General Memory-Independent Lower Bound for MTTKRP*

Grey Ballard† Kathryn Rouse‡

Abstract

Our goal is to establish lower bounds on the communication required to perform the Matricized-Tensor Times Khatri-Rao Product (MTTKRP) computation on a distributed-memory parallel machine. MTTKRP is the bottleneck computation within algorithms for computing the CP tensor decomposition, which is an approximation by a sum of rank-one tensors and frequently used in multidimensional data analysis. The main result of this paper is a communication lower bound that generalizes previous results, tightening the bound so that it is attainable even when the tensor dimensions vary (the tensor is not cubical) and when the number of processors is small relative to the tensor dimensions. The attainability of the bound proves that the algorithm that attains it, which is based on a block distribution of the tensor and communicating only factor matrices, is communication optimal. The proof technique utilizes an established inequality that relates computations to data access as well as a novel approach based on convex optimization.

1 Introduction

The CP decomposition approximates a tensor by a sum of rank-one tensors, where a rank-one tensor is an outer product of a collection of vectors. CP, which is also known as canonical polyadic and CANDECOMP/PARAFAC, is used for example to discover latent patterns or detect anomalies in multidimensional data sets. It is particularly useful when the underlying components are to be interpreted, and adding domain-specific constraints to CP can help identify meaningful representations of the data.

While there are many popular algorithms for computing CP decompositions, nearly all of them are computationally bound by an operation known as Matricized-Tensor Times Khatri-Rao Product, or MTTKRP [11]. This operation is expensive compared to others within the algorithms because it involves the original data tensor, which is typically much larger than the CP approximation. The computational cost of MTTKRP scales linearly with the number of tensor entries as well as the rank of the decomposition, or the number of rank-one components to be computed. Not only is MTTKRP the computational bottleneck for CP algorithms, it also tends to require the most communication. Thus, in order to efficiently compute CP decompositions, it is necessary to use MTTKRP algorithms that minimize both computation and communication.

The goal of this paper is to establish communication lower bounds for MTTKRP in the case of large, dense tensors that are distributed across the memories of a parallel computer. Lower bounds set targets for algorithmic development, and identifying communication-optimal algorithms that attain them can guide lower-level and more hardware-specific optimizations. Previous work by Ballard, Knight, and Rouse [4] establishes lower bounds for sequential and parallel MTTKRP computations. While the parallel bound is generally applicable, it is not attainable when either (1) the number of processors is not sufficiently large or (2) the different dimensions of the tensor are not sufficiently cubical (nearly equal). Our main result, given in Section 5, generalizes this previous bound and tightens it in those cases so that it is attainable for nearly all tensor dimensions and numbers of processors. We show that the new lower bound is tight by proving that an existing algorithm can attain it (to within constant factors) with the right tuning parameters and is therefore optimal.

The basic idea of the communication-optimal algorithm is to organize the processors into a multidimensional grid with as many modes as the data tensor and assign tensor entries to processors in a block, or Cartesian, fashion. Given this tensor distribution, the algorithm follows an owner-computes rule based on tensor entries so that only the matrices that represent the CP decomposition are communicated and the tensor remains stationary. The algorithm can be tuned by varying the processor grid dimensions, which affects the amount of communication performed. In the case that all the dimensions of the optimal processor grid are larger than 1, the previous communication lower bound is attained. However, when some set of dimensions of the optimal processor grid is equal to 1, that bound is not attained. We show in this paper that the algorithm

*This material is based upon work supported by the National Science Foundation under Grant No. OAC-1642385.

†Department of Computer Science; Wake Forest University; PO Box 7311; Winston-Salem NC 27109, USA; ballard@wfu.edu.

‡Inmar Analytics Department; Inmar Intelligence, A Data and Tech-enabled Services Company; 635 Vine Street; Winston-Salem, NC, 27101, USA; kathryn.rouse@inmar.com

using such a processor grid is indeed communication optimal by tightening the lower bound for that case. We give more details of this algorithm in Section 3.

The proof of the lower bound has two main ingredients. The first ingredient is a Hölder-Brascamp-Lieb-type inequality [5] that has previously been demonstrated to be useful in relating the volume of a subset of computation to the size of projections of that computation onto the data it requires [7, 10]. The second ingredient is a novel technique that casts communication costs as the objective function value in a constrained optimization problem, where the constraint set corresponds to valid algorithms (subject to certain assumptions). We show that the optimization problem is convex, and we use results from convex optimization to establish the global minimum, which serves as the lower bound. We believe this lower bound technique can be applicable for other computations as well. Section 4 details the result for the 3D case, which allows for cleaner notation and intuition, and the general result is presented in Section 5. We argue the optimality of the algorithm and tightness of the lower bound in Section 6.

The main result states that, for a given number of processors, the factor matrices can be divided into J "small" matrices and $N-J$ "large" matrices. Any MTTKRP algorithm that does not communicate the tensor requires that each processor must communicate all of the small matrices and equal amounts of (parts of) the large matrices. This yields a cost of

$$\Theta\left(\sum_{j=1}^{J} I_j R + (N-J)\left(\frac{\prod_{k=J+1}^{N} I_k}{P}\right)^{\frac{1}{N-J}} R\right)$$

words communicated, where P is the number of processors, R is the CP rank, and the tensor dimensions $\{I_n\}$ are ordered from smallest to largest. We state this in full detail in Theorem 5.1.

2 Preliminaries

2.1 CP and MTTKRP Notation We'll use the notation $\mathcal{T}$ to represent an N-way tensor of dimensions $I_1 \times I_2 \times \cdots \times I_N$ where $N > 2$, and we'll assume throughout that the tensor modes are ordered so that $I_1 \leq I_2 \leq \cdots \leq I_N$. A CP decomposition of $\mathcal{T}$ will be represented by a rank R and set of N factor matrices $\{\mathbf{U}^{(n)}\}_{n\in[N]}$ such that $\mathbf{U}^{(n)}$ has dimension $I_n \times R$, where

$$\mathcal{T}(i_1, \ldots, i_N) \approx \sum_{r=1}^{R} \mathbf{U}^{(1)}(i_1, r) \cdots \mathbf{U}^{(N)}(i_N, r).$$

The notation $[N]$ denotes the set $\{1, 2, \ldots, N\}$, and $[N] - \{n\}$ denotes the set $\{1, \ldots, n-1, n+1, \ldots, N\}$. For illustrative examples (and to simplify notation), we will use 3-way tensors with dimensions $I \times J \times K$ that are approximated by factor matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$.

We next define the MTTKRP operation, which involves the tensor and the factor matrices of a CP decomposition. MTTKRP is often presented in matrix notation as $\mathbf{T}_{(n)}(\mathbf{U}^{(N)} \odot \cdots \mathbf{U}^{(n+1)} \odot \mathbf{U}^{(n-1)} \cdots \odot \mathbf{U}^{(1)})$, where $\mathbf{T}_{(n)}$ is a matricization, or flattening, of the tensor and $\odot$ is the Khatri-Rao product, which can be thought of as a column-wise Kronecker product or a row-wise Hadamard (elementwise) product. We show in Section 3.1 why MTTKRP is central to algorithms for computing CP.

