A PARALLEL ALGORITHM FOR BIG TENSOR DECOMPOSITION USING RANDOMLY COMPRESSED CUBES (PARACOMP)

N.D. Sidiropoulos[*]

Dept. of ECE, Univ. of Minnesota
Minneapolis, MN 55455, USA

E.E. Papalexakis, and C. Faloutsos[†]

Dept. of CS, Carnegie-Mellon Univ.
Pittsburgh, PA 15213, USA

ABSTRACT

A parallel algorithm for low-rank tensor decomposition that is especially well-suited for big tensors is proposed. The new algorithm is based on parallel processing of a set of randomly compressed, reduced-size 'replicas' of the big tensor. Each replica is independently decomposed, and the results are joined via a master linear equation per tensor mode. The approach enables massive parallelism with guaranteed identifiability properties: if the big tensor has low rank and the system parameters are appropriately chosen, then the rank-one factors of the big tensor will be exactly recovered from the analysis of the reduced-size replicas. The proposed algorithm is proven to yield memory / storage and complexity gains of order up to $\frac{IJ}{F}$ for a big tensor of size $I \times J \times K$ of rank F with $F \leq I \leq J \leq K$.

***Index Terms*—** Tensor decomposition, CANDECOMP / PARAFAC, Big Data, Parallel and Distributed Computation, Cloud Computing and Storage.

1. INTRODUCTION

Tensors are data structures indexed by three or more indices - a generalization of matrices, which are datasets indexed by two indices (row, column). Tensor algebra has many similarities but also many striking differences with matrix algebra - e.g., determining tensor rank is NP-hard, while low-rank tensor factorization is unique under mild conditions. Tensor factorizations have already found many applications in signal processing (speech, audio, communications, radar, signal intelligence, machine learning) and well beyond. Tensors are becoming increasingly important, especially for analyzing big data, and tensors easily turn really big, e.g., $1000 \times 1000 \times 1000 = 1$ billion entries.

Memory issues related to tensor computations with large but sparse tensors have been considered in [1], [2], and incorporated in the sparse tensor toolbox [3]. The main idea in those references is to avoid intermediate product 'explosion' when computing sequential mode products, but the assumption is that the entire tensor fits in memory (in 'coordinate-wise' representation), and the mode products expand (as opposed to reduce) the size of the 'core' array that they multiply. Adaptive tensor decomposition algorithms for cases where the data is serially acquired (or 'elongated') along one mode have been developed in [4], but these assume that the other two modes are relatively modest in size. More recently, a divide-and-conquer approach to tensor decomposition of large tensors has been proposed in [5], whose idea is to break the data in smaller 'boxes', each one of which is factored independently, and the results are subsequently concatenated using an iterative process. This assumes that each smaller box admits a unique factorization (which cannot be guaranteed from 'global' uniqueness conditions alone), requires reconciling the different permutations and scalings of the different blocks, and significant communication and signaling overhead.

All of the aforementioned techniques require that the full data be stored in (possibly distributed) memory. Realizing that this is a show-stopper for truly big tensors, [6] proposed a random sampling approach, wherein judiciously sampled 'significant' parts of the tensor are independently analyzed, and a common piece of data is used to anchor the different permutations and scalings. The downside of [6] is that it only works for sparse tensors, and it offers no identifiability guarantees - although it usually works well for sparse tensors. A different approach was taken in [7], which proposed *randomly* compressing a big tensor down to a far smaller one. Assuming that the big tensor admits a low-rank decomposition with sparse latent factors, such a random compression guarantees identifiability of the low-rank decomposition of the big tensor from the low-rank decomposition of the small tensor. The line of proof is a generalization of compressed sensing ideas from the linear to the multi-linear case. Still, this approach works only when the latent low-rank factors of the big tenor are known to be sparse - and this is often not the case.

