The I/O complexity of Strassenâs matrix multiplication with recomputation

The I/O Complexity of Strassen's Matrix Multiplication with Recomputation*

Gianfranco Bilardi[1] and Lorenzo De Stefani[2]

[1] Department of Information Engineering, University of Padova,
Via Gradenigo 6B/Padova, Italy
bilardi@dei.unipd.it

[2] Department of Computer Science, Brown University,
115 Waterman Street/Providence, United States of America
lorenzo@cs.brown.edu

Abstract. A tight $\Omega((n/\sqrt{M})^{\log_2 7}M)$ lower bound is derived on the I/O complexity of Strassen's algorithm to multiply two $n \times n$ matrices, in a two-level storage hierarchy with M words of fast memory. A proof technique is introduced, which exploits the Grigoriev's flow of the matrix multiplication function as well as some combinatorial properties of the Strassen computational directed acyclic graph (CDAG). Applications to parallel computation are also developed. The result generalizes a similar bound previously obtained under the constraint of no-recomputation, that is, that intermediate results cannot be computed more than once.

1 Introduction

Data movement is increasingly playing a major role in the performance of computing systems, in terms of both time and energy. This technological trend [1] is destined to continue, since the very fundamental physical limitations on minimum device size and on maximum message speed lead to inherent costs when moving data, whether across the levels of a hierarchical memory system or between processing elements of a parallel system [2]. The communication requirements of algorithms have been the target of considerable research in the last four decades; however, obtaining significant lower bounds based on such requirements remains an important and challenging task.

In this paper, we focus on the I/O complexity of Strassen's matrix multiplication algorithm. Matrix multiplication is a pervasive primitive utilized in many applications. Strassen [3] showed that two $n \times n$ matrices can be multiplied with $O(n^{\omega})$ operations, where $\omega = \log_2 7 \approx 2.8074$, hence with asymptotically fewer than the n^3 arithmetic operations required by the straightforward implementation of the definition of matrix multiplication. This result has motivated a number of efforts which have lead to increasingly faster algorithms, at least asymptotically, with the current record being at $\omega < 2.3728639$ [4].

* This work was supported, in part, by MIUR of Italy under project AMANDA 2012C4E3KT 004 and by the University of Padova under projects CPDA121378/12, and CPDA152255/15.

Previous and Related Work: I/O complexity has been introduced in the seminal work by Hong and Kung [5]; it is essentially the number of data transfers between the two levels of a memory hierarchy with a fast memory of M words and a slow memory with an unbounded number of words. Hong and Kung presented techniques to develop lower bounds to the I/O complexity of computations modeled by *computational directed acyclic graphs* (CDAGs). The resulting lower bounds apply to all the schedules of the given CDAG, including those with recomputation, that is, where some vertices of the CDAG are evaluated multiple times. Among other results, they established an $\Omega\left(n^3/\sqrt{M}\right)$ lower bound to the I/O complexity of the definition-based matrix multiplication algorithm, which matched a known upper bound [6]. The techniques of [5] have also been extended to obtain tight communication bounds for the definition-based matrix multiplication in some parallel settings [7–9] and for the special case of "*sparse matrix multiplication*" [10]. Ballard et al. generalized the results on matrix multiplication of Hong and Kung [5] in [11, 12] by using the approach proposed in [8] based on the Loomis-Whitney geometric theorem [13, 14]. The same papers present tight I/O complexity bounds for various classical linear algebra algorithms, for problems such as LU/Cholesky/LDLT/QR factorization and eigenvalues and singular values computation.

It is natural to wonder what is the impact of Strassen's reduction of the number of arithmetic operations on the number of data transfers. In an important contribution, Ballard et al. [15], obtained an $\Omega((n/\sqrt{M})^{\log_2 7}M)$ I/O lower bound for Strassen's algorithm, using the "*edge expansion approach*". The authors extend their technique to a class of "*Strassen-like*" fast multiplication algorithms and to fast recursive multiplication algorithms for rectangular matrices [16]. This result was later generalized to a broader class of "*Strassen-like*" algorithms by Scott et. al [17] using the "*path routing*" technique. In [18] (Chap. 4.5), De Stefani presented an alternative technique for obtaining I/O lower bounds for a large class of Strassen-like algorithms characterized by a recursive structure. This result combines the concept of Grigoriev's flow of a function and the "*dichotomy width*" technique [19]; it generalizes previous results and simplifies the analysis.

A parallel, "*communication avoiding*" implementation of Strassen's algorithm whose performance matches the known lower bound [15, 17], was proposed by Ballard et al. [20]. A communication efficient algorithm for the special case of sparse matrices based on Strassen's algorithm was presented in [21].

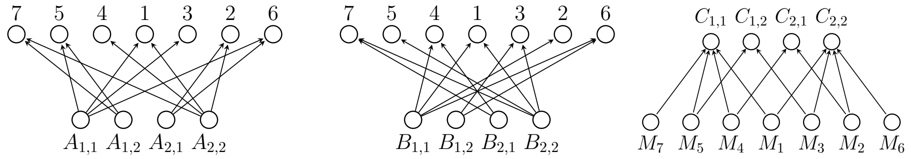
On the impact of recomputation: The edge expansion technique of [15], the path routing technique of [17], and the "*closed dichotomy width*" technique of [19] all yield I/O lower bounds that apply only to computational schedules for which no intermediate result is ever computed more than once (*nr-computations*). While it is of interest to know what is the I/O complexity achievable by nr-computations, it is also important to investigate what can be achieved with recomputation. In fact, for some CDAGs, recomputing intermediate values reduces the space and/or the I/O complexity of an algorithm [22]. In [23], it is shown that some algorithms admit a *portable schedule* (i.e., a schedule which achieves optimal performance across memory hierarchies with different access costs) only if recomputation is allowed. Recomputation can also enhance the performance of simulations among networks (see [24] and references therein) and plays a key role in the design of efficient area-universal VLSI architectures with constant slowdown [25]. A number of lower bound techniques that allow for recomputation have been presented in the literature, including the "*S-partition* technique" [5], the "*S-span* technique" [22], and the "*S-covering* technique" [26] which merges and extends aspects from both [5] and [22]. However, none of these have been previously applied to fast matrix multiplication algorithms.