Definition 2.1. *[4, Definition 2.1] An MTTKRP algorithm maps*

$$\left(\mathcal{T}, \left\{\mathbf{U}^{(k)}\right\}_{k\in[N]-\{n\}}\right) \mapsto \mathbf{M}^{(n)},$$

where for each $(i_n, r) \in [I_n] \times [R]$,

$$\mathbf{M}^{(n)}(i_n, r) = \sum_{\mathbf{i}} \mathcal{T}(\mathbf{i}) \prod_{k\in[N]-\{n\}} \mathbf{U}^{(k)}(i_k, r),$$

where the summation occurs over all $\mathbf{i} \in [I_1] \times \cdots \times [I_N]$ *with* n*-th entry* i_n*. We require that the* $N-1$ *multiplies for each* $\mathbf{i}$ *are evaluated atomically as an* N*-ary multiply.*

Each N-ary multiply performed by an MTTKRP algorithm corresponds to a point $(i_1, \ldots, i_N, r)$ in the iteration space $[I_1] \times \cdots \times [I_N] \times [R]$. We let F be some subset of points in the iteration space corresponding to a subset of N-ary multiplies. We define the projections of F onto the tensor and factor matrices by $\phi_0(F) = \{(i_1, \ldots, i_N) | (i_1, \ldots, i_N, r) \in F\}$, the projection onto the tensor $\mathcal{T}$, and $\phi_m(F) = \{(i_m, r) | (i_1, \ldots, i_m, \ldots, i_N, r) \in F\}$, the projection onto $\mathbf{U}^{(m)}$ if $m \neq n$, or $\mathbf{M}^{(n)}$ if $m = n$. This means that $\phi_0(F)$ corresponds to the set of the tensor entries required to perform the N-ary multiplies of F, and $\phi_m(F)$ corresponds to the set of mth matrix entries required to perform the N-ary multiplies of F.

2.2 HBL Inequality

Lemma 2.1. *Let* F *be a subset of* N*-ary multiplies within an MTTKRP computation,* $\phi_0(F)$ *be the projection of* F *onto the tensor* $\mathcal{T}$*, and* $\phi_m(F)$ *the projection of* F *onto the* m*-th factor matrix. Then*

$$|F| \leq |\phi_0(F)|^{1-1/N} \cdot |\phi_1(F)|^{1/N} \cdots |\phi_N(F)|^{1/N}.$$

Proof. Consider the matrix $\mathbf{\Delta}$ defined by the projections ϕ_i for $i = 0$ and $i \in [N]$,

$$\mathbf{\Delta} = \begin{pmatrix} \mathbf{1}_{N\times 1} & \mathbf{I}_{N\times N} \\ 0 & \mathbf{1}_{1\times N} \end{pmatrix}.$$

Let $\mathbf{s}^* = (1 - 1/N, 1/N, \ldots, 1/N)^T$, then $\mathbf{\Delta s}^* \geq \mathbf{1}$, and the result holds by [4, Lemma 4.1]. ∎

Note that while the inequality is true for all feasible $\mathbf{s}$, we see that $\mathbf{s}^*$ is optimal, after reordering the columns of $\mathbf{\Delta}$ and entries of $\mathbf{s}^*$, by [4, Lemma 4.2].

2.3 Quasiconvexity We use the following result in Sections 4 and 5 to prove that the constrained optimization problem central to the lower bound argument is a convex problem.

LEMMA 2.2. *The function* $g(x_1, \ldots, x_N) = L - \prod_{i\in[N]} x_i$, *for some constant* L, *is quasiconvex on the positive orthant.*

Proof. By [15, Theorem 21.14] a function g is quasiconvex on a convex set S if and only if for any $x, y \in S$, $g(x) \leq g(y)$ implies $\nabla g(y) \cdot (x - y) \leq 0$. Thus we want to show that $\prod_{i\in[N]} y_i \leq \prod_{i\in[N]} x_i$ implies $-\sum_{i\in[N]} \prod_{j\in[N]-\{i\}} y_j(x_i - y_i) \leq 0$, or equivalently, $N \prod_{i\in[N]} y_n \leq \sum_{n\in[N]} \left(x_n \prod_{i\in[N]-\{n\}} y_i\right)$. Because all entries are positive, this is also equivalent to $1 \leq \frac{1}{N}\left(\sum_{n\in[N]} \frac{x_n}{y_n}\right)$, or that the arithmetic mean of the ratios $x_1/y_1, x_2/y_2, \ldots, x_N/y_N$ is at least 1. The assumption $\prod_{n\in[N]} y_n \leq \prod_{n\in[N]} x_n$ implies that the geometric mean of these ratios is at least 1, so the result follows from the arithmetic mean–geometric mean inequality. ∎

3 MTTKRP Algorithms

3.1 CP Algorithms Multiple algorithms are used to compute CP decompositions. We will consider those that minimize the residual in terms of the tensor norm that generalizes the matrix Frobenius and vector ℓ_2 norms (this corresponds to a least-squares objective function). The simplest is known as Alternating Least Squares (CP-ALS), which uses a block coordinate descent approach. Each factor matrix corresponds to a block, which is updated while holding all other factor matrices fixed. With a least-squares CP objective function, each subproblem is a linear least squares problem that can be solved in closed form.

For example, in the 3-way case, CP-ALS repeats the following iteration:

1. solve $(\mathbf{W}^\mathsf{T}\mathbf{W} * \mathbf{V}^\mathsf{T}\mathbf{V})\mathbf{U} = \mathbf{T}_{(1)}(\mathbf{W} \odot \mathbf{V})$ for $\mathbf{U}$
2. solve $(\mathbf{W}^\mathsf{T}\mathbf{W} * \mathbf{U}^\mathsf{T}\mathbf{U})\mathbf{V} = \mathbf{T}_{(2)}(\mathbf{W} \odot \mathbf{U})$ for $\mathbf{V}$
3. solve $(\mathbf{V}^\mathsf{T}\mathbf{V} * \mathbf{U}^\mathsf{T}\mathbf{U})\mathbf{W} = \mathbf{T}_{(3)}(\mathbf{V} \odot \mathbf{U})$ for $\mathbf{W}$

The $*$ notation represents Hadamard (elementwise) multiplication. These linear systems are the normal equations for each linear least squares subproblem. Note that the right-hand-side matrices for these systems are the results of MTTKRPs in each mode. For more details and the general algorithm, see [6, 11], for example.

CP decompositions can also be computed using other optimization algorithms, such as Gauss-Newton for nonlinear least squares problems. Rather than update factor matrices individually in alternating fashion, factor matrix entries are updated all at once each iteration. Nearly all gradient-based optimization algorithms are bottlenecked by the computation of the gradient. Expressed in matrix form, the gradients with respect to each factor matrix are as follows:

1. $\mathbf{G}_U = \mathbf{T}_{(1)}(\mathbf{W} \odot \mathbf{V}) - (\mathbf{W}^\mathsf{T}\mathbf{W} * \mathbf{V}^\mathsf{T}\mathbf{V})\mathbf{U}$
2. $\mathbf{G}_V = \mathbf{T}_{(2)}(\mathbf{W} \odot \mathbf{U}) - (\mathbf{W}^\mathsf{T}\mathbf{W} * \mathbf{U}^\mathsf{T}\mathbf{U})\mathbf{V}$
3. $\mathbf{G}_W = \mathbf{T}_{(3)}(\mathbf{V} \odot \mathbf{U}) - (\mathbf{V}^\mathsf{T}\mathbf{V} * \mathbf{U}^\mathsf{T}\mathbf{U})\mathbf{W}$

Again, note the appearance of the MTTKRPs in each mode, which require the bulk of the computation within the gradient computation. For more details on gradient-based CP optimization, see [1, 14, 17], for example.

While there are multiple algorithms for computing CP, nearly all of them require computing MTTKRPs in each mode. Because the MTTKRP operation involves the data tensor, while the other computations involve only the factor matrices, it is nearly always the bottleneck. Thus, to efficiently parallelize algorithms for CP, it is necessary to use an efficient parallel algorithm for MTTKRP.

