additions and multiplications.

It remains to compute the entries $G_1(T_1,T_2)$ for $T_1 \subseteq U_1$ and $T_2 \subseteq U_2$ such that $|T_1|, |T_2| > \tau h$. This can be computed as a product of two matrices (submatrices of F_1 and $F_2^{\top}$) whose sizes are at most $R \times C$ and $C \times R$, where C is as before and

$$R := \sum_{j=\tau h+1}^{h} \binom{h}{j} \;\le\; b(\tau)^h\,, \tag{7}$$

where the inequality follows by Fact 3 (since $\tau > \frac{1}{2}$).

Let us combine the bounds in (6) and (7):

Algorithm 2 The ROWS&COLUMNS algorithm for the multi-subset transform.

function ROWS-TRIMMED$(\tau, \mathcal{S})$

$G[T] \leftarrow 0$ for all $T \subseteq U$
$\mathcal{T}_p \leftarrow \{T_p \subseteq U_p : |T_p| > \tau h\}$ for $p \leftarrow 1, 2$
for $S \subseteq T \subseteq U$ s.t. $S \in \mathcal{S}$ and $(T_1 \notin \mathcal{T}_1$ or $T_2 \notin \mathcal{T}_2)$
    $G[T] \leftarrow G[T] + F_1(T_1, S)F_2(T_2, S)$
$G \leftarrow G + \text{FAST-RMM}\big(\mathcal{T}_1, \mathcal{S}, \mathcal{T}_2\big)$
return G

Algorithm ROWS&COLUMNS$\big((f_i)_{i \in U}\big)$

$G[T] \leftarrow 0$ for all $T \subseteq U$
select $\sigma \in (\frac{1}{3}, \frac{1}{2})$ and $\tau \in (\frac{1}{2}, \frac{2}{3})$
$\mathcal{S}_1 \leftarrow \{S \in U : |S| \leq \sigma n\}$
$G \leftarrow G + \text{ROWS-TRIMMED}(\tau, \mathcal{S}_1)$
$G \leftarrow G + \text{COLUMNS-DIRECTLY}\big(2^U \setminus \mathcal{S}_1\big)$
return G

▶ **Proposition 9.** *For any $\epsilon > 0$, the number of operations required by* ROWS&COLUMNS *is*

$$O\Big(n(3 \cdot 2^\tau b(\tau))^{n/2} + b(\tau)^{(\omega(k)+\epsilon)n/2} + n\big(2^{1-\sigma} b(\sigma)\big)^n\Big)\,, \quad \text{where } k = 2\log_{b(\tau)} b(\sigma)\,.$$

To set the parameters σ and τ, we resort to the bound $\omega(k) \leq 1.271591 + k$ (Fact 7 and Fact 8). Balancing the latter two terms in the bound yields the equation

$$(1.271591 + k)H(\tau) = 2\big(1 - \sigma + H(\sigma)\big)\,.$$

Equivalently, $1.271591 \cdot H(\tau) = 2(1 - \sigma)$. Solving for σ and equating the first and the third term in the bound leaves us the equation

$$\log_2 3 + \tau + H(\tau) = 1.271591 \cdot H(\tau) + 2H\big(1 - 0.6357955 \cdot H(\tau)\big)\,.$$

By numerical calculations we find one solution in the valid range, $\tau \approx 0.59777$, and correspondingly $\sigma \approx 0.38185$. With these choices the complexity bound becomes $O(2.985^n)$. This completes the proof of Theorem 1.

2.4 A covering based algorithm

The previous algorithms were based on pruning some columns and rows of the matrices F_1 and F_2, and applying fast RMM to the remaining multiplication of two reduced matrices. Now, we take a different approach and reduce the original problem instance into multiple, smaller RMM instances applying Lemma 5 with some $M > 2$. To this end, we cover – in the sense of a set cover – the columns by multiple groups such that the columns in one group contain a large block of zero entries (in the same set of rows) in the matrices F_1 and F_2.