Our results: We extend the $\Omega((n/\sqrt{M})^{\log_2 7}M)$ I/O complexity lower bound for Strassen's algorithm to schedules with recomputation. A matching upper bound is known, and obtained without recomputation; hence, we can conclude that, for Strassen's algorithm, recomputation does not help in reducing I/O complexity if not, possibly, by a constant factor. Our proof technique is of independent interest, since it exploits to a significant extent the "*divide and conquer*" nature exhibited by many algorithms. We follow the dominator set approach pioneered by Hong and Kung in [5]. However, we focus the dominator analysis only on a select set of target vertices, specifically the outputs of the sub-CDAGs of Strassen's CDAG that correspond to sub-problems of a suitable size (i.e., chosen as a function of the fast memory capacity M). Any dominator set of a set of target vertices can be partitioned into two subsets, one internal and one external to the sub-CDAGs. The analysis of the internal component can be carried out based only on the fact that the sub-CDAGs compute matrix products, irrespective of the algorithm (in our case, Strassen's) by which the products are computed. To achieve this independence of the algorithm, we resort on the concept of Grigoriev's flow of a function [27] and on a lower bound to such flow established by Savage [28] for matrix multiplication..

In order to obtain our *general* lower bound for the I/O complexity, we then build on this result combining it with the analysis of the external component of the dominator, which requires instead rather elaborate arguments that are specific to Strassen's CDAG. The paper is organized as follows: In the first part of Sect. 2, we provide the details of our model and of several theoretical notions needed in our analysis. In the second part of Sect. 2, we analyze the relation between the Grigoriev's flow of a function and the size of the dominator sets of subsets of output vertices of a CDAG. In Sect. 3, we present the I/O complexity lower bound for Strassen's algorithm when recomputation is allowed. Extensions of the result to a parallel model are also discussed.

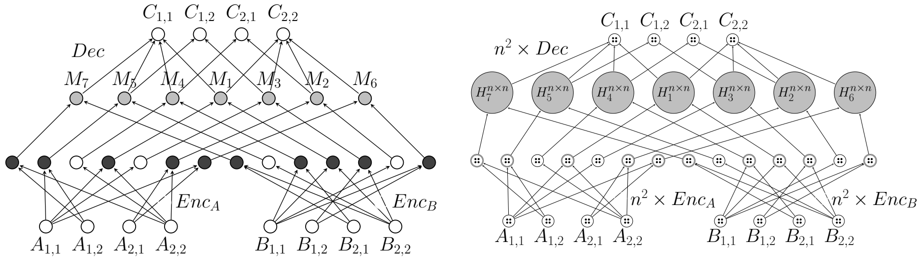
2 I/O Complexity, Dominator Sets and Grigoriev's Flow

We consider algorithms which compute the product $C = AB$ of $n \times n$ matrices A, B with entries from a ring $\mathcal{R}$. We focus on algorithms whose execution, for any given n, can be modeled as a *computational directed acyclic graph* (CDAG) $G = (V, E)$, where each vertex $v \in V$ represents either an input value or the result of a unit time operation (i.e., an intermediate result or one of the output values), while the directed edges in E represent data dependences. A *directed*


7 5 4 1 3 2 6
$A_{1,1}$ $A_{1,2}$ $A_{2,1}$ $A_{2,2}$
7 5 4 1 3 2 6
$B_{1,1}$ $B_{1,2}$ $B_{2,1}$ $B_{2,2}$
$C_{1,1}$ $C_{1,2}$ $C_{2,1}$ $C_{2,2}$
M_7 M_5 M_4 M_1 M_3 M_2 M_6

(a) Enc_A (b) Enc_B (c) Dec

Fig. 1: Basic building blocks of Strassen's CDAG. Enc_A and Enc_B are isomorphic.


$C_{1,1}$ $C_{1,2}$ $C_{2,1}$ $C_{2,2}$
Dec
M_7 M_5 M_4 M_1 M_3 M_2 M_6
Enc_A
Enc_B
$A_{1,1}$ $A_{1,2}$ $A_{2,1}$ $A_{2,2}$ $B_{1,1}$ $B_{1,2}$ $B_{2,1}$ $B_{2,2}$
$C_{1,1}$ $C_{1,2}$ $C_{2,1}$ $C_{2,2}$
$n^2 \times Dec$
$H_7^{n\times n}$ $H_5^{n\times n}$ $H_4^{n\times n}$ $H_1^{n\times n}$ $H_3^{n\times n}$ $H_2^{n\times n}$ $H_6^{n\times n}$
$n^2 \times Enc_A$
$n^2 \times Enc_B$
$A_{1,1}$ $A_{1,2}$ $A_{2,1}$ $A_{2,2}$ $B_{1,1}$ $B_{1,2}$ $B_{2,1}$ $B_{2,2}$

(a) Strassen's $H^{2\times 2}$ CDAG (b) Recursive construction of $H^{2n\times 2n}$

Fig. 2: Black vertices represent combinations of the input values from the factor matrices A and B which are used as input values for the sub-problems M_i; Grey vertices represent the output of the seven sub-problems which are used to compute the output values of the product matrix C.

path connecting vertices $u, v \in V$ is an ordered sequence of vertices for which u and v are respectively the first and last vertex such that there is in E a (directed) edge pointing from each vertex in the sequence to its successor. We say that $G' = (V', E')$ is a *sub-CDAG* of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E \cap (V' \times V')$.