We note that there exist important optimizations that exploit the overlap among all N MTTKRPs performed in each iteration of a CP algorithm, such as the use of "dimension trees" to organize temporary values that can be re-used across MTTKRPs [8, 3]. The results of this paper focus on a single MTTKRP rather than all N, but the efficient algorithms for a single MTTKRP can be easily adapted to employ such optimizations. We discuss this issue further in Section 7.

3.2 Parallel MTTKRP Ballard, Knight, and Rouse propose the Parallel General MTTKRP Algorithm [4, Algorithm 3]. In this algorithm, the processors are organized into an $(N + 1)$-dimensional grid with dimensions $P_0 \times P_1 \times \cdots \times P_N$. The tensor is first distributed in block (Cartesian) fashion over the last N dimensions of the processor grid, so that each block has dimension approximately I_n/P_n in the nth mode. Then, each block of the tensor is distributed across the P_0 processors whose indices correspond to the block. The nth factor matrix is first distributed in block fashion (2D matrix dimension) over dimensions 0 and n, so that each block has dimension approximately

$I_n/P_n \times R/P_0$. Then, each block is distributed across $P/(P_0P_n)$ processors whose indices correspond to the block. The algorithm works by first gathering data necessary for local computation (this is organized into All-Gather communication collectives), performing the local computation (which corresponds to a local MTTKRP), and then summing the output over multiple processors (via Reduce-Scatter communication collectives).

We will refer to this as the "General Algorithm" and classify instantiations of the algorithm based on the processor grid dimensions:

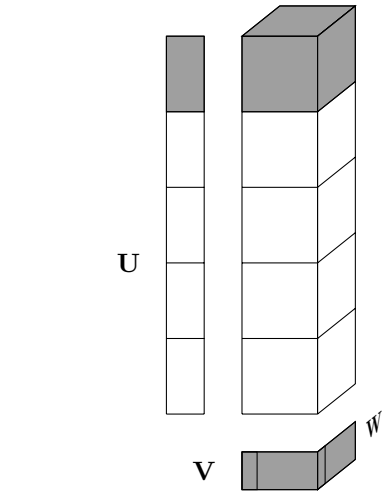
- $(N+1)$-D: $P_0 > 1$, $P_n > 1$ for all $1 \le n \le N$
- N-D: $P_0 = 1$, $P_n > 1$ for all $1 \le n \le N$
- K-D: $P_0 = 1$, $P_n = 1$ for all $1 \le n \le N-K$, and $P_n > 1$ for all $N-K < n \le N$
- 1-D: $P_0 = 1$, $P_n = 1$ for all $1 \le n \le N-1$, $P_N = P$.

This implies that a K-D algorithm has K processor grid dimensions larger than 1. We note the arbitrary order imposed on these definitions corresponds to the assumption that $I_1 \le \cdots \le I_N$.

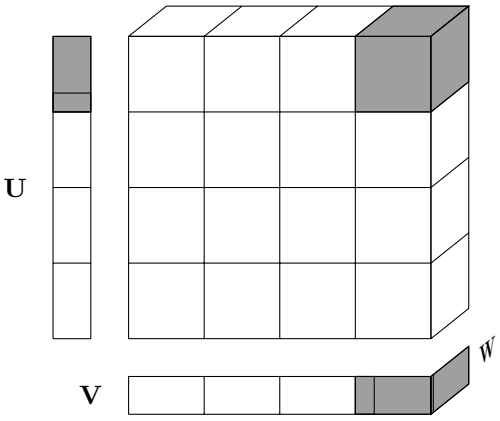
The tensor distribution corresponding the N-D algorithm is natural for parallel tensor computations and has been used before for dense [2, 3, 13] and sparse [9, 16] tensors, where it is known as "medium-grained distribution." The N-D algorithm is also referred to as the "Stationary Tensor" algorithm, because it corresponds to not computing tensor entries (because $P_0 = 1$) [4]. In the sparse case, alternatives are the "fine-grained" distribution, which is not an instantiation of the General Algorithm, and "coarse-grained" distribution, which corresponds to the 1-D algorithm.

We illustrate this classification for 3-way tensors ($N = 3$) in Figure 1. The 1-D algorithm example corresponds to a $1 \times 1 \times 1 \times 5$ processor grid. In this case, $\mathbf{U}$ is not communicated, but all of $\mathbf{V}$ and $\mathbf{W}$ must be shared across all processors. The 2-D algorithm example corresponds to a $1 \times 1 \times 4 \times 4$ processor grid. In this case, each processor must gather parts of $\mathbf{U}$ and $\mathbf{V}$ and all of $\mathbf{W}$ must be shared across all processors. The 3-D algorithm example corresponds to a $1 \times 3 \times 3 \times 3$ processors grid. Parts of all matrices must be communicated, but no full matrix needs to be shared across all processors.

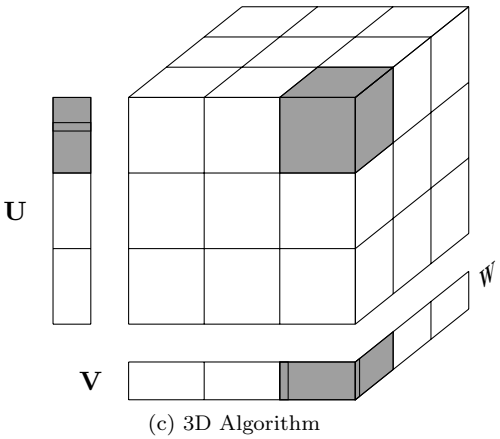
The examples given in Figure 1 show local subtensors that are perfectly cubical. Indeed, to minimize communication within the constraints of the general algorithm, the processor grid should be chosen so that local tensors are as cubical as possible. This may not be possible if the tensor dimensions vary widely and P is not sufficiently large, which is precisely the case when



(a) 1D Algorithm



(b) 2D Algorithm



(c) 3D Algorithm

Figure 1: Algorithm classification for 3-way tensors. Shading highlights data required by given processor. Submatrices within shaded regions correspond to data already owned. Two matrices are input and one is output, depending on the mode of the MTTKRP.

K-D algorithms (with $K < N$) are more communication efficient. We show in the following sections that optimizing the processor grid of the general algorithm achieves the optimal communication across all MTTKRP algorithms.

4 Example: $N = 3$ case

For clarity and to help build intuition, we prove our main result in this section for the 3-way case specifically. We prove the result for general N in Section 5.

The basic argument of the lower bound is that, given a subset of computation assigned to a processor, the structure of the MTTKRP computation requires that the processor must have access to certain data from the tensor and factor matrices. The MTTKRP structure yields an HBL-like inequality (Lemma 2.1) that relates the amount of the computation to the *product* of sizes of projections onto the tensor and each matrix (these quantities are also raised to fractional exponents in the bound). To obtain a communication lower bound, we would like to lower bound the *sum* of these projections, as this corresponds to the amount of data the processor must access (either already owning or receiving from other processors). Previous work used unconstrained optimization techniques (Lagrange multipliers) to convert the bound on the product to a bound on the sum to obtain communication bounds for MTTKRP [4].

In order obtain tighter bounds for more rectangular tensors, we apply constraints to the optimization problem based on the fact that the projection onto a matrix cannot be larger than the matrix itself. Adding these constraints yield a constrained optimization problem which can still be solved using more sophisticated techniques than Lagrange multipliers. We first present the abstract constrained optimization problem along with its solution. Although not all of the constraint functions are convex, the feasible region is convex, and the objective function has a global minimum that satisfies the Karush-Kuhn-Tucker (KKT) conditions.