It will be convenient to consider sets of fixed sizes. For a set V and a nonnegative integer s, write $\binom{V}{s}$ for the set of all s-element subsets of V. Let $s_1, s_2 \in \{0, 1, \ldots, h\}$ fix the sizes of the intersection of a column with the sets U_1 and U_2. We wish to cover the set (of set pairs) $\binom{U_1}{s_1} \times \binom{U_2}{s_2}$ by a small number of sets of the form $\binom{K_1}{s_1} \times \binom{K_2}{s_2}$, where the sets K_1 and K_2 are of some fixed sizes $k_1 \geq s_1$ and $k_2 \geq s_2$. The following classic result [5] shows that this covering design problem has an efficient solution:

▶ **Theorem 10** ([5]). *Let $c(v,k,s)$ be the minimum number of subsets of $\{1,2,\ldots,v\}$ of size k such that every subset of size $s \le k$ is contained by at least one of the sets. We have*

$$c(v,k,s)\binom{k}{s}\binom{v}{s}^{-1} \le 1+\ln\binom{k}{s}.$$

In particular, $c(v,k,s)$ is within the factor k of the obvious lower bound $\binom{v}{s}\binom{k}{s}^{-1}$.

▶ Remark 11. Although the work needed for constructing a covering does not contribute to the number of operations in the ring $\mathcal{R}$, a remark is in order if one is interested in the required number of other operations. The authors are not aware of any deterministic algorithm for constructing an optimal covering in time polynomial in $\binom{v}{k}+\binom{v}{s}$, while asymptotically optimal randomized polynomial-time algorithms are known [9].

Fortunately, for our purposes it suffices to run the well known greedy algorithm that iteratively picks a set that covers the largest number of yet uncovered elements. It finds a set cover whose size is within a logarithmic factor of the optimum, which is sufficient in our context. Furthermore, it can be implemented to run in time linear in the input size [4, Ex. 35.3–3], which is $\binom{v}{k}\binom{k}{s} \le 3^v$ in our case (with $v = h = n/2$).

From now on, we assume that for $p = 1,2$ we are given a set family $\mathcal{K}_p \subseteq \binom{U_p}{k_p}$ that has the desired coverage property, i.e., $\left\{\binom{K_p}{s_p} : K_p \in \mathcal{K}_p\right\}$ is a set cover of $\binom{U_p}{s_p}$, so that for every column $S \subseteq U$ satisfying $|S_1| = s_1$, $|S_2| = s_2$ there is a pair $(K_1,K_2) \in \mathcal{K}_1 \times \mathcal{K}_2$ such that $S_1 \subseteq K_1$, $S_2 \subseteq K_2$. In what follows, we will assume that some appropriate values of k_1, k_2 are chosen based on s_1, s_2; we will return back to the issue of finding good values at the end of this subsection.

For each pair (K_1,K_2), we construct a submatrix E_1 of F_1 as follows: remove from F_1 all columns S not covered by (K_1,K_2), and all rows T_1 whose intersection with K_1 contains less than s_1 elements (as otherwise we cannot have $S_1 \subseteq T_1$ and the entry $F_1(T_1,S)$ vanishes). We construct a matrix E_2 analogously by removing columns and rows from F_2. The dimensions of the matrix product $E_1E_2^\top$ are $R_1 \times C' \times R_2$, where

$$R_1 := \sum_{j=s_1}^{k_1}\binom{k_1}{j}2^{h-k_1}, \quad C' := \binom{k_1}{s_1}\binom{k_2}{s_2}, \quad R_2 := \sum_{j=s_2}^{k_2}\binom{k_2}{j}2^{h-k_2}.$$

Algorithm Cover-Columns, given in Algorithm 3, organizes the reduction to multiple RMM instances like this using Lemma 5. Specifically, from the set cover of the columns it extracts a set partition by trivially keeping track of the already covered columns.

To analyze the complexity of the algorithm, let us first bound the dimensions R_1, C', and R_2 for fixed s_1, s_2, k_1, k_2. We aim at bounds of the form N^α for some $0 < \alpha < 2$, and therefore parameterize the set sizes as

$$s_p = \sigma_p h \quad \text{and} \quad k_p = \kappa_p h, \qquad p = 1,2.$$

Thus $0 \le \sigma_p \le \kappa_p \le 1$. In what follows, we let σ_p/κ_p evaluate to 0 if $\sigma_p = \kappa_p = 0$.

▶ **Lemma 12.** *We have*

$$R_1 \le N^{\beta_1}, \quad C' \le N^{\alpha_1+\alpha_2}, \quad R_2 \le N^{\beta_2},$$

where

$$\alpha_p := \kappa_p H\Big(\frac{\sigma_p}{\kappa_p}\Big) \quad \text{and} \quad \beta_p := 1-\kappa_p+\kappa_p H\Big(\max\Big\{\frac{\sigma_p}{\kappa_p},\frac{1}{2}\Big\}\Big), \qquad p = 1,2. \tag{8}$$

Algorithm 3 The COVER-COLUMNS algorithm for the multi-subset transform.