Properties of Strassen's CDAG: Consider Strassen's algorithm [3] when used to compute $C = AB$, where A and B are $n \times n$ matrices with entries from the ring $\mathcal{R}$. Let $H^{n\times n}$ denote the corresponding CDAG. For $n \geq 2$, $H^{n\times n}$ can be obtained by using a recursive construction which mirrors the recursive structure of the algorithm. The base of the construction is the $H^{2\times 2}$ CDAG which corresponds to the multiplication of two 2×2 matrices using Strassen's algorithm (Fig. 2a). $H^{2n\times 2n}$ can then be constructed by composing seven copies of $H^{n\times n}$, each corresponding to one of the seven sub-products generated by the algorithm (see Fig. 2b): n^2 disjoint copies of CDAG Enc_A (resp., Enc_B) are used to connect the input vertices of $H^{2n\times 2n}$, which correspond to the values of the input matrix A (resp., B) to the appropriate input vertices of the seven sub-CDAGs $H_i^{n\times n}$; the output vertices of the sub-CDAGs $H_i^{n\times n}$ (which correspond to the outputs of the seven sub-products) are connected to the appropriate output vertices of the entire $H^{2n\times 2n}$ CDAG using n^2 copies of the decoder sub-CDAG Dec.

We will exploit the following recursive structure of Strassen's CDAG:

Lemma 1. *Let $H^{n\times n}$ denote the CDAG of Strassen's algorithm for input matrices of size $n \times n$. For $0 \leq i \leq \log n - 1$, there are exactly 7^i disjoint sub-CDAGs $H^{n/2^i \times n/2^i}$.*

We will also capitalize on the existence of vertex-disjoint paths connecting the "*global*" input vertices of $H^{n\times n}$ to the "*local*" input vertices of the sub-CDAGs $H^{n/2^i \times n/2^i}$ for $0 \leq i \leq \log n - 1$, with the help of the following lemma.

Lemma 2. *Given an encoder CDAG, for any subset Y of its output vertices, there exists a subset X of its input vertices, with $\min\{|Y|, 1 + \lceil (|Y| - 1)/2 \rceil\} \leq |X| \leq |Y|$, such that there exist $|X|$ vertex-disjoint paths connecting the vertices in X to vertices in Y.*

We refer the reader to the extended on-line version of this paper [29] for a detailed presentation of Strassen's algorithm and for the proofs of Lemmas 1 and 2.

Model: We assume that sequential computations are executed on a system with a two-level memory hierarchy consisting of a fast memory or *cache* of size M, measured in words, and a *slow memory* of unlimited size. A memory word can store at most one value from $\mathcal{R}$. An operation can be executed only if all its operands are in cache. Data can be moved from the slow memory to the cache by *read* operations, and, in the other direction, by *write* operations. Read and write operations are also called *I/O operations*. We assume the input data to be stored in slow memory at the beginning of the computation. The evaluation of a CDAG in this model can be analyzed by means of the "*red-blue pebble game*" [5]. The number of I/O operations executed when evaluating a CDAG depends on the "*computational schedule*," that is, on the order in which vertices are evaluated and on which values are kept in/discarded from cache. The *I/O complexity* $IO_G(M)$ of a CDAG G is defined as the minimum number of I/O operations over all possible computational schedules.

We also consider a parallel model where P processors, each with a local memory of size M, are connected by a network. We assume that the input is initially distributed among the processors, thus requiring that $MP \geq 2n^2$. Processors can exchange point-to-point messages among each other. For this model, we derive lower bounds to the number of words that must be either sent or received by at least one processor during the CDAG evaluation.

Grigoriev's flow and dominator sets: The concept of *dominator set* was originally introduced in [5]. We use the following, slightly different, definition:

Definition 1 (Dominator set). *Given a CDAG $G = (V, E)$, let $I \subset V$ denote the set of input vertices. A set $D \subseteq V$ is a* dominator set *for $V' \subseteq V$ with respect to $I' \subseteq I$ if every path from a vertex in I' to a vertex in V' contains at least a vertex of D. When $I' = I$, D is simply referred as "a dominator set for $V' \subseteq V$".*

The "*flow of a function*" was introduced by Grigoriev [27]. We use a revised formulation by Savage [28]. The flow is an inherent property of a function, not of a specific algorithm by which the function may be computed.

Definition 2 (Grigoriev's flow). *A function $f : \mathcal{R}^p \to \mathcal{R}^q$ has a $w(u, v)$ Grigoriev's flow if for all subsets X_1 and Y_1, of its p input and q output variables,* *with $|X_1| \geq u$ and $|Y_1| \geq v$, there is a sub-function h of f obtained by making some assignment to variables of f not in X_1 and discarding output variables not in Y_1, such that h has at least $|\mathcal{R}|^{w(u,v)}$ points in the image of its domain.*

A lower bound on the Grigoriev's flow for the square matrix multiplication function $f_{n\times n} : \mathcal{R}^{2n^2} \rightarrow \mathcal{R}^{n^2}$ over the ring $\mathcal{R}$ was presented in [28] (Thm. 10.5.1).