Figure 2 provides a visualization of the optimization problem for $N = 2$: it shows the feasible region, contour lines of the objective function, and optimal solution for three different cases.

LEMMA 4.1. *Consider the constrained optimization problem for* $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$*:*

$$\min_{\mathbf{x}} x_1 + x_2 + x_3,$$

s.t. $L \leq x_1x_2x_3$, $0 \leq x_1 \leq K_1$, $0 \leq x_2 \leq K_2$, $0 \leq x_3 \leq K_3$ *for some positive constants* L *and* $K_1 \leq K_2 \leq K_3$. *The global minimum* $\mathbf{x}^*$ *depends on the relative sizes of the constants, yielding five cases.*

1. *If* $L^{1/3} < K_1 \leq K_2 \leq K_3$, *then* $\mathbf{x}^* = \begin{bmatrix} L^{1/3} & L^{1/3} & L^{1/3} \end{bmatrix}$ *with objective function value* $3L^{1/3}$.
2. *If* $K_1 \leq L^{1/3}$ *and* $(L/K_1)^{1/2} < K_2 \leq K_3$, *then* $\mathbf{x}^* = \begin{bmatrix} K_1 & (L/K_1)^{1/2} & (L/K_1)^{1/2} \end{bmatrix}$ *with objective function value* $K_1 + 2(L/K_1)^{1/2}$.
3. *If* $K_1 \leq K_2 \leq (L/K_1)^{1/2}$ *and* $L/(K_1K_2) < K_3$, *then* $\mathbf{x}^* = \begin{bmatrix} K_1 & K_2 & (L/K_1K_2) \end{bmatrix}$ *with objective function value* $K_1 + K_2 + L/(K_1K_2)$.
4. *If* $K_1 \leq K_2 \leq K_3 = L/(K_1K_2)$, *then* $\mathbf{x}^* = \begin{bmatrix} K_1 & K_2 & K_3 \end{bmatrix}$ *with objective function value* $K_1 + K_2 + K_3$.
5. *If* $K_1 \leq K_2 \leq K_3 < L/(K_1K_2)$, *then the optimization problem is infeasible.*

Proof. In order to use the KKT conditions to find a global minimum, we must verify that the feasible region is convex, Slater's condition holds, and that the constraint functions are nondegenerate on the feasible region [12, Theorem 2.3]. We note that while the first constraint function is not convex, it is quasiconvex by Lemma 2.2. A quasiconvex function has convex lower level sets, so a quasiconvex constraint function yields a convex feasible region. All other functions are affine and therefore convex, and the intersection of convex regions is convex.

Slater's condition, that the feasible region has an interior point, and the nondegeneracy condition are satisfied as long as $L < K_1K_2K_3$. (See the proof of Lemma 5.1 for a rigorous argument.)

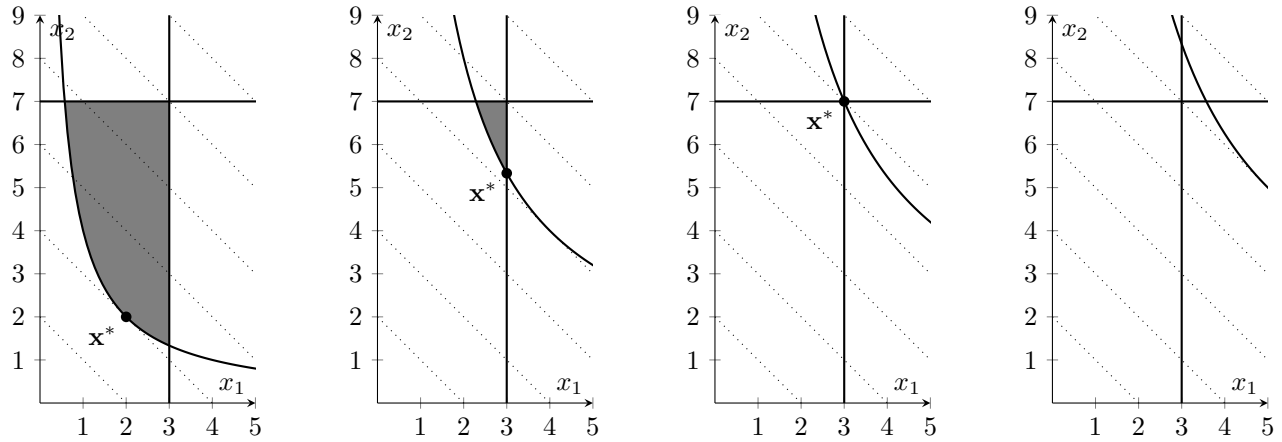
To state the KKT conditions, we first convert the constrained optimization problem to standard notation. The objective function is $f(\mathbf{x}) = x_1 + x_2 + x_3$, and the constraint functions are defined as $g_0(\mathbf{x}) = L - x_1x_2x_3$ and $g_i(\mathbf{x}) = x_i - K_i$ for $i = 1, 2, 3$, so that the constraints can be written as $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$. (We do not include the nonnegativity constraints as they are never active.) The gradients are as follows: $\nabla f = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$, $\nabla g_0 = \begin{bmatrix} -x_2x_3 & -x_1x_3 & -x_1x_2 \end{bmatrix}$, $\nabla g_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$, $\nabla g_2 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$, $\nabla g_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$.

For dual variables $\boldsymbol{\mu}$, the stationarity condition is

$$-\nabla f(\mathbf{x}^*) = \sum_{i=0}^{3} \mu_i \nabla g_i(\mathbf{x}^*),$$

the feasibility conditions are $\boldsymbol{\mu} \geq \mathbf{0}$ and $\mathbf{g}(\mathbf{x}^*) \leq \mathbf{0}$, and the complementary slackness condition is $\mu_i g_i(\mathbf{x}^*) = 0$ for $i = 0, 1, 2, 3$. We now prove each case separately.

(Case 1) Suppose $L^{1/3} < K_1 \leq K_2 \leq K_3$. Then for $x_1^* = x_2^* = x_3^* = L^{1/3}$, the only active constraint



(a) $L = 4$ (b) $L = 16$ (c) $L = 21$ (d) $L = 25$

Figure 2: Three cases of 2D constrained optimization problem $\min_{\mathbf{x}} x_1+x_2$, s.t. $L \leq x_1x_2$, $0 \leq x_1 \leq 3$, $0 \leq x_2 \leq 7$. The solid lines represent constraints, the feasible region is shaded, the dotted lines are contours of the objective function, and the optimal solution $\mathbf{x}^*$ is highlighted. In (a), the optimal solution is achieved with one active constraint, in (b) there are two active constraints, in (c) all constraints are active, and in (d) there is no solution.

is g_0, and the stationarity condition is satisfied by $\mu_0 = L^{-2/3}$, which is positive. Note that $\mathbf{x}^*$ is feasible by the assumptions of this case.

(Case 2) Suppose $K_1 \leq L^{1/3}$ and $(L/K_1)^{1/2} < K_2 \leq K_3$. Then for $x_1 = K_1$ and $x_2 = x_3 = (L/K_1)^{1/2}$, the active constraints are g_0 and g_1. The stationarity condition is satisfied by $\mu_0 = (LK_1)^{-1/2}$ and $\mu_1 = (L/K_1^3)^{1/2} - 1$. Note that $\mu_1 \geq 0$ because $K_1 \leq L^{1/3}$, and x_2^* and x_3^* are also feasible, by the assumption of this case.