This paper considers appropriate compression strategies for big (sparse or dense) tensors that admit a low-rank decomposition / approximation, whose latent factors need not be sparse. Latent sparsity is usually associated with membership problems such as clustering and co-clustering [8]. A novel parallel algorithm for low-rank tensor decomposition that is especially well-suited for big tensors is proposed. The new algorithm is based on parallel processing of a set of randomly compressed, reduced-size 'replicas' or the big tensor. Each replica is independently decomposed, and the results are joined via a master linear equation per tensor mode. The approach enables massive parallelism with guaranteed identifiability properties: if the big tensor has low rank, and the system parameters are appropriately chosen, then the rank-one factors of the big tensor will indeed be recovered from the analysis of the reduced-size replicas. Furthermore, the approach affords memory / storage and complexity gains of order up to $\frac{IJ}{F}$ for a big tensor of size $I \times J \times K$ of rank F with $F \leq I \leq J \leq K$. No sparsity is required in the tensor or the underlying latent factors, although such sparsity can be exploited to improve memory, storage and computational savings.

2. TENSOR DECOMPOSITION: PRELIMINARIES

What is a *tensor*? A matrix is a dataset indexed by two indices, say (r, c) for (row,column). A tensor is a dataset indexed by three

[*]E-mail: nikos@umn.edu, Tel: +16126251242, Fax: +16126254583. Supported by NSF IIS-1247632.

[†]E-mail: (epapalex,christos)@cs.cmu.edu. Supported by NSF IIS-1247489.

or more indices, say $(i, j, k, \cdots)$. The term *tensor* has a different meaning in Physics, however it has been widely adopted in recent years to describe what was previously known as a *multi-way array*. Matrices are two-way tensors, and they are special because it turns out that there is an interesting dichotomy between two-way and three- or higher-way tensors, with the latter sharing common algebraic properties which are simply very different from those of matrices.

Notation: A scalar is denoted by an italic letter, e.g. a. A column vector is denoted by a bold lowercase letter, e.g. $\mathbf{a}$ whose i-th entry is $\mathbf{a}(i)$. A matrix is denoted by a bold uppercase letter, e.g., $\mathbf{A}$ with (i, j)-th entry $\mathbf{A}(i, j)$; $\mathbf{A}(:, j)$ ($\mathbf{A}(i, :)$) denotes the j-th column (resp. i-th row) of $\mathbf{A}$. A three-way array is denoted by an underlined bold uppercase letter, e.g., $\underline{\mathbf{X}}$, with (i, j, k)-th entry $\underline{\mathbf{X}}(i, j, k)$. Vector, matrix and three-way array size parameters (mode lengths) are denoted by uppercase letters, e.g. I. $\circ$ stands for the vector outer product: for two vectors $\mathbf{a}$ ($I \times 1$) and $\mathbf{b}$ ($J \times 1$), $\mathbf{a} \circ \mathbf{b}$ is an $I \times J$ rank-one matrix with (i, j)-th element $\mathbf{a}(i)\mathbf{b}(j)$; i.e., $\mathbf{a} \circ \mathbf{b} = \mathbf{a}\mathbf{b}^T$. For three vectors, $\mathbf{a}$ ($I \times 1$), $\mathbf{b}$ ($J \times 1$), $\mathbf{c}$ ($K \times 1$), $\mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$ is an $I \times J \times K$ three-way array with (i, j, k)-th element $\mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k)$. The vec$(\cdot)$ operator stacks the columns of its matrix argument in one tall column; $\otimes$ stands for the Kronecker product; $\odot$ stands for the Khatri-Rao (column-wise Kronecker) product: given $\mathbf{A}$ ($I \times F$) and $\mathbf{B}$ ($J \times F$), $\mathbf{A} \odot \mathbf{B}$ is the $JI \times F$ matrix

$$\mathbf{A} \odot \mathbf{B} = \big[\mathbf{A}(:, 1) \otimes \mathbf{B}(:, 1) \cdots \mathbf{A}(:, F) \otimes \mathbf{B}(:, F)\big]$$

Rank decomposition: The rank of an $I \times J$ matrix $\mathbf{X}$ is the smallest number of rank-one matrices (vector outer products of the form $\mathbf{a} \circ \mathbf{b}$) needed to synthesize $\mathbf{X}$ as

$$\mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f \circ \mathbf{b}_f = \mathbf{A}\mathbf{B}^T,$$

where $\mathbf{A} := [\mathbf{a}_1, \cdots, \mathbf{a}_F]$, and $\mathbf{B} := [\mathbf{b}_1, \cdots, \mathbf{b}_F]$. This relation can be expressed element-wise as

$$\mathbf{X}(i, j) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j).$$

The rank of an $I \times J \times K$ three-way array $\underline{\mathbf{X}}$ is the smallest number of outer products needed to synthesize $\underline{\mathbf{X}}$ as

$$\underline{\mathbf{X}} = \sum_{f=1}^{F} \mathbf{a}_f \circ \mathbf{b}_f \circ \mathbf{c}_f.$$