Lemma 3 (Grigoriev's flow of $f_{n\times n} : \mathcal{R}^{2n^2} \rightarrow \mathcal{R}^{n^2}$ [28]). *$f_{n\times n} : \mathcal{R}^{2n^2} \rightarrow \mathcal{R}^{n^2}$ has a $w_{n\times n}(u,v)$ Grigoriev's flow, where:*

$$w_{n\times n}(u,v) \geq \frac{1}{2}\left(v - \frac{\left(2n^2 - u\right)^2}{4n^2}\right), \text{for } 0 \leq u \leq 2n^2,\ 0 \leq v \leq n^2. \quad (1)$$

The "*flow of a function*" measures the amount of information that suitable subsets of outputs encode about suitable subsets of inputs. Such information must be encoded by any dominator of those outputs, thus implying the following lower bound on the size of dominators.

Lemma 4. *Let $G = (V,E)$ be a CDAG computing $f : \mathcal{R}^p \rightarrow \mathcal{R}^q$ with Grigoriev's flow $w_f(u,v)$. Let I (resp., O) denote the set of input (resp., output) vertices of G. Any dominator set D for any subset $O' \subseteq O$ with respect to any subset $I' \subseteq I$ satisfies $|D| \geq w_f(|I'|,|O'|)$.*

Proof. Given $I' \subseteq I$ and $O' \subseteq O$, suppose the values of the input variables in $I \setminus I'$ to be fixed. Let D be a dominator set for $O' \subseteq O$ with respect to $I' \subseteq I$. The lemma follows combining statements (i) and (ii):
(i) By Definition 2, there exists an assignment of the input variables in I', such that the output variables in O' can assume $|\mathcal{R}|^{w_f\left(|I'|,|O'|\right)}$ distinct values.
(ii) Since all paths I' to O' intercept D, the values of the outputs in O' are determined by the inputs in $I \setminus I'$, which are fixed, and by the values of the vertices in D; hence, the outputs in O' can take at most $|\mathcal{R}|^{|D|}$ distinct values. □

We let $G^{n\times n}$ denote the CDAG corresponding to the execution of an *unspecified* algorithm for the square matrix multiplication function.

Lemma 5. *Given $G^{n\times n}$, let $O' \subseteq O$ be a subset of its output vertices O. For any subset D of the vertices of $G^{n\times n}$ with $|O'| \geq 2|D|$, there exists a set $I' \subseteq I$ of the input vertices I with cardinality $|I'| \geq 2n\sqrt{|O'| - 2|D|}$, such that all vertices in I' are connected to some vertex in O' by directed paths with no vertex in D.*

Proof. Lemma 5 follows by applying the results in Lemmas 3 and 4 to the CDAG $G^{n\times n}$. Let $I'' \subseteq I$ denote the set of all input vertices of $G^{n\times n}$, such that all paths connecting these vertices to the output vertices in O' include at least a vertex in D (i.e., I'' is the largest subset of I with respect to whom D is a dominator set for O'). From Lemmas 3 and 4 the following must hold:

$$|D| \geq w_{n\times n} \geq \frac{1}{2}\left(|O'| - \frac{\left(2n^2 - |I''|\right)^2}{4n^2}\right). \quad (2)$$

Let $I' = I \setminus I''$. By the definition of I'', the vertices in I' are exactly those that are connected to vertices in O' by directed paths with no vertex in D. Since $|I| = 2n^2$, from (2) we have $|I'|^2 \geq 4n^2\left(|O'| - 2|D|\right)$. □

3 Lower Bounds for Schedules with Recomputation

Without recomputation, once an input value is loaded in memory or an intermediate result is computed, it must be kept in memory (either cache or slow) until the result of each operation which uses it has been evaluated. This is exploited by the "*dichotomy width* technique" [19], the "*boundary flow* technique" [30], and those yielding I/O lower bounds for Strassen's algorithm [16–18]. With recomputation, intermediate results can instead be deleted from *all* memory and recomputed starting from the global input values. This considerably complicates the analysis of the I/O cost (see [11] for an extensive discussion). In this section, we present a technique which addresses these complications. First, we obtain a lower bound for the minimum size of the dominator set of subset of vertices corresponding to the output values of the $(n/(2\sqrt{M}))^{\log_2 7}$ Strassen's sub-problems with input size $2\sqrt{M} \times 2\sqrt{M}$. In turn, this dominator bound yields an asymptotically tight I/O lower bound both in the sequential and the parallel model.

For $1 \leq M \leq n^2/4$, with M a power of four, we focus on the subset $\mathcal{Y}$ of the input vertices and the subset $\mathcal{Z}$ of the output vertices of the $(n/(2\sqrt{M}))^{\log_2 7}$ sub-CDAGs $H^{2\sqrt{M}\times 2\sqrt{M}}$ of $H^{n\times n}$. Further, we let $\mathcal{X}$ be the set of the "*global input vertices*" of $H^{n\times n}$ which correspond to the entries of matrices A and B.