(Case 3) Suppose $K_1 \leq K_2 \leq (L/K_1)^{1/2}$ and $L/(K_1K_2) < K_3$. Then for $x_1^* = K_1$, $x_2^* = K_2$, and $x_3^* = L/(K_1K_2)$, the active constraints are g_0, g_1, and g_2. The stationarity condition is satisfied by $\mu_0 = 1/(K_1K_2)$, $\mu_1 = L/(K_1^2K_2) - 1$, and $\mu_2 = L/(K_1K_2^2) - 1$. Note that μ_1 and μ_2 are both nonnegative because $K_1^2K_2 \leq K_1K_2^2 \leq K_1((L/K_1)^{1/2})^2 = L$, and x_3^* is also feasible, by the assumptions of this case.

(Case 4) Suppose $K_1 \leq K_2 \leq K_3 = L/(K_1K_2)$. Then $x_1^* = K_1$, $x_2^* = K_2$, and $x_3^* = K_3$ is the only feasible point and is therefore optimal.

(Case 5) Suppose $K_1 \leq K_2 \leq K_3 < L/(K_1K_2)$. Then by constraints g_1, g_2, and g_3, $x_1x_2x_3 < K_1K_2K_3 < K_1K_2(L/(K_1K_2)) = L$, which contradicts constraint g_0, and therefore the problem is infeasible. ∎

Given the result for the abstract convex optimization problem, we now state and prove the general communication lower bound for $N = 3$. The three cases of the result correspond to when the 3-D, 2-D, and 1-D algorithms are the most communication efficient. For example, the first case (3-D) reproduces the bound from previous work [4, Theorem 4.3], but the other two cases (2-D and 1-D) provide tighter bounds when the tensor dimensions and number of processors satisfy the corresponding conditions. We also note that a tighter bound exists when the number of processors is very large [4, Theorem 4.2], which can be attained by the 4-D algorithm.

THEOREM 4.1. *Any parallel MTTKRP algorithm involving a 3D tensor with dimensions $I_1 \leq I_2 \leq I_3$ that evenly distributes one copy of the input and output communicates either $((3/2)^{1/2}-1)I/P$ words of tensor data or at least the following amount of factor matrix data:*

$$\left(\frac{2}{3}\right)^{1/3} 3\left(\frac{I_1I_2I_3}{P}\right)^{1/3} R - \sum_k I_kR/P$$

if $I_1 \geq \left(\frac{2I_2I_3}{3P}\right)^{1/2}$,

$$I_1R + \left(\frac{2}{3}\right)^{1/2} 2\left(\frac{I_2I_3}{P}\right)^{1/2} R - \sum_k I_kR/P$$

if $I_1 < \left(\frac{2I_2I_3}{3P}\right)^{1/2}$ *and* $I_2 \geq \frac{2}{3}\frac{I_3}{P}$,

$$I_1R + I_2R + \frac{2}{3}I_3R/P - \sum_k I_kR/P$$

if $I_1 < \left(\frac{2I_2I_3}{3P}\right)^{1/2}$ *and* $I_2 < \frac{2}{3}\frac{I_3}{P}$.

Proof. Some processor must perform at least IR/P N-ary multiplies. Let F be the set of N-ary multiplies performed by this processor, so that $|F| \geq IR/P$. We let $\phi_0(F)$ be the projection of F onto the entries of the tensor required to complete the computation, and we let $\phi_i(F)$ be the projection of F onto the required ith factor matrix entries for $i = 1, 2, 3$. By [4, Lemma 4.1], with $\mathbf{s} = \begin{bmatrix} 2/3 & 1/3 & 1/3 & 1/3 \end{bmatrix}$, we have

(4.1)
$$\frac{IR}{P} \leq |\phi_0(F)|^{2/3} |\phi_1(F)|^{1/3} |\phi_2(F)|^{1/3} |\phi_3(F)|^{1/3}.$$

We consider two cases based on the size of the projection onto the tensor entries ϕ_0. If $|\phi_0(F)| > (3/2)^{1/2} I/P$, then by the assumption on load balanced input, the processor starts with only I/P tensor entries and must receive $((3/2)^{1/2} - 1)I/P$ tensor entries. If $|\phi_0(F)| \leq (3/2)^{1/2} I/P$, then eq. (4.1) simplifies to

$$|\phi_1(F)| |\phi_2(F)| |\phi_3(F)| \geq \frac{2}{3} \frac{I}{P} R^3. \tag{4.2}$$

Given this constraint on the product, we wish to lower bound the sum of the sizes of projections onto the factor matrices. However, we also observe that no factor matrix projection can exceed the number of entries in that matrix, or $|\phi_i(F)| \leq I_i R$ for $i = 1, 2, 3$. Given these constraints, along with eq. (4.2), we obtain a lower bound on $|\phi_1(F)| + |\phi_2(F)| + |\phi_3(F)|$ by applying Lemma 4.1. Because the processor can start and end with its fair share of the factor matrices, to obtain a communication lower bound we subtract from the objective function value the quantity $\sum_{k=1}^{3} I_k R/P$. Since either of the two cases may apply, the applicable lower bound is the minimum of the two expressions, and the result follows. ∎

5 General MTTKRP Lower Bound

We prove our main results in this section, following the proof structure of the $N = 3$ example in Section 4. We first state the solution to the convex optimization problem in N variables.

LEMMA 5.1. *Consider the constrained optimization problem for* $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix}$*:*

$$\min_{\mathbf{x}} \sum_{n \in [N]} x_n$$

s.t. $L \leq \prod_{n \in [N]} x_n$, $0 \leq x_n \leq K_n$ *for all* $n \in [N]$ *for some positive constants* L *and* $K_1 \leq K_2 \leq \cdots \leq K_N$*. We consider three cases.*

1. *If* $L > \prod_{i \in [N]} K_i$*, then the optimization problem is infeasible.*
2. *If* $L = \prod_{i \in [N]} K_i$*, the minimum value of the objective function,* $\sum_{i \in [N]} K_i$*, is achieved at the only feasible point* $\mathbf{x}^* = \begin{bmatrix} K_1 & \cdots & K_N \end{bmatrix}$*.*
3. *If* $L < \prod_{i \in [N]} K_i$ *then the minimum value of the objective function is*

$$\sum_{j \in [J]} K_j + (N - J) \left(L / Q_J\right)^{1/(N-J)},$$

where $0 \leq J < N$ *is defined such that*

$$\begin{aligned} K_j &\leq \left(L / Q_{j-1}\right)^{1/(N-j+1)} \quad \text{for } 0 \leq j \leq J \\ K_k &> \left(L / Q_{k-1}\right)^{1/(N-k+1)} \quad \text{for } J < k \leq N, \end{aligned} \tag{5.3}$$

and $Q_n = \prod_{i \in [n]} K_i$ *(with* $Q_0 = 1$*). The minimum is achieved at the point* $\mathbf{x}^*$ *defined by* $x_j = K_j$ *for* $j \in [J]$*, and* $x_k = \left(L / Q_J\right)^{1/(N-J)}$ *for* $k > J$*.*

Proof. Let $f(\mathbf{x}) = \sum_{i \in [N]} x_i$, $g_0(\mathbf{x}) = L - \prod_{i \in [N]} x_i$, and $g_n(\mathbf{x}) = x_n - K_n$. (We do not include the nonnegativity constraints as they are never active.) We prove the cases separately:

(Case 1) Suppose $L > \prod_{i \in [N]} K_i$, and let $\mathbf{x}$ satisfy constraints g_n for all $n \in [N]$. Then $\prod_{i \in [N]} x_i \leq \prod_{i \in [N]} K_i < L$, which contradicts constraint g_0. Therefore the problem is infeasible.