Algorithm COVER-COLUMNS$\big((f_i)_{i\in U}\big)$

$G[T] \leftarrow 0$ for all $T \subseteq U$
$\mathcal{C} \leftarrow \emptyset$ // Already covered columns
for $(s_1, s_2) \in \{0, 1, \ldots, h\}^2$
  select k_1 and k_2
  $\mathcal{K}_p \leftarrow$ COVERING-DESIGN(s_p, k_p, U_p) for $p \leftarrow 1, 2$
  for $(K_1, K_2) \in \mathcal{K}_1 \times \mathcal{K}_2$
    $\mathcal{S} \leftarrow \{S_1 \cup S_2 : S_1 \in K_1$ and $S_2 \in K_2\}$
    $G \leftarrow G +$ ROWS-TRIMMED$(0, \mathcal{S} \setminus \mathcal{C})$ // Trim only all-zero rows
    $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{S}$
return G

Proof. The bound for C' follows directly from the definitions of σ_p, κ_p, α_p and from Fact 3.

For the bound on R_1 (equivalently R_2), suppose first that $\kappa_1 \geq 2\sigma_1$. Then using the simple inequality $\sum_{j=s_1}^{k_1} \binom{k_1}{j} \leq 2^{k_1} = N^{\kappa_1 H(1/2)}$ gives the claimed bound. Otherwise, $\kappa_1 \leq 2\sigma_1$ and thus, by Fact 3, $\sum_{j=s_1}^{k_1} \binom{k_1}{j} \leq 2^{k_1 H(1-\sigma_1/\kappa_1)} = N^{\kappa_1 H(\sigma_1/\kappa_1)}$, implying the claimed bound. ◀

It remains to turn the bounds on the dimensions to a bound on the complexity of the corresponding RMM and sum up these bounds over the multiple matrix multiplication tasks.

▶ **Proposition 13.** *For any $\epsilon > 0$, the number of operations required by* COVER-COLUMNS *is $O\big(2^{(\gamma+\epsilon)n/2}\big)$, where*

$$\gamma := \max_{\substack{0\leq\sigma_1\leq 1\\ 0\leq\sigma_2\leq 1}} \min_{\substack{\sigma_1\leq\kappa_1\leq 1\\ \sigma_2\leq\kappa_2\leq 1}} H(\sigma_1) + H(\sigma_2) - \alpha_1 - \alpha_2 + \beta_1 + \beta_2 + \beta_*\Big(\omega\Big(\frac{\alpha_1+\alpha_2}{\beta_*}\Big) - 2\Big), \tag{9}$$

with α_p and β_p as defined in (8)*, and $\beta_* := \min\{\beta_1, \beta_2\}$.*

Proof. Let $\epsilon > 0$.

Consider first the complexity of a single matrix multiplication with fixed σ_p, κ_p, for $p = 1, 2$. By Lemma 12 we obtain an upper bound by taking $N^{\max\{\beta_1,\beta_2\}-\beta_*} = N^{\beta_1+\beta_2-2\beta_*}$ matrix multiplications of dimensions $N^{\beta_*} \times N^{\alpha_1+\alpha_2} \times N^{\beta_*}$. This gives us the upper bound $O\big(N^{\beta_1+\beta_2+\beta_*(\omega(k)-2)+\epsilon/2}\big)$, where $k = (\alpha_1+\alpha_2)/\beta_*$. Note that we used only a half of ϵ – we will need the other half for tolerating a nonzero underestimation that is due to minimizing κ_p over reals. We will return to this issue at the end of the proof.

Consider then the number of matrix multiplications for fixed s_p, k_p, for $p = 1, 2$. By Theorem 10 and by the approximation ratio of the greedy algorithm, the number is at most

$$\begin{aligned} n^4 \binom{h}{s_1}\binom{h}{s_2}\binom{k_1}{s_1}^{-1}\binom{k_2}{s_2}^{-1} &\leq n^5 b(\sigma_1)^h b(\sigma_2)^h b(\sigma_1/\kappa_1)^{-\kappa_1 h} b(\sigma_2/\kappa_2)^{-\kappa_2 h} \\ &= n^5 N^{H(\sigma_1)+H(\sigma_2)-\alpha_1-\alpha_2}. \end{aligned}$$

Here we used Fact 3 to bound the binomial coefficients, observing that $(2k_1)^{1/2}(2k_2)^{1/2} \leq n$.