Lemma 6. *Given $H^{n\times n}$, let Q be a set of* internal *(i.e., not input) vertices of its $\left(n/(2\sqrt{M})\right)^{\log_2 7}$ sub-CDAGs $H^{2\sqrt{M}\times 2\sqrt{M}}$. For any $Z \subseteq \mathcal{Z}$ with $|Z| \geq 2|Q|$ there exist $X \subseteq \mathcal{X}$ and $Y \subseteq \mathcal{Y}$ with $|X| = |Y| \geq 4\sqrt{M\left(|Z| - 2|Q|\right)}$ such that, (a) there are $|X| = |Y|$ vertex-disjoint paths from X to Y, and (b) each vertex in Y is connected to some vertex in Z by a directed path with no vertex in Q.*

Proof. For a fixed M, we proceed by induction on $n = 2\sqrt{M}, 4\sqrt{M}, \ldots$ In the base case, $H^{n\times n} = H^{2\sqrt{M}\times 2\sqrt{M}}$, and the sets $\mathcal{Y}$ and $\mathcal{X}$ coincide. The statement is a consequence of Lemma 5 as $H^{2\sqrt{M}\times 2\sqrt{M}}$ is a $G^{2\sqrt{M}\times 2\sqrt{M}}$ CDAG.

Assuming now inductively that the statement holds for $H^{n\times n}$, with $n \geq 2\sqrt{M}$, we shall show it also holds for $H^{2n\times 2n}$. Let $H_1^{n\times n}, H_2^{n\times n}, \ldots, H_7^{n\times n}$ denote the seven sub-CDAGs of $H^{2n\times 2n}$, each corresponding to one of the seven sub-products generated by the first recursive step of Strassen's algorithm.

Let Z_i, $\mathcal{Y}_i$ and Q_i respectively denote the subsets of Z, $\mathcal{Y}$ and Q in $H_i^{n\times n}$. Since, from Lemma 1, the seven sub-CDAGs $H_i^{n\times n}$ are mutually vertex-disjoint, clearly $Z_1, Z_2, \ldots, Z_7$ partition Z, $\mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_7$ partition $\mathcal{Y}$ and $Q_1, Q_2, \ldots, Q_7$ partition Q. This implies $\sum_{i=1}^{7} |Z_i| = |Z|$, and $\sum_{i=1}^{7} |Q_i| = |Q|$. Letting $\delta_i = \max\{0, |Z_i| - 2|Q_i|\}$, we have $\delta = \sum_{i=1}^{7} \delta_i \geq |Z| - 2|Q|$.

Applying the inductive hypothesis to each $H_i^{n\times n}$, we have that there is a subset $Y_i \subseteq \mathcal{Y}_i$ with $|Y_i| \geq 4\sqrt{M\delta_i}$ such that vertices of Y_i are connected to vertices in Z_i via paths with no vertex in Q_i. In the sequel, the set Y referred to in the statement will be identified as a suitable subset of $\cup_{i=1}^{7} Y_i$ so that property (b) will be automatically satisfied. Towards property (a), we observe by the inductive hypothesis that vertices in Y_i can be connected to a subset K_i of the input vertices of $H_i^{n\times n}$ with $|K_i| = |Y_i|$ using vertex-disjoint paths. Since the sub-CDAGs $H_i^{n\times n}$ are vertex-disjoint, so are the paths connecting vertices in Y_i to vertices in K_i. It remains to show that at least $4\sqrt{M\left(|Z| - 2|Q|\right)}$ of these paths can be extended to $\mathcal{X}$ while maintaining them vertex-disjoint.

In Strassen's CDAG $H^{2n\times 2n}$ (Sect. 2), vertices in $\mathcal{X}$ corresponding to input matrix A (resp., B) are connected to vertices in $K_1, K_2, \ldots, K_7$ by means of n^2 encoding sub-CDAGs Enc_A (resp., Enc_B). None of these $2n^2$ encoding sub-CDAGs share any input or output vertices. No two output vertices of the same encoder sub-CDAG belong to the same sub-CDAG $H_i^{n\times n}$. This fact ensures that for a single sub-CDAG $H_i^{n\times n}$ it is possible to connect all the vertices in K_i to a subset of the vertices in $\mathcal{X}$ via vertex-disjoint paths.

For each of the $2n^2$ encoder sub-CDAGs, let us consider the vector $\mathbf{y}_j \in \{0,1\}^7$ such that $\mathbf{y}_j[i] = 1$ iff the corresponding i-th output vertex (respectively according to the numbering indicated in Fig. 1a or Fig. 1b) is in K_i. Therefore, $|\mathbf{y}_j|$ equals the number of output vertices of the j-th encoder sub-CDAG which are in K. From Lemma 2, for each encoder sub-CDAG there exists a subset $X_j \in \mathcal{X}$ of the input vertices of the j-th encoder sub-CDAG for which it is possible to connect each vertex in X_j to a distinct output vertex of the j-th encoder sub-CDAG using vertex-disjoint paths, each constituted by a singular edge with $\min\{|\mathbf{y}_j|, 1 + \lceil (|\mathbf{y}_j| - 1)/2 \rceil\} \leq |X_j| \leq |\mathbf{y}_j|$. Therefore, the number of vertex-disjoint paths connecting vertices in $\mathcal{X}$ to vertices in $\cup_{i=1}^{7} K_i$ is at least $\sum_{j=1}^{2n^2} \min\{|\mathbf{y}_j|, 1 + \lceil (|\mathbf{y}_j| - 1)/2 \rceil\}$ under the constraint that $\sum_{j=1}^{2n^2} \mathbf{y}_j[i] = 4\sqrt{M\delta_i}$. Let us assume, w.l.o.g., that $\delta_1 \geq \delta_2 \geq \ldots \geq \delta_7$. As previously stated, it is possible to connect all vertices in K_1 to vertices in $\mathcal{X}$ through vertex-disjoint paths. Consider now all possible dispositions of the vertices in $\cup_{i=2}^{7} K_i$ over the outputs of the $2n^2$ encoder sub-CDAGs. Recall that the output vertices of an encoder sub-CDAG belong each to a different $H^{n\times n}$ sub-CDAG. From Lemma 2, we have that for each encoder, there exists a subset $X_j \subset X$ of the input vertices of the j-th encoder sub-CDAG with $|X_j| \geq \min\left\{|\mathbf{y}_j|, 1 + \lceil (|\mathbf{y}_j| - 1)/2 \rceil \right\} \geq \mathbf{y}_j[1] + \left(\sum_{i=2}^{7} \mathbf{y}_j[i]\right)/2$, for which it is possible to connect all vertices in X_j to $|X_j|$ *distinct* output vertices of the j-th encoder sub-CDAG which are in $\cup_{i=1}^{7} K_i$ using $|X_j|$, thus using vertex-disjoint paths. As all the *Enc* sub-CDAGs are vertex-disjoint, we can add their contributions so that the number of vertex-disjoint paths connecting vertices in $\mathcal{X}$ to vertices in $\cup_{i=1}^{7} K_i$ is at least $|K_1| + \frac{1}{2}\sum_{i=2}^{7} |K_i| = 4\sqrt{M}\left(\sqrt{\delta_1} + \frac{1}{2}\sum_{i=2}^{7}\sqrt{\delta_i}\right)$. Squaring this quantity leads to:

$$\left(4\sqrt{M}\left(\sqrt{\delta_1} + \frac{1}{2}\sum_{i=2}^{7}\sqrt{\delta_i}\right)\right)^2 = 16M\left(\delta_1 + \sqrt{\delta_1}\sum_{i=2}^{7}\sqrt{\delta_i} + \left(\frac{1}{2}\sum_{i=2}^{7}\sqrt{\delta_i}\right)^2\right),$$

since, by assumption, $\delta_1 \geq \ldots \delta_7$, we have: $\sqrt{\delta_1}\sqrt{\delta_i} \geq \delta_i$ for $i = 2, \ldots, 7$. Thus:

$$\left(4\sqrt{M}\left(\sqrt{\delta_1} + \frac{1}{2}\sum_{i=2}^{7}\sqrt{\delta_i}\right)\right)^2 \geq 16M\sum_{i=1}^{7}\delta_i \geq \left(4\sqrt{M\left(|Z| - 2|Q|\right)}\right)^2.$$

Thus,there are at least $4\sqrt{M\left(|Z| - 2|Q|\right)}$ vertex-disjoint paths connecting vertices in $\mathcal{X}$ to vertices in $\cup_{i=2}^{7} K_i$ as desired. □

Lemma 7. *For $1 \leq M \leq n^2/4$, and for any subset $Z \subseteq \mathcal{Z}$ in $H^{n\times n}$ with $|Z| = 4M$, any dominator set D of Z satisfies $|D| \geq |Z|/2 = 2M$.*

Proof. Suppose for contradiction that D is a dominator set for Z in $H^{n\times n}$ such that $|D| \leq 2M - 1$. Let $D' \subseteq D$ be the subset of the vertices of D composed by vertices which are *not* internal to the sub-CDAGs $H^{2\sqrt{M}\times 2\sqrt{M}}$. From Lemma 6, with $Q = D \setminus D'$, there exist $X \subseteq \mathcal{X}$ and $Y \subseteq \mathcal{Y}$ with $|X| = |Y| \geq 4\sqrt{M\left(|Z| - 2\left(|D| - |D'|\right)\right)}$ such that vertices in X are connected to vertices in Y by vertex-disjoint paths. Hence, each vertex in D' can be on at most one of these paths. Thus, there exists $X' \subseteq X$ and $Y' \subseteq Y$ with $|X'| = |Y'| \geq \nu = 4\sqrt{M\left(|Z| - 2\left(|D| - |D'|\right)\right)} - |D'|$ paths from X' to Y' with no vertex in D'. From Lemma 6, we also have that all vertices in Y, and, hence, in Y', are connected to some vertex in Z by a path with no vertex in $D \setminus D'$. Thus, there are at least ν paths connecting vertices in $X' \subseteq \mathcal{X}$ to vertices in Z with no vertex in D. We shall now show that the contradiction assumption $|D| \leq 2M - 1$ implies $\nu > 0$:

$$\begin{aligned}\left(4\sqrt{M\left(|Z| - 2\left(|D| - |D'|\right)\right)}\right)^2 &= 16M\left(|Z| - 2\left(|D| - |D'|\right)\right), \\ &= 16M\left(|Z| - 2|D|\right) + 32M|D'|.\end{aligned}$$

By $|D| \leq 2M - 1$, we have $|Z| - 2|D| > 4M - 2(M-1) > 0$. Furthermore, from $D' \subseteq D$, we have $32M > 2M - 1 > |D| \geq |D'|$. Therefore:

$$\left(\nu + |D'|\right)^2 = \left(4\sqrt{M\left(|Z| - 2\left(|D| - |D'|\right)\right)}\right)^2 > |D'|^2. \tag{3}$$

Again, $|D| \leq 2M - 1$ implies $M\left(|Z| - 2\left(|D| - |D'|\right)\right) > 0$. Hence, we can take the square root on both sides of (3) and conclude that $\nu > 0$. Therefore, for $|D| \leq 2M - 1$ there are at least $\nu > 0$ paths connecting a global input vertex to a vertex in Z with no vertex in D, contradicting the assumption that D is a dominator of Z. □

Lemma 7 provides us with the tools required to obtain our main result.