This relation can be expressed element-wise as

$$\underline{\mathbf{X}}(i, j, k) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j)\mathbf{c}_f(k).$$

In the sequel we will assume that F is minimal, i.e., $F = \text{rank}(\underline{\mathbf{X}})$, unless otherwise noted. The tensor $\underline{\mathbf{X}}$ comprises K 'frontal' slabs of size $I \times J$; denote them $\{\mathbf{X}_k\}_{k=1}^{K}$, with $\mathbf{X}_k := \underline{\mathbf{X}}(:, :, k)$. Re-arranging the elements of $\underline{\mathbf{X}}$ in a tall matrix $\mathbf{X} := [\text{vec}(\mathbf{X}_1), \cdots, \text{vec}(\mathbf{X}_K)]$, it can be shown that

$$\mathbf{X} = (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T \Longleftrightarrow \mathbf{x} := \text{vec}(\mathbf{X}) = (\mathbf{C} \odot \mathbf{B} \odot \mathbf{A})\,\mathbf{1},$$

where, $\mathbf{A}$, $\mathbf{B}$ are as defined for the matrix case, $\mathbf{C} := [\mathbf{c}_1, \cdots, \mathbf{c}_F]$, $\mathbf{1}$ is a vector of all 1's, and we have used the vectorization property of the Khatri-Rao product $\text{vec}(\mathbf{A}\mathbf{D}(\mathbf{d})\mathbf{B}^T) = (\mathbf{B} \odot \mathbf{A})\,\mathbf{d}$, where $\mathbf{D}(\mathbf{d})$ is a diagonal matrix with the vector $\mathbf{d}$ as its diagonal.

CANDECOMP-PARAFAC: The above rank decomposition model for tensors is known as *parallel factor analysis* (PARAFAC) [9, 10] or *canonical decomposition* (CANDECOMP) [11], or CP for CANDECOMP-PARAFAC. CP is in a way the most basic tensor model, because of its direct relationship to tensor rank and the concept of rank decomposition; but other algebraic tensor models exist, and we refer the reader to [12, 13] for gentle introductions to tensor decompositions and applications.

Uniqueness: The key feature of the CP model is its essential uniqueness: under certain conditions, $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ can be identified from $\mathbf{X}$ up to permutation and scaling [9–11, 14–18]. The *Kruskal-rank* of $\mathbf{A}$, denoted $k_{\mathbf{A}}$, is the maximum k such that *any* k columns of $\mathbf{A}$ are linearly independent ($k_{\mathbf{A}} \leq r_{\mathbf{A}} := \text{rank}(\mathbf{A})$). Given $\underline{\mathbf{X}}$ ($\Leftrightarrow \mathbf{x}$), $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ are unique up to a common column permutation and scaling (e.g., scaling the first column of $\mathbf{A}$ and counter-scaling the first column of $\mathbf{B}$ and/or $\mathbf{C}$, so long as their product remains the same), provided that $k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2F + 2$. This is Kruskal's celebrated uniqueness result, see [14–17]. Kruskal's result applies to given $(\mathbf{A}, \mathbf{B}, \mathbf{C})$, i.e., it can establish uniqueness of a given decomposition. Recently, more relaxed uniqueness conditions have been obtained, which only depend on the size and rank of the tensor - albeit they cover *almost* all tensors of the given size and rank, i.e., except for a set of measure zero. The latest available condition on this front is the following.