Now, combine the above two bounds, recall that $N = 2^{n/2}$, and observe that replacing the sum over (s_1, s_2) by the maximum over (σ_1, σ_2) is compensated by adding a factor of n^2 to the bound. The algorithm can select optimal k_1 and k_2 by optimizing the upper bound, which costs yet another factor of n^2. Due to the constant ϵ in the exponent, we can ignore the $n^{O(1)}$ factor in the asymptotic complexity bound.

To complete the proof, we show that for any values of σ_p and κ_p (hence also for the optimal values) and for any large enough integer h, there are rational numbers $\kappa'_p \geq \sigma_p$ such that (i) $\kappa'_p h$ are integers and (ii) $\Gamma(\sigma_1, \sigma_2, \kappa'_1, \kappa'_2) \leq \Gamma(\sigma_1, \sigma_2, \kappa_1, \kappa_2) + \epsilon/2$, where

$$\Gamma(\sigma_1, \sigma_2, \kappa_1, \kappa_2) := H(\sigma_1) + H(\sigma_2) + \beta_1 + \beta_2 + \beta_* \Big(\omega\Big(\frac{\alpha_1 + \alpha_2}{\beta_*}\Big) - \Big(\frac{\alpha_1 + \alpha_2}{\beta_*}\Big) - 2\Big) . \quad (10)$$

Note that we rearranged some terms in (9), for a reason that will be revealed in a moment.

We will consider two cases: either σ_1 or σ_2 is near the boundary values 0 or 1, or both are in $[c, 1-c]$, where $c > 0$ is a small constant. We choose $c < \frac{1}{2}$ such that if $0 \leq \sigma_1 < c$ or $1 - c < \sigma_1 \leq 1$, then regardless of σ_2,

$$\Gamma(\sigma_1, \sigma_2, 1, 1) \leq \omega(1) + \epsilon/2 \,,$$

and symmetrically for σ_2. To see that this is possible, observe first that at $\kappa_1 = \kappa_2 = 1$ we have $\alpha_1 = H(\sigma_1)$, $\alpha_2 = H(\sigma_2)$, and thus

$$\begin{aligned} \Gamma(\sigma_1, \sigma_2, 1, 1) &= \beta_1 + \beta_2 + \beta_* \Big(\omega\Big(\frac{\alpha_1 + \alpha_2}{\beta_*}\Big) - 2\Big) \\ &\leq \beta_1 + \beta_2 + \beta_* \Big(\omega\Big(\frac{\alpha_*}{\beta_*}\Big) + \frac{\alpha_1 + \alpha_2 - \alpha_*}{\beta_*} - 2\Big) , \end{aligned}$$

where $\alpha_* := \alpha_p$ if $\beta_* = \beta_p$. Observe that $\alpha_* \leq \beta_*$. Since $\omega(1) - 2 \geq 0$ and $\alpha_1, \alpha_2, \beta_1, \beta_2 \leq 1$,

$$\Gamma(\sigma_1, \sigma_2, 1, 1) \;\leq\; \alpha_1 + \alpha_2 - \alpha_* + \beta_1 + \beta_2 + \omega(1) - 2 \;\leq\; \omega(1) + H(\sigma_1) \,.$$

For the latter inequality we used the facts that $\alpha_* = \alpha_2$ if $\sigma_1 < c$ and that $\beta_1 = H(\sigma_1)$ if $\sigma_1 > 1 - c$. Finally, we observe that $H(\sigma_1)$ tends to 0 when σ_1 tends to 0 or 1.

On the other hand, we have the lower bound $\Gamma(\frac{1}{2}, \frac{1}{2}, \kappa_1, \kappa_2) \geq 2 + \beta_1 + \beta_2 - \beta_* \geq 2.5 > \omega(1)$, since $\omega(z) - z \geq 1$ and $\beta_p = 1 - \kappa_p + \kappa_p H\big(1/(2\kappa_p)\big) \geq \kappa_p \geq \frac{1}{2}$; here we used the fact that $H(x) \geq 2 - 2x$ for $x \in \big[\frac{1}{2}, 1\big]$.