(Case 2) Suppose $L = \prod_{i \in [N]} K_i$. Clearly $\mathbf{x}^* = \begin{bmatrix} K_1 \cdots K_N \end{bmatrix}$ is feasible. Consider any point $\mathbf{x} \neq \mathbf{x}^*$ that satisfies g_n for all $n \in [N]$, then there exists i such that $x_i < K_i$ and thus $\prod_{i \in [N]} x_i < L$ so $\mathbf{x}$ is not feasible. Hence the only feasible point is $\mathbf{x}^*$ implying that the global minimum occurs at $\mathbf{x}^*$ and is $\sum_{i \in [N]} K_i$.

(Case 3) Suppose that $L < \prod_{i \in [N]} K_i$. We use sufficiency of the KKT conditions for convex optimization problems to identify the global minimum. Although g_0 is not a convex function, because it is quasiconvex by Lemma 2.2 and $\{g_n\}$ are all affine, the feasible region is convex. In order to use the KKT conditions to find a global minimum, we must also verify that Slater's condition holds and demonstrate the nondegeneracy condition [12, Theorem 2.3]

We begin by showing that Slater's condition holds: we will show that there exists a region of points near the point $\begin{bmatrix} K_1 \cdots K_N \end{bmatrix}$. Let $d = \prod_{i \in [N]} K_i - L$, and let $\delta = d / \left(2^N \prod_{i \in [N]} K_i\right)$. All the points in the box with side dimension δ and largest corner at $\begin{bmatrix} K_1 \cdots K_N \end{bmatrix}$ are feasible because the nth coordinate is less than K_n (and greater than $K_n - \delta$), and $\prod_{i \in [N]} (K_i - \delta_i) \geq \prod_{i \in [N]} (K_i - \delta) \geq \prod_{i \in [N]} K_i - 2^N \delta \prod_{i \in [N]} K_i = \prod_{i \in [N]} K_i - d = L$.

We now demonstrate that g_n are nondegenerate for $n = 0$ and $n \in [N]$ [12, Assumption 2.1]. Note

that for $n \in [N]$, nondegeneracy is implied by Slater's condition as g_n are affine. To see that g_0 also satisifies the nondegeneracy assumption, we note that $g_0(\mathbf{x}) = 0$ implies $x_i \neq 0$ for all $i \in [N]$, while $\nabla g_0(\mathbf{x}) = 0$ implies that there exist $i, k \in [N], i \neq k$ such that $x_i = 0 = x_k$. Thus, for all $\mathbf{x}$ such that $g_0(\mathbf{x}) = 0$ we have that $\nabla g_0(\mathbf{x}) \neq 0$.

To state the KKT conditions, we consider the gradients:

$$\nabla f = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}$$
$$\nabla g_0 = \begin{bmatrix} -\prod_{i\in[N]-\{1\}} x_i & \cdots & -\prod_{i\in[N-1]} x_i \end{bmatrix}$$
$$\nabla g_n = e_n, \qquad n \in [N].$$

For dual variables $\boldsymbol{\mu}$, the stationarity condition is $-\nabla f(\mathbf{x}^*) = \sum_{n\in[N]} \mu_n \nabla g_n(\mathbf{x}^*)$, the feasibility conditions are $\boldsymbol{\mu} \geq \mathbf{0}$ and $\mathbf{g}(\mathbf{x}^*) \leq \mathbf{0}$, and the complementary slackness condition is $\mu_n g_n(\mathbf{x}^*) = 0$ for all n.

We now show that there exists a J that satisfies eq. (5.3). Let J be one less than the smallest k such that $K_k > (L/Q_{k-1})^{1/(N-k+1)}$. Note that there is at least one such k as $L < \prod_{i\in[N]} K_i = Q_{N-1}K_N$ implies that $k = N$ has this property. By the definition of J, we have that $K_j \leq (L/Q_{j-1})^{1/(N-j+1)}$ for all $j \in [J]$. We prove that $K_k > (L/Q_{k-1})^{1/(N-k+1)}$ for all $k > J$ by induction. Our base case, $k = J+1$, holds by the definition of J. Assume that $K_k > (L/Q_{k-1})^{1/(N-k+1)}$, then $K_{k+1} \geq K_K > (L/Q_{k-1})^{1/(N-k+1)} > (L/Q_k)^{1/(N-k)}$. The first inequality holds by our assumptions about the order of the K_i, the second inequality is the induction hypothesis, and the third inequality follows from the second inequality and the definition of Q_k and Q_{k+1}. Thus by induction on k, we can see that $K_k > (L/Q_{k-1})^{1/(N-k+1)}$ for all $k > J$.

Now we identify the global maximizer. Let $\mathbf{x}^*$ be defined such that $x_n^* = K_n$ if $n \in [J]$, $x_{J+1}^* = \cdots = x_N^* = (L/Q_J)^{1/(N-J)}$. Then the active constraints are $g_0, g_1, \ldots, g_J$. The stationarity condition is satisfied by $\mu_0 = (L/Q_J)^{\frac{1}{N-J}}/L$ which is positive, and $\mu_j = (1/K_j)\,(L/Q_J)^{\frac{1}{N-J}} - 1$ for $j \in [J]$. To see that μ_j is positive recall the assumptions that $K_j \leq K_J$ for all $j \in J$ and that $K_J \leq (L/Q_{J-1})^{1/(N-J+1)}$. Then

$$\begin{aligned}
\mu_j &= \frac{1}{K_j}\left(\frac{L}{Q_J}\right)^{1/(N-J)} - 1 \\
&> \frac{1}{K_J^{\frac{N-J+1}{N-J}}}\left(\frac{L}{Q_{J-1}}\right)^{1/(N-J)} - 1 \\
&\geq \frac{(L/Q_{J-1})^{1/(N-J)}}{(L/Q_{J-1})^{1/(N-J)}} - 1 = 0.
\end{aligned}$$

Thus $\mathbf{x}^*$ satisfies the KKT conditions and hence is a global minimum. ∎

Given the solution to the convex optimization problem, we now apply it to obtain our main result. In short, the result states that if an algorithm does not communicate (much of) the tensor, then it must communicate significant amounts of the factor matrices. Depending on the relative sizes of the tensor dimensions and P, an algorithm must communicate *all* of the smallest factor matrices and *part* of the largest factor matrices, where the distinction between small and large matrices depends on the number of processors.