Theorem 1 *[18] Consider an $I \times J \times K$ tensor $\underline{\mathbf{X}}$ of rank F, and order the dimensions so that $I \leq J \leq K$ Let i be maximal such that $2^i \leq I$, and likewise j maximal such that $2^j \leq J$. If $F \leq 2^{i+j-2}$, then $\underline{\mathbf{X}}$ has a unique decomposition almost surely. For I, J powers of 2, the condition simplifies to $F \leq \frac{IJ}{4}$. More generally, the condition implies that if $F \leq \frac{(I+1)(J+1)}{16}$, then $\underline{\mathbf{X}}$ has a unique decomposition almost surely.*

3. BIG TENSOR COMPRESSION

When dealing with big tensors $\underline{\mathbf{X}}$ that do not fit in main memory, a reasonable idea is to try to compress $\underline{\mathbf{X}}$ to a much smaller tensor that somehow captures most of the systematic variation in $\underline{\mathbf{X}}$. The commonly used compression method is to fit a low-dimensional orthogonal Tucker3 model (with low mode-ranks) [12,13], then regress the data onto the fitted mode-bases. This idea has been exploited in existing PARAFAC model-fitting software, such as COMFAC [19], as a useful quick-and-dirty way to initialize alternating least squares computations in the uncompressed domain, thus accelerating convergence. A key issue with Tucker3 compression of big tensors is that it requires computing singular value decompositions of the various matrix unfoldings of the full data, in an alternating fashion. This is a serious bottleneck for big data. Another issue is that Tucker3 compression is lossy, and it cannot guarantee that identifiability properties will be preserved. Finally, fitting a PARAFAC model to the compressed data can only yield an approximate model for the original uncompressed data, and eventually decompression and iterations with the full data are required to obtain fine estimates.

Consider compressing $\mathbf{x}$ into $\mathbf{y} = \mathbf{S}\mathbf{x}$, where $\mathbf{S}$ is $d \times IJK$, $d \ll IJK$. Sidiropoulos & Kyrillidis [7] proposed using a specially structured compression matrix $\mathbf{S} = \mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T$, which corresponds to multiplying (every slab of) $\underline{\mathbf{X}}$ from the I-mode with $\mathbf{U}^T$, from the J-mode with $\mathbf{V}^T$, and from the K-mode with $\mathbf{W}^T$, where $\mathbf{U}$ is $I \times L$, $\mathbf{V}$ is $J \times M$, and $\mathbf{W}$ is $K \times N$, with $L \leq I$, $M \leq J$, $N \leq K$ and $LMN \ll IJK$. Such an $\mathbf{S}$ corresponds

to compressing each mode individually, which is often natural, and the associated multiplications can be efficiently implemented when the tensor is sparse. Due to a fortuitous property of the Kronecker product [20],

$$\left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)(\mathbf{A} \odot \mathbf{B} \odot \mathbf{C}) =$$

$$\left((\mathbf{U}^T\mathbf{A}) \odot (\mathbf{V}^T\mathbf{B}) \odot (\mathbf{W}^T\mathbf{C})\right),$$

from which it follows that

$$\mathbf{y} = \left((\mathbf{U}^T\mathbf{A}) \odot (\mathbf{V}^T\mathbf{B}) \odot (\mathbf{W}^T\mathbf{C})\right)\mathbf{1} = \left(\tilde{\mathbf{A}} \odot \tilde{\mathbf{B}} \odot \tilde{\mathbf{C}}\right)\mathbf{1}.$$

i.e., the compressed data follow a PARAFAC model of size $L \times M \times N$ and order F parameterized by $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}})$, with $\tilde{\mathbf{A}} := \mathbf{U}^T\mathbf{A}$, $\tilde{\mathbf{B}} := \mathbf{V}^T\mathbf{B}$, $\tilde{\mathbf{C}} := \mathbf{W}^T\mathbf{C}$.

Remark 1 *It can be shown that multi-way tensor compression (i.e., computing $\underline{\mathbf{Y}}$ from $\underline{\mathbf{X}}$, or, equivalently, $\mathbf{y}$ from $\mathbf{x}$) can be accomplished at computational complexity $O(\max(L, M, N)IJK)$ if memory and speed of memory access are not an issue, or $O(LMNIJK)$ if memory and access are severely limited. If $\underline{\mathbf{X}}$ has only $NZ(\underline{\mathbf{X}})$ nonzero elements, then computational complexity can be reduced to $NZ(\underline{\mathbf{X}})LMN$ without requiring any extra memory or memory access. We skip details due to space limitations, but full details will be included in the journal version.*

Using a Lemma in [7] and Theorem 1 from [18], we have established the following result.