We may thus restrict out attention to the domain

$$\Lambda_c := \big\{(\sigma_1, \sigma_2, \kappa_1, \kappa_2) : c \leq \sigma_1, \sigma_2 \leq 1 - c,\ \sigma_1 \leq \kappa_1 \leq 1,\ \sigma_2 \leq \kappa_2 \leq 1\big\} \,.$$

We now show that Γ is continuous on Λ_c. Observe first that the functions H, α_p, and β_p are continuous on Λ_c (as $\kappa_p > c$). We also have that β_* is continuous and strictly positive (as $\sigma_p \leq 1 - c$) and that $z \mapsto \omega(z)$ is continuous (as $|\omega(z + \delta) - \omega(z)| \leq \delta$ for all $\delta > 0$).

Since the domain Λ_c is compact, we have that Γ is uniformly continuous on Λ_c. This in turn implies that there is a $\delta_\epsilon > 0$ such that (ii) holds whenever $|\kappa'_p - \kappa_p| < \delta_\epsilon$, implying that we can make both (i) and (ii) hold for all $h > 1/\delta_\epsilon$ by putting $\kappa'_p := \lceil \kappa_p h \rceil / h$. ◀

Now we know that the complexity of the algorithm is $O\big(2^{(\gamma+\epsilon)n/2}\big)$, but we do not know how large γ is. Unlike for the simpler algorithms given in the previous subsections, we cannot just select some values of the parameters σ_p and κ_p and bound γ from above by $\Gamma(\sigma_1, \sigma_2, \kappa_1, \kappa_2)$, as defined in (10), for we do not know the maximizing values of σ_p. Since Γ is uniformly continuous on the domain Λ_c, one could in principle prove any fixed strict upper bound on γ with a sufficiently large, finite computation. While at the present time the authors have not produced such a proof, evaluations of $\Gamma(\sigma_1, \sigma_2, \kappa_1, \kappa_2)$ at various values of the four parameters suggest the following:

▶ **Conjecture 14.** *The number of operations required by* Cover-Columns *is* $O(2.930^n)$.

3 Fast weighted counting of acyclic digraphs: proof of Theorem 2

Let us write the inclusion–exclusion recurrence (2) as a multi-subset transform:

▶ **Lemma 15.** *Without loss of generality, suppose $0 \notin V$. Let $0 \in T \subseteq V \cup \{0\}$ and*

$$g(T) = \sum_{S \subseteq T} \prod_{i \in T} f_i(S)\,,$$

where

$$f_i(S) = \begin{cases} 0 & \text{if } 0 \notin S \text{ or } |S| = |T|; \\ (-1)^{|S|-1} a_{S \setminus \{0\}} & \text{else if } i = 0; \\ \sum_{D_i \subseteq S \setminus \{0\}} w_i(D_i) & \text{else if } i \notin S; \\ 1 & \text{otherwise.} \end{cases}$$

Then $a_{T \setminus \{0\}} = (-1)^{|T|} g(T)$.

Proof. Because the summand vanishes unless $0 \in S \neq T$ and because $f_i(S) = 1$ unless $i \in \{0\} \cup (T \setminus S)$, we have

$$\begin{aligned} (-1)^{|T|} g(T) &= (-1)^{|T|} \sum_{0 \in S \subsetneq T} f_0(S) \prod_{i \in T \setminus S} f_i(S) \\ &= \sum_{0 \in S \subsetneq T} (-1)^{|T|+|S|-1} a_{S \setminus \{0\}} \prod_{i \in T \setminus S} \sum_{D_i \subseteq S \setminus \{0\}} w_i(D_i)\,. \end{aligned}$$

Writing in terms of $T' := T \setminus \{0\}$ and $S' := T \setminus S$, and observing that $|S|$ and $-|S|$ have the same parity,

$$(-1)^{|T|} g(T) = \sum_{\emptyset \neq S' \subseteq T'} (-1)^{|S'|-1} a_{T' \setminus S'} \prod_{i \in S'} \sum_{D_i \subseteq T' \setminus S'} w_i(D_i) = a_{T'}\,.$$

The last equality follows immediately from (2). ◀

It remains to organize the computations so that when computing a_T for some $T \subseteq V$, the values a_S have already been computed for all $S \subsetneq T$. To this end, we proceed in increasing order by $|T|$: for each $t = 1, 2, \ldots, n$ in this order we simultanously compute the values a_T for all $T \in \binom{V}{t}$ by calling the fast multi-subset transform, as detailed in algorithm Sum-Acyclic-Digraphs given in Algorithm 4. As we only need n calls, the asymptotic complexity bound (with a rounded constant base of the exponential) remains valid.