THEOREM 5.1. *Any parallel MTTKRP algorithm involving an N dimensional tensor with dimensions $I_1 \leq \cdots \leq I_N$ that evenly distributes one copy of the input and output communicates either $\left((3/2)^{\frac{1}{N-1}} - 1\right) I/P$, where $I = \prod_{i\in[N]} I_i$, words of tensor data or at least the following amount of factor matrix data:*

$$\sum_{j\in[J]} I_j R + \frac{2}{3}(N-J)R\left(\frac{\prod_{k=J+1}^{N} I_k}{P}\right)^{\frac{1}{(N-J)}} - \sum_{i\in[N]} \frac{I_i R}{P}$$

where J is the value that satisfies

(5.4)

$$I_j \leq \left(\frac{2}{3}\right)^{\frac{1}{N-j}}\left(\frac{\prod_{i=j+1}^{N} I_i}{P}\right)^{\frac{1}{N-j}} \text{ for all } 0 \leq j \leq J$$

$$I_k > \left(\frac{2}{3}\right)^{\frac{1}{N-k}}\left(\frac{\prod_{i=k+1}^{N} I_i}{P}\right)^{\frac{1}{N-k}} \text{ for all } J < k \leq N.$$

Proof. Some processor must perform at least IR/P N-ary multiplies. Let F be the set of N-ary multiplies performed by this processor, so that $|F| \geq IR/P$. We let $\phi_0(F)$ be the projection of F onto the entries of the tensor required to complete the computation, and we let $\phi_i(F)$ be the projection of F onto the required ith factor matrix entries for $i \in [N]$. By [4, Lemma 4.1], with $\mathbf{s} = \begin{bmatrix}(N-1)/N & 1/N & \cdots & 1/N\end{bmatrix}$, we have

$$\frac{IR}{P} \leq |\phi_0(F)|^{(N-1)/N} \prod_{i\in[N]} |\phi_i(F)|^{1/N}. \tag{5.5}$$

We consider two cases based on the size of the projection onto the tensor entries ϕ_0. If $|\phi_0(F)| > (3/2)^{\frac{1}{N-1}} I/P$, then by the assumption on load balanced input, the processor starts with only I/P tensor entries and must receive $\left((3/2)^{\frac{1}{N-1}} - 1\right) I/P$ tensor entries. If $|\phi_0(F)| \leq (3/2)^{\frac{1}{N-1}} I/P$, then eq. (5.5) simplifies to

$$\prod_{i\in[N]} |\phi_i(F)| \geq \frac{2}{3}\frac{IR}{P}. \tag{5.6}$$

Given this constraint on the product, we wish to lower bound the sum of the sizes of projections onto the factor matrices. However, we also observe that no factor matrix projection can exceed the number of entries in that matrix, or $|\phi_i(F)| \leq I_i R$ for $i \in [N]$. Given these constraints, along with eq. (5.6), we obtain the lower bound

$$\sum_{i\in[N]} |\phi_i(F)| \geq \sum_{j\in[J]} I_j R + \frac{2}{3}(N-J)R\left(\frac{\prod_{k=J+1}^{N} I_k}{P}\right)^{\frac{1}{(N-J)}}$$

by applying Lemma 5.1. Note that we used $(2/3)^{1/(N-J)} \geq 2/3$ to simplify the constant factor.

Because the processor can start and end with its fair share of the factor matrices, to obtain a communication lower bound we subtract from the objective function value the quantity $\sum_{i\in[N]} I_i R/P$. Since either of the two cases may apply, the applicable lower bound is the minimum of the two expressions, and the result follows. ∎

COROLLARY 5.1. *Assume without loss of generality that $I_1 \leq \cdots \leq I_N$. If the J that satisfies eq. (5.4) is smaller than $N-1$, then any parallel MTTKRP algorithm where each processor initially and finally owns at most $I_i R/P$ entries of factor matrix i for all $i \in [N]$, I/P entries of the tensor, some processor performs at least*

$$\Omega\left(\min\left\{\sum_{j\in[J]} I_j R + (N-J)R\left(\frac{\prod_{k=J+1}^{N} I_k}{P}\right)^{\frac{1}{N-J}}, \left(\left(\frac{3}{2}\right)^{\frac{1}{N-J}} - 1\right)\frac{I}{P}\right\}\right)$$

sends and receives

Proof. This follows from Theorem 5.1 by showing that the subtractive terms are dominated by the positive terms when $J < N-1$. We note that the smallest J subtractive terms are clearly dominated by the initial sum of the sizes of the J smallest factor matrices. To see that the remaining $N-J$ terms are dominated by the final term, we need only see that the largest $I_N R/P$ is dominated by $R\left(\prod_{k=J+1}^{N} I_k/P\right)^{\frac{1}{N-J}}$. This follows by a similar induction argument used in the proof of the existence of J in Lemma 5.1. ∎

Note that Corollary 5.1 is not applicable to the one dimensional case, when $J = N-1$ as $(2/3)I_N R/P < I_N R/P$. This implies that the largest factor matrix need not be communicated in this case (indeed, 1D algorithms avoid communicating the largest factor matrix). However, this negative term may also dominate the projections onto the smaller factor matrices. We present an alternative proof of the one dimensional lower bound ($J = N-1$ case).

THEOREM 5.2. *Assume without loss of generality that $I_1 \leq \cdots \leq I_N$. In any parallel MTTKRP algorithm where each processor initially and finally owns at most $I_i R/P$ entries of factor matrix i for all $i \in [N]$ and at most I/P tensor entries, some processor performs at least*

$$\Omega\left(\min\left\{\sum_{j\in[N-1]} I_j R, \frac{I_N R}{P}\right\}\right)$$

sends and receives, assuming that $I_N R < 2\left(\left(\frac{3}{2}\right)^{1/(N-1)} - 1\right) I$.

Proof. Some processor must perform at least IR/P N-ary multiplies, and we let F be the set of the N-ary multiplies it performs. Let $\phi_0(F)$ be the projection corresponding to the tensor elements the processor must access to complete the computation, and let $\phi_i(F)$ be the projection onto the ith factor matrix elements the processor must access, for $i \in [N]$. By Lemma 2.1,

$$\frac{IR}{P} \leq |F| \leq |\phi_0(F)|^{1-1/N} \cdot |\phi_1(F)|^{1/N} \cdots |\phi_N(F)|^{1/N}. \tag{5.7}$$

We consider three cases based on the sizes of the projection onto the tensor and the largest factor matrix, $\phi_0(F)$ and $\phi_N(F)$. Suppose $|\phi_0(F)| > (3/2)^{1/(N-1)} I/P$. Then by assumption, the processor starts out with only I/P tensor elements, so it must receive at least $\left((3/2)^{1/(N-1)} - 1\right) I/P$ tensor elements. Suppose $|\phi_N(F)| > 3I_N R/2P$. Then by assumption, the processor starts out and ends with only $I_N R/P$ matrix entries, so it must communicate at least $I_N R/2P$ matrix entries (or unreduced sums).

If neither of those cases apply, then $|\phi_0(F)| \leq (3/2)^{1/(N-1)} I/P$ and $|\phi_N(F)| \leq 3I_N R/2P$. By eq. (5.7), this implies $\frac{4}{9}\frac{IR^{N-1}}{I_N} \leq |\phi_1(F)| \cdots |\phi_{N-1}(F)|$. Furthermore, we have $|\phi_j(F)| \leq I_j R$ for all $j \in [N-1]$, so for each $j \in [N-1]$ we have

$$\frac{4}{9}\frac{IR^{N-1}}{I_N} \leq I_1 R \cdots I_{j-1} R \cdot |\phi_j(F)| \cdot I_{j+1} R \cdots I_{N-1} R,$$

or $(4/9) I_j R \leq |\phi_j(F)|$. By assumption, the processor has access to only $I_j R/P$ at the start and end of the algorithm, so it must communicate $(4/9) I_j R - I_j R/P$ entries of the jth factor matrix, for each $j \in [N-1]$. Thus, the processor must send or receive at least $(4/9)\sum_{j\in[N-1]} I_j R - \sum_{j\in[N-1]} I_j R/P$ words of data.

Because any of the three cases is possible, the result is the minimum of the three bounds. Assuming

$I_N R < 2\left(\left(\frac{3}{2}\right)^{1/(N-1)} - 1\right) I$, which is expected or else the largest factor matrix would be nearly as large as the tensor, the bound involving the tensor entries is never the minimum, yielding a minimum of the two other bounds. ∎

6 Optimality

In order to know which dimension algorithm to use, one can recover the relevant ranges of P from the definition of J in Theorem 5.1. Specifically, the K-D algorithm with $K = N - J$ should be applied when $\frac{\prod_{k=J+2}^{N} I_k}{I_{J+1}^{N-J-1}} < P \leq \frac{\prod_{k=J+1}^{N} I_k}{I_J^{N-J}}$.