Theorem 2 *Let $\mathbf{x} = (\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\mathbf{1} \in \mathbb{R}^{IJK}$, where $\mathbf{A}$ is $I \times F$, $\mathbf{B}$ is $J \times F$, $\mathbf{C}$ is $K \times F$, and consider compressing it to $\mathbf{y} = \left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)\mathbf{x} = \left((\mathbf{U}^T\mathbf{A}) \odot (\mathbf{V}^T\mathbf{B}) \odot (\mathbf{W}^T\mathbf{C})\right)\mathbf{1} = \left(\tilde{\mathbf{A}} \odot \tilde{\mathbf{B}} \odot \tilde{\mathbf{C}}\right)\mathbf{1} \in \mathbb{R}^{LMN}$, where the mode-compression matrices $\mathbf{U}$ ($I \times L, L \leq I$), $\mathbf{V}$ ($J \times M, M \leq J$), and $\mathbf{W}$ ($K \times N, N \leq K$) are independently drawn from an absolutely continuous distribution with respect to the Lebesgue measure in $\mathbb{R}^{IL}$, $\mathbb{R}^{JM}$, and $\mathbb{R}^{KN}$, respectively. If $F \leq \min(I, J, K)$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are all full column rank (F), $L \leq M \leq N$, and*

$$(L + 1)(M + 1) \geq 16F,$$

then $\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}}$ are almost surely identifiable from the compressed data $\mathbf{y}$ up to a common column permutation and scaling.

Theorem 2 can establish uniqueness of $\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}}$, but we are ultimately interested in $\mathbf{A}, \mathbf{B}, \mathbf{C}$. We know that $\tilde{\mathbf{A}} = \mathbf{U}^T\mathbf{A}$, and we know $\mathbf{U}^T$, but, unfortunately, it is a fat matrix that cannot be inverted. In order to uniquely recover $\mathbf{A}$, one needs additional structural constraints. Sidiropoulos & Kyrillidis [7] proposed exploiting column-wise sparsity in $\mathbf{A}$ (and likewise $\mathbf{B}, \mathbf{C}$), which is often plausible in practice[1]. Sparsity is a powerful constraint, but it is not always valid (or a sparsifying basis may be unknown). For this reason, we propose here a different solution, based on creating and factoring a number of randomly reduced 'replicas' of the full data.

[1] $\mathbf{A}$ need only be sparse with respect to (when expressed in) a suitable basis, provided the sparsifying basis is known *a priori*.

4. THE PARACOMP APPROACH

Consider spawning P randomly compressed reduced-size 'replicas' $\{\underline{\mathbf{Y}}_p\}_{p=1}^P$ of the tensor $\underline{\mathbf{X}}$, where $\underline{\mathbf{Y}}_p$ is created using mode compression matrices $(\mathbf{U}_p, \mathbf{V}_p, \mathbf{W}_p)$, see Fig. 1. Assume that identifiability conditions per Theorem 2 hold, so that $\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p$ are almost surely identifiable (up to permutation and scaling of columns) from $\underline{\mathbf{Y}}_p$. Then, upon factoring $\underline{\mathbf{Y}}_p$ into F rank-one components, one will obtain

$$\tilde{\mathbf{A}}_p = \mathbf{U}_p^T\mathbf{A}\mathbf{\Pi}_p\mathbf{\Lambda}_p. \tag{1}$$

Assume that the first two columns of each $\mathbf{U}_p$ (rows of $\mathbf{U}_p^T$) are common, and let $\bar{\mathbf{U}}$ denote this common part, and $\bar{\mathbf{A}}_p$ denote the first two rows of $\tilde{\mathbf{A}}_p$. We therefore have

$$\bar{\mathbf{A}}_p = \bar{\mathbf{U}}^T\mathbf{A}\mathbf{\Pi}_p\mathbf{\Lambda}_p.$$

Dividing each column of $\bar{\mathbf{A}}_p$ by the element of maximum modulus in that column, and denoting the resulting $2 \times F$ matrix $\hat{\mathbf{A}}_p$, we obtain

$$\hat{\mathbf{A}}_p = \bar{\mathbf{U}}^T\mathbf{A}\mathbf{\Lambda}\mathbf{\Pi}_p.$$