References

1. Rasmus Resen Amossen and Rasmus Pagh. Faster join-projects and sparse matrix multiplications. In *12th International Conference on Database Theory*, ICDT '09, pages 121–126. ACM, 2009.
2. Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets Möbius: Fast subset convolution. In *39th ACM Symposium on Theory of Computing*, pages 67–74. ACM, 2007.
3. Andreas Björklund, Petteri Kaski, and Łukasz Kowalik. Counting thin subgraphs via packings faster than meet-in-the-middle time. *ACM Trans. Algorithms*, 13(4):48:1–48:26, 2017.
4. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition.* The MIT Press, 3rd edition, 2009.
5. Paul Erdős and Joel Spencer. *Probabilistic Methods in Combinatorics.* Akadémiai Kiadó, Budapest, 1974.

Algorithm 4 The SUM-ACYCLIC-DIGRAPHS algorithm for the sum over acyclic digraphs with modular weights. FAST-MULTI-SUBSET-TRANSFORM$\big((f_i)_{i \in U}\big)$ returns the multi-subset transform of $(f_i)_{i \in U}$.

Algorithm SUM-ACYCLIC-DIGRAPHS$\big((w_i)_{i \in V}\big)$

$a[\emptyset] \leftarrow 1$; $a[S] \leftarrow 0$ for all $\emptyset \neq S \subseteq V$
compute $f_i[S \cup \{0\}] \leftarrow \sum_{X \subseteq S} w_i(X)$ for all $i \in V$, $S \subseteq V$ using fast zeta transform
$f_i[S] \leftarrow 0$ for all $i \in V$, $S \subseteq V$.
$f_i[S \cup \{0\}] \leftarrow 1$ for all $i \in S \subseteq V$.
for $t \leftarrow 1$ **to** n
    for $S \in \binom{V}{t-1}$
        $f_0[S \cup \{0\}] \leftarrow (-1)^{|S|-1} a[S]$
    $g \leftarrow$ FAST-MULTI-SUBSET-TRANSFORM$\big((f_i)_{i \in V \cup \{0\}}\big)$
    for $T \in \binom{V}{t}$
        $a[T] \leftarrow (-1)^{|T|+1} g[T \cup \{0\}]$
return $a[V]$

---

6 Nir Friedman and Daphne Koller. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Mach. Learn.*, 50(1-2):95–125, 2003.

7 François Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the Coppersmith-Winograd tensor. In *Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018*, pages 1029–1046. SIAM, 2018.

8 François Le Gall. Powers of tensors and fast matrix multiplication. In *International Symposium on Symbolic and Algebraic Computation, ISSAC '14*, pages 296–303. ACM, 2014.

9 Daniel M. Gordon, Oren Patashnik, Greg Kuperberg, and Joel H. Spencer. Asymptotically optimal covering designs. *Journal of Combinatorial Theory, Series A*, 75(2):270–280, 1996.

10 Frank Harary and Edgar M. Palmer. *Graphical Enumeration*, pages 191–194. Academic Press, 1973. Section 8.8 "Acyclic Digraphs".

11 Haim Kaplan, Micha Sharir, and Elad Verbin. Colored intersection searching via sparse rectangular matrix multiplication. In *Twenty-Second Annual Symposium on Computational Geometry*, SCG '06, pages 52–60. ACM, 2006.

12 Robert Kennes. Computational aspects of the Möbius transformation of graphs. *IEEE Transactions on Systems, Man and Cybernetics*, 22(2):201–223, 1992.

13 Mikko Koivisto. *Sum–Product Algorithms for the Analysis of Genetic Risks*. PhD thesis, Department of Computer Science, University of Helsinki, January 2004.

14 Robert W. Robinson. Counting labeled acyclic digraphs. In *New Directions in the Theory of Graphs*, pages 239–273. Academic Press, New York, 1973.

15 Topi Talvitie, Aleksis Vuoksenmaa, and Mikko Koivisto. Exact sampling of directed acyclic graphs from modular distributions. In *Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019*, pages 345–352, 2019.

16 Jin Tian and Ru He. Computing posterior probabilities of structural features in Bayesian networks. In *25th Conference on Uncertainty in Artificial Intelligence*, pages 538–547. AUAI Press, 2009.

17 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

18 Frank Yates. *The Design and Analysis of Factorial Experiments*. Imperial Bureau of Soil Science, 1937.

19 Raphael Yuster and Uri Zwick. Fast sparse matrix multiplication. *ACM Trans. Algorithms*, 1(1):2–13, 2005.