Theorem 1 (Lower bound I/O complexity Strassen's algorithm). *The I/O-complexity of Strassen's algorithm to multiply two matrices $A, B \in \mathcal{R}^{n\times n}$, on a sequential machine with cache of size $M \leq n^2$, satisfies:*

$$IO_{H^{n\times n}}\left(M\right) \geq \frac{1}{7}\left(\frac{n}{\sqrt{M}}\right)^{\log_2 7} M. \tag{4}$$

On P processors, each with a local memory of size $M \leq n^2$, the I/O complexity satisfies:

$$IO_{H^{n\times n}}(P,M) \geq \frac{1}{7}\left(\frac{n}{\sqrt{M}}\right)^{\log_2 7}\frac{M}{P}. \tag{5}$$

Proof. We start by proving (4). Let $n = 2^a$ and $\sqrt{M} = 2^b$ for some $a, b \in \mathbb{N}$. At least $3n^2 \geq 3M$ I/O operations are necessary in order to read the $2n^2$ input values from slow memory to the cache and to write the n^2 output values to the slow memory. The bound in (4) is therefore verified if $n \leq 2\sqrt{M}$.

For $n \geq 4\sqrt{M}$, let $\mathcal{Z}$ denote the set of output vertices of the $\left(n/(2\sqrt{M})\right)^{\log_2 7}$ sub-CDAGs $H^{2\sqrt{M}\times 2\sqrt{M}}$ of $H^{n\times n}$. Let $\mathcal{C}$ be any computation schedule for the sequential execution of Strassen's algorithm using a cache of size M. We partition $\mathcal{C}$ into segments $\mathcal{C}_1, \mathcal{C}_2, \ldots$ such that during each $\mathcal{C}_i$ exactly $4M$ distinct vertices in $\mathcal{Z}$ (denoted as Z_i) are evaluated for the *first time*. Since $|\mathcal{Z}| = 4M\left(n/(2\sqrt{M})\right)^{\log 7}$, there are $\left(n/(2\sqrt{M})\right)^{\log 7}$ such segments. Below we show that the number q_i of I/O operations executed during each $\mathcal{C}_i$ satisfies $q_i \geq M$, from which (4) follows.

To bound q_i, consider the set D_i of vertices of $H^{n\times n}$ corresponding to the at most M values stored in the cache at the beginning of $\mathcal{C}_i$ and to the at most q_i values loaded into the cache from the slow memory during $\mathcal{C}_i$ by means of a *read* I/O operation. Clearly, $|D_i| \leq M + q_i$. In order for the $4M$ values from Z_i to be computed during $\mathcal{C}_i$ there cannot be any path connecting any vertex in Z_i to any input vertex of $H^{n\times n}$ which does not have at least one vertex in D_i; that is, D_i has to be a *dominator set* of Z_i. We recall that $|Z_i| = 4M$ and, from Lemma 7, we have that any subset of $4M$ elements of $\mathcal{Z}$ has dominator size at least $2M$, whence $M + q_i \geq |D_i| \geq 2M$, which implies $q_i \geq M$ as stated above.

The proof for the bound for the parallel model in (5), follows a similar strategy: At least one of the P processors being used, denoted as P^*, must compute at least $|\mathcal{Z}|/P = 4M\left(n/(2\sqrt{M})\right)^{\log 7}/P$ values corresponding to vertices in $\mathcal{Z}$. The bound follows by applying the same argument discussed for the sequential case to the computation executed by P^* (details the extended on-line version [29]). $\square$

Ballard et al. [20] presented a version of Strassen's algorithm whose I/O cost matches the lower bound of Theorem 1 to within a constant factor. Therefore, our bound is asymptotically tight, and the algorithm in [20] is asymptotically I/O optimal. Since in this algorithm no intermediate result is recomputed, recomputation can lead at most to a constant factor reduction of the I/O complexity.

The lower bound of Theorem 1 generalizes to $\Omega((n/\sqrt{M})^{\log_2 7}\frac{M}{B})$ in the *External Memory Model* introduced by Aggarwal and Vitter [31], where $B \geq 1$ values can be moved between cache and consecutive slow memory locations with a single I/O operation.

4 Conclusion

This work has contributed to the characterization of the I/O complexity of Strassen's algorithm by establishing asymptotically tight lower bounds that hold even when recomputation is allowed. Our technique exploits the recursive nature of the CDAG, which makes it promising for the analysis of other recursive algorithms, e.g., for fast rectangular matrix multiplication [32].

The relationship we have exploited between dominator size and Grigoriev's flow points at connections between I/O complexity, (pebbling) space-time tradeoffs [28], and VLSI area-time tradeoffs [33]; these connections deserve further attention.

Some CDAGs for which non-trivial I/O complexity lower bounds are known only in the case of no recomputations are described in [19]. These CDAGs are of interest in the "*limiting technology*" model, defined by fundamental limitations on device size and message speed, as they allow for speedups super-linear in the number of processors. Whether such speedups hold even when recomputation is allowed remains an open question, which our new technique might help answer.

While we know that recomputation may reduce the I/O complexity of some CDAGs, we are far from a characterization of those CDAGs for which recomputation is effective. This broad goal remains a challenge for any attempt toward a general theory of the communication requirements of computations.