Notice that $\mathbf{\Lambda}$ does not affect the ratio of elements in each 2×1 column. If these ratios are distinct (which is guaranteed almost surely if $\bar{\mathbf{U}}$ and $\mathbf{A}$ are independently drawn from absolutely continuous distributions), then the different permutations can be matched by sorting the ratios of the two coordinates of each 2×1 column of $\hat{\mathbf{A}}_p$. In practice using a few more 'anchor' rows will improve the permutation-matching performance, and is recommended in difficult cases with high noise variance. When S anchor rows are used, the optimal permutation matching problem can be cast as a *Linear Assignment Problem* (LAP), which can be efficiently solved using the *Hungarian Algorithm*. After this column permutation-matching process, we go back to (1) and permute its columns to obtain $\breve{\mathbf{A}}_p$ satisfying

$$\breve{\mathbf{A}}_p = \mathbf{U}_p^T\mathbf{A}\mathbf{\Pi}\mathbf{\Lambda}_p.$$

It remains to get rid of $\mathbf{\Lambda}_p$. For this, we can again resort to the first two common rows, and divide each column of $\breve{\mathbf{A}}_p$ with its top element. This finally yields

$$\breve{\mathbf{A}}_p = \mathbf{U}_p^T\mathbf{A}\mathbf{\Pi}\mathbf{\Lambda}.$$

For recovery of $\mathbf{A}$ up to permutation and scaling of its columns, we then require that the matrix of the linear system

$$\begin{bmatrix} \breve{\mathbf{A}}_1 \\ \vdots \\ \breve{\mathbf{A}}_P \end{bmatrix} = \begin{bmatrix} \mathbf{U}_1^T \\ \vdots \\ \mathbf{U}_P^T \end{bmatrix} \mathbf{A}\mathbf{\Pi}\mathbf{\Lambda} \tag{2}$$

be full column rank. This implies that

$$2 + \sum_{p=1}^{P}(L_p - 2) \geq I$$

i.e.,

$$\sum_{p=1}^{P}(L_p - 2) \geq I - 2.$$

Note that every sub-matrix contains the two anchor rows which are common, and duplicate rows clearly do not increase the rank. Also note that once the dimensionality requirement is met, the matrix will be full rank with probability 1, because its non-redundant entries are drawn from a jointly continuous distribution (by design).

Assuming $L_p = L, \forall p \in \{1, \cdots, P\}$ for simplicity (and symmetry of computational load), we obtain $P(L-2) \geq I-2$, or, in terms of the number of threads $P \geq \frac{I-2}{L-2}$. Likewise, from the corresponding full column rank requirements for the other two modes, we obtain $P \geq \frac{J}{M}$, and $P \geq \frac{K}{N}$. Notice that we do not subtract 2 from numerator and denominator for the other two modes, because the permutation of columns of $\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p$ is *common* - so it is enough to figure it out from one mode, and apply it to other modes as well. One can pick the mode used to figure out the permutation ambiguity, leading to a symmetrized condition. If the compression ratios in the different modes are similar, it makes sense to use the longest mode for this purpose; if this is the last mode, then

$$P \geq \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$$

We have thus established the following result.

Theorem 3 *In reference to Fig. 1, assume* $\mathbf{x} := vec\left(\underline{\mathbf{X}}\right) = (\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\,\mathbf{1} \in \mathbb{R}^{IJK}$, *where* $\mathbf{A}$ *is* $I \times F$, $\mathbf{B}$ *is* $J \times F$, $\mathbf{C}$ *is* $K \times F$ *(i.e., the rank of* $\underline{\mathbf{X}}$ *is at most* F*). Assume that* $F \leq I \leq J \leq K$, *and* $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ *are all full column rank* (F)*. Further assume that* $L_p = L$, $M_p = M$, $N_p = N$, $\forall p \in \{1, \cdots, P\}$, $L \leq M \leq N$, $(L+1)(M+1) \geq 16F$, *the elements of* $\{\mathbf{U}_p\}_{p=1}^P$ *are drawn from a jointly continuous distribution, and likewise for* $\{\mathbf{V}_p\}_{p=1}^P$, *while each* $\mathbf{W}_p$ *contains two common anchor columns, and the elements of* $\{\mathbf{W}_p\}_{p=1}^P$ *(except for the repeated anchors, obviously) are drawn from a jointly continuous distribution. Then the data for each thread* $\mathbf{y}_p := vec\left(\underline{\mathbf{Y}}_p\right)$ *can be uniquely factored, i.e.,* $\left(\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p\right)$ *is unique up to column permutation and scaling. If, in addition to the above, we also have* $P \geq \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$ *parallel threads, then* $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ *are almost surely identifiable from the thread outputs* $\left\{\left(\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p\right)\right\}_{p=1}^P$ *up to a common column permutation and scaling.*

The above result is indicative of a family of results that can be derived. Its significance may not be immediately obvious, so it is worth elaborating further at this point. On one hand, Theorem 3 shows that fully parallel computation of the big tensor decomposition is possible – the first such result, to the best of our knowledge, that *guarantees* identifiability of the big tensor decomposition from the intermediate small tensor decompositions, without placing stringent additional constraints. On the other hand, the memory/storage and computational savings are not necessarily easy to see. The following claim nails down the take-home message.

Claim 1 *Under the conditions of Theorem 3, if* $\frac{K-2}{N-2} = \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$, *then the memory / storage and computational complexity savings afforded by the PARACOMP approach in Fig. 1 relative to brute-force computation are of order* $\frac{IJ}{F}$.

5. SIMULATIONS

For simulations, we set $I = J = K = 500$ and $L = M = N = 50$, so $P \geq 11 \geq 10.375 = \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$ for identifiability. The big tensor has 125M elements, or about 1Gbyte if stored in double precision. This allows in-memory factoring of the big tensor for comparison purposes. Each small tensor has 125K elements, or about 1Mbyte in double precision; for $P = 11$, *the overall storage needed for all 11 small tensors is only 11Mbytes, or about 1% of the big tensor*. We set $F = 5$ and generate the noiseless tensor by randomly and independently drawing $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ from `randn(500,5)`, and then taking the sum of outer products of their corresponding columns. Gaussian i.i.d. measurement noise of standard deviation $\sigma = 0.01$ is added to the noiseless tensor to produce $\underline{\mathbf{X}}$. Fig. 6 shows $||\mathbf{A} - \hat{\mathbf{A}}||_2^2$, where $\hat{\mathbf{A}}$ denotes the PARACOMP estimate of $\mathbf{A}$, as a function of P. The baseline is the total squared error attained by directly fitting the uncompressed $\underline{\mathbf{X}}$. PARACOMP yields respectable accuracy with only 1.2% of the full data, and total squared error of order $10^{-6} - 10^{-5}$ with 24% of the full data, vs. 10^{-7} for the baseline algorithm that has access to 100% of the data.

6. CONCLUSIONS

We have proposed a new parallel algorithm for tensor decomposition that is especially well-suited for big tensors. This new approach is the first to enable complete parallelism of the core computation together with substantial memory/storage savings *and* identifiability guarantees. An in-depth study of pertinent trade-offs and extensions is currently underway, and the results will be reported in follow-up publications.

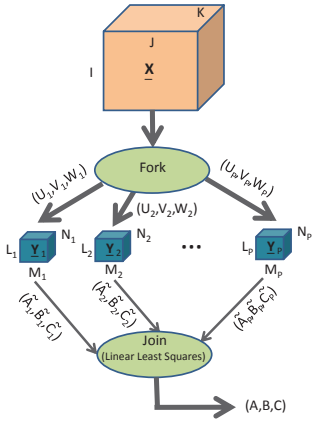

Fig. 1. The fork spawns $\left\{\underline{\mathbf{Y}}_p\right\}_{p=1}^P$, where $\underline{\mathbf{Y}}_p$ is obtained by applying $(\mathbf{U}_p, \mathbf{V}_p, \mathbf{W}_p)$ to $\underline{\mathbf{X}}$. Each $\underline{\mathbf{Y}}_p$ is independently factored in parallel. The join step anchors all $\left\{\tilde{\mathbf{A}}_p\right\}_{p=1}^P$ to a common reference, and solves a linear problem for the overall $\mathbf{A}$; likewise for $\mathbf{B}$, $\mathbf{C}$.