References

1. Patterson, C.A., Snir, M., Graham, S.L.: Getting Up to Speed:: The Future of Supercomputing. National Academies Press (2005)
2. Bilardi, G., Preparata, F.P.: Horizons of parallel computation. Journal of Parallel and Distributed Computing **27**(2) (1995) 172–182
3. Strassen, V.: Gaussian elimination is not optimal. Numerische Mathematik **13**(4) (1969) 354–356
4. Le Gall, F.: Powers of tensors and fast matrix multiplication. In: Proc. ACM ISSAC, ACM (2014) 296–303
5. Hong, J., Kung, H.: I/o complexity: The red-blue pebble game. In: Proc. ACM STOC, ACM (1981) 326–333
6. Cannon, L.E.: A cellular computer to implement the Kalman filter algorithm. Technical report, DTIC Document (1969)
7. Ballard, G., Demmel, J., Holtz, O., Lipshitz, B., Schwartz, O.: Brief announcement: strong scaling of matrix multiplication algorithms and memory-independent communication lower bounds. In: Proc. ACM SPAA, ACM (2012) 77–79
8. Irony, D., Toledo, S., Tiskin, A.: Communication lower bounds for distributed-memory matrix multiplication. Journal of Parallel and Distributed Computing **64**(9) (2004) 1017–1026
9. Scquizzato, M., Silvestri, F.: Communication lower bounds for distributed-memory computations. arXiv preprint arXiv:1307.1805 (2013)
10. Pagh, R., Stöckel, M.: The input/output complexity of sparse matrix multiplication. In: Proc. ESA, Springer (2014) 750–761
11. Ballard, G., Demmel, J., Holtz, O., Schwartz, O.: Minimizing communication in numerical linear algebra. SIAM Journal on Matrix Analysis and Applications **32**(3) (2011) 866–901

12. Ballard, G., Demmel, J., Holtz, O., Schwartz, O.: Communication-optimal parallel and sequential Cholesky decomposition. SIAM Journal on Scientific Computing **32**(6) (2010) 3495–3523
13. Loomis, L.H., Whitney, H.: An inequality related to the isoperimetric inequality. Bull. Amer. Math. Soc. **55**(10) (10 1949) 961–962
14. V. A. Zalgaller, A. B. Sossinsky, Y.D.B. The American Mathematical Monthly **96**(6) (1989) 544–546
15. Ballard, G., Demmel, J., Holtz, O., Schwartz, O.: Graph expansion and communication costs of fast matrix multiplication. JACM **59**(6) (2012) 32
16. Ballard, G., Demmel, J., Holtz, O., Lipshitz, B., Schwartz, O.: Graph expansion analysis for communication costs of fast rectangular matrix multiplication. In: Design and Analysis of Algorithms. Springer (2012) 13–36
17. Scott, J., Holtz, O., Schwartz, O.: Matrix multiplication I/O complexity by Path Routing. In: Proc. ACM SPAA. (2015) 35–45
18. De Stefani, L.: On space constrained computations. PhD thesis, University of Padova (2016)
19. Bilardi, G., Preparata, F.: Processor-time trade offs under bounded speed message propagation. Lower Bounds. Theory of Computing Systems **32**(5) (1999) 531–559
20. Ballard, G., Demmel, J., H., O., Lipshitz, B., Schwartz, O.: Communication-optimal parallel algorithm for Strassen's matrix multiplication. In: Proc. ACM SPAA. (2012) 193–204
21. Jacob, R., Stóckel, M.: Fast output-sensitive matrix multiplication. In: Proc. ESA. Springer (2015) 766–778
22. Savage, J.E.: Extending the Hong-Kung model to memory hierarchies. In: Computing and Combinatorics. Springer (1995) 270–281
23. Bilardi, G., Peserico, E.: A characterization of temporal locality and its portability across memory hierarchies. In: Automata, Languages and Programming. Springer (2001) 128–139
24. Koch, R.R., Leighton, F.T., Maggs, B.M., Rao, S.B., Rosenberg, A.L., Schwabe, E.J.: Work-preserving emulations of fixed-connection networks. JACM **44**(1) (1997) 104–147
25. Bhatt, S.N., Bilardi, G., Pucci, G.: Area-time tradeoffs for universal VLSI circuits. Theoret. Comput. Sci. **408**(2-3) (2008) 143–150
26. Bilardi, G., Pietracaprina, A., D'Alberto, P.: On the space and access complexity of computation DAGs. In: Graph-Theoretic Concepts in Computer Science, Springer (2000) 47–58
27. Grigor'ev, D.Y.: Application of separability and independence notions for proving lower bounds of circuit complexity. Zapiski Nauchnykh Seminarov POMI **60** (1976) 38–48
28. Savage, J.E.: Models of Computation: Exploring the Power of Computing. 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1997)
29. Bilardi, G., Stefani, L.D.: The i/o complexity of strassen's matrix multiplication with recomputation. arXiv preprint arXiv:1605.02224 (2016)
30. Ranjan, D., Savage, J.E., Zubair, M.: Upper and lower I/O bounds for pebbling r-pyramids. Journal of Discrete Algorithms **14** (2012) 2–12
31. Aggarwal, A., Vitter, Jeffrey, S.: The input/output complexity of sorting and related problems. Commun. ACM **31**(9) (September 1988) 1116–1127
32. Le Gall, F.: Faster algorithms for rectangular matrix multiplication. In: Proc. IEEE FOCS, IEEE (2012) 514–523
33. Thompson, C.: Area-time complexity for VLSI. In: Proc. ACM STOC, ACM (1979) 81–88