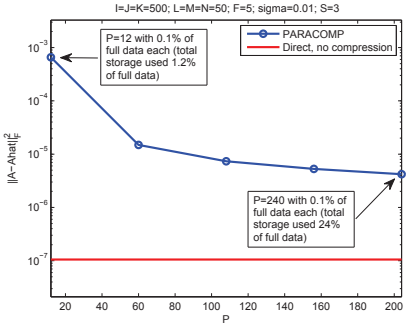

Fig. 2. MSE vs. # of cores/repetitions P.

7. REFERENCES

[1] B.W. Bader and T.G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 205–231, December 2007.

[2] T.G. Kolda and J. Sun, "Scalable tensor decompositions for multi-aspect data mining," in *ICDM 2008: Proceedings of the 8th IEEE International Conference on Data Mining*, December 2008, pp. 363–372.

[3] B.W. Bader and T.G. Kolda, "Matlab tensor toolbox version 2.5," `http://www.sandia.gov/~tgkolda/TensorToolbox/`, January 2012.

[4] D. Nion and N.D. Sidiropoulos, "Adaptive algorithms to track the parafac decomposition of a third-order tensor," *IEEE Trans. on Signal Processing*, vol. 57, no. 6, pp. 2299–2310, 2009.

[5] A.H. Phan and A. Cichocki, "Parafac algorithms for large-scale problems," *Neurocomputing*, vol. 74, no. 11, pp. 1970–1984, 2011.

[6] Evangelos E. Papalexakis, Christos Faloutsos, and Nicholas D. Sidiropoulos, "Parcube: Sparse parallelizable tensor decompositions.," in *ECML/PKDD (1)*, Peter A. Flach, Tijl De Bie, and Nello Cristianini, Eds. 2012, vol. 7523 of *Lecture Notes in Computer Science*, pp. 521–536, Springer.

[7] N.D. Sidiropoulos and A. Kyrillidis, "Multi-way compressed sensing for sparse low-rank tensors," *IEEE Signal Processing Letters*, vol. 19, no. 11, pp. 757–760, 2012.

[8] E.E. Papalexakis, N.D. Sidiropoulos, and R. Bro, "From k -means to higher-way co-clustering: Multilinear decomposition with sparse latent factors," *IEEE Trans. on Signal Processing*, vol. 61, no. 2, pp. 493–506, 2013.

[9] R.A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, pp. 1–84, 1970.

[10] R.A. Harshman, "Determination and proof of minimum uniqueness conditions for PARAFAC-1," *UCLA Working Papers in Phonetics*, vol. 22, pp. 111–117, 1972.

[11] J.D. Carroll and J.J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[12] A.K. Smilde, R. Bro, P. Geladi, and J. Wiley, *Multi-way analysis with applications in the chemical sciences*, Wiley, 2004.

[13] P.M. Kroonenberg, *Applied multiway data analysis*, Wiley, 2008.

[14] J.B. Kruskal, "Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics," *Linear Algebra and its Applications*, vol. 18, no. 2, pp. 95–138, 1977.

[15] N.D. Sidiropoulos and R. Bro, "On the uniqueness of multilinear decomposition of N-way arrays," *Journal of chemometrics*, vol. 14, no. 3, pp. 229–239, 2000.

[16] T. Jiang and N.D. Sidiropoulos, "Kruskal's permutation lemma and the identification of CANDECOMP/PARAFAC and bilinear models with constant modulus constraints," *IEEE Transactions on Signal Processing*, vol. 52, no. 9, pp. 2625–2636, 2004.

[17] A. Stegeman and N.D. Sidiropoulos, "On Kruskal's uniqueness condition for the CANDECOMP/PARAFAC decomposition," *Linear Algebra and its Applications*, vol. 420, no. 2-3, pp. 540–552, 2007.

[18] L. Chiantini and G. Ottaviani, "On generic identifiability of 3-tensors of small rank," *SIAM. J. Matrix Anal. & Appl.*, vol. 33, no. 3, pp. 1018–1037, 2012.

[19] R. Bro, N.D. Sidiropoulos, and G.B. Giannakis, "A fast least squares algorithm for separating trilinear mixtures," in *Proc. ICA99 Int. Workshop on Independent Component Analysis and Blind Signal Separation*, 1999, pp. 289–294.

[20] J.W. Brewer, "Kronecker products and matrix calculus in system theory," *IEEE Trans. on Circuits and Systems*, vol. 25, no. 9, pp. 772–781, 1978.