THEOREM 6.1. *Let J be as defined in Theorem 5.1, and assume that $\left(\prod_{k=J+1}^{N} I_k/P\right)^{1/(N-J)}$ divides I_k for all $J < k \leq N$. Under these assumptions, and the assumption that the tensor is not communicated, the General Parallel MTTKRP algorithm [4, Algorithm 3] is communication optimal with the correct processor grid.*

Proof. We specify the processor grid as $P_0 = 1, P_j = 1$ for all $j \in [J]$, and for all $i > J$, $P_i = I_i P^{1/(N-J)} / \left(\prod_{k=J+1}^{N} I_k\right)^{1/(N-J)}$. Note that $P_i > 1$ because $I_i > \left(\prod_{k=i+1}^{N} I_k/P\right)^{1/(N-i)} > \left(\prod_{k=J+1}^{N} I_k/P\right)^{1/(N-J)}$.

Filling in the appropriate values to [4, Equation 11] and simplifying the terms involving processor dimensions, we obtain

$$\sum_{j\in[J]} (P-1)\frac{I_j R}{P} + \sum_{i=J+1}^{N} \left(\frac{P}{I_i}\left(\frac{\prod_{k=J+1}^{N} I_k}{P}\right)^{\frac{1}{N-J}} - 1\right)\frac{I_i R}{P}$$

which simplifies to

$$\sum_{j\in[J]} I_j R + (N-J)R\left(\frac{\prod_{k=J+1}^{N} I_k}{P}\right)^{\frac{1}{N-J}} - \sum_{i\in[N]} \frac{I_i R}{P}$$

matching Corollary 5.1 (for $J < N-1$) or Theorem 5.2 (for $J = N-1$) to within constant factors. ∎

7 Conclusion

The main result of this paper is the tightening of the parallel communication bound for MTTKRP in the cases that K-D algorithm (for $K < N$) is preferred to an N-D algorithm. While 1-D and 2-D algorithms have been used for 3-way and higher dimensional tensors before [13, 3], they did not attain the previously established lower bounds [4]. The new lower bounds prove that algorithms based on block (Cartesian) distributions of the tensor that avoid communicating the tensor are communication optimal even when P is so small that some of the tensor dimensions are not split across processors.

The convex optimization approach to establishing the lower bound is a novel proof technique that can be applied more broadly. In particular, the use of HBL-like inequalities proposed in [7] implicitly assumes that the number of loops that define the computation is constant and that the loop bounds are sufficiently large. Our approach uses optimization to convert the HBL-like inequality, which is based on a product of projections onto data, to a sum of the projections in order to obtain the quantity related to communication cost. Furthermore, we use the loop bounds to constrain the optimization problem, excluding sizes of projections that are infeasible for algorithms, and establishing a tighter bound over the space of feasible projections and algorithms.

We note that the bounds here apply to a single MTTKRP computation, while algorithms for CP require computing an MTTKRP for each mode of the tensor every iteration. In addition, the N MTTKRPs across modes share both data and computation. For example, the MTTKRPs in the first two modes both involve the tensor and the 3rd through Nth factor matrices. This implies that those matrices need not be communicated twice and that some temporary computation can be saved. Communicating each factor matrix only once per iteration (which involves all N MTTKRPs) means that the communication for all N MTTKRPs is only a constant factor more than performing a single MTTKRP [9, 3]. Avoiding the computation can be done by organizing the temporary values into a "dimension tree" and saves a factor of $O(N)$ flops [14, 8, 3].

Optimizations over multiple MTTKRPs are therefore crucial to efficient algorithms for CP, and the lower bounds presented here do not apply to these optimizations. This is because saving factor matrix data across MTTKRPs violates the assumption that one copy of the inputs are evenly distributed across processors, and dimension tree optimizations break the atomicity assumption in the definition of MTTKRP (Definition 2.1). However, the optimal algorithm for a single MTTKRP is easily adapted to the N MTTKRP case (dimension trees can be employed locally to reduce computation). Future work will apply this lower bound technique to entire iterations of CP decomposition algorithms, taking into account the overlap across MTTKRPs.

References

[1] E. Acar, D. M. Dunlavy, and T. G. Kolda, *A scalable optimization approach for fitting canonical tensor decompositions*, Journal of Chemometrics, 25 (2011), pp. 67–86, https://doi.org/10.1002/cem.1335, http://dx.doi.org/10.1002/cem.1335.

[2] W. Austin, G. Ballard, and T. G. Kolda, *Parallel tensor compression for large-scale scientific data*, in Proceedings of the 30th IEEE International Parallel and Distributed Processing Symposium, May 2016, pp. 912–922, https://doi.org/10.1109/IPDPS.2016.67, https://www.computer.org/csdl/proceedings/ipdps/2016/2140/00/2140a912-abs.html.

[3] G. Ballard, K. Hayashi, and R. Kannan, *Parallel nonnegative CP decomposition of dense tensors*, in 25th IEEE International Conference on High Performance Computing (HiPC), Dec 2018, pp. 22–31, https://doi.org/10.1109/HiPC.2018.00012.

[4] G. Ballard, N. Knight, and K. Rouse, *Communication lower bounds for matricized tensor times Khatri-Rao product*, in Proceedings of the 32nd IEEE International Parallel and Distributed Processing Symposium, May 2018, pp. 557–567, https://doi.org/10.1109/IPDPS.2018.00065, https://ieeexplore.ieee.org/document/8425209/.

[5] J. Bennett, A. Carbery, M. Christ, and T. Tao, *Finite bounds for Hölder-Brascamp-Lieb multilinear inequalities*, Mathematical Research Letters, 17 (2010), pp. 647–666.

[6] R. Bro, *PARAFAC. tutorial and applications*, Chemometrics and Intelligent Laboratory Systems, 38 (1997), pp. 149 – 171, https://doi.org/https://doi.org/10.1016/S0169-7439(97)00032-4, http://www.sciencedirect.com/science/article/pii/S0169743997000324.

[7] M. Christ, J. Demmel, N. Knight, T. Scanlon, and K. Yelick, *Communication lower bounds and optimal algorithms for programs that reference arrays - part 1*, Tech. Report UCB/EECS-2013-61, EECS Department, University of California, Berkeley, May 2013, http://www.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-61.html.

[8] O. Kaya and B. Uçar, *Parallel candecomp/parafac decomposition of sparse tensors using dimension trees*, SIAM Journal on Scientific Computing, 40 (2018), pp. C99–C130, https://doi.org/10.1137/16M1102744, https://doi.org/10.1137/16M1102744.

[9] O. Kaya and B. Uçar, *Scalable sparse tensor decompositions in distributed memory systems*, in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '15, New York, NY, USA, 2015, ACM, pp. 77:1–77:11, https://doi.org/10.1145/2807591.2807624, http://doi.acm.org/10.1145/2807591.2807624.

[10] N. Knight, *Communication-Optimal Loop Nests*, PhD thesis, EECS Department, University of California, Berkeley, Aug 2015, http://www2.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-185.html.

[11] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Review, 51 (2009), pp. 455–500, https://doi.org/10.1137/07070111X, http://epubs.siam.org/doi/abs/10.1137/07070111X.

[12] J. B. Lasserre, *On representations of the feasible set in convex optimization*, Optimization Letters, 4 (2010), pp. 1–5, https://doi.org/10.1007/s11590-009-0153-6.

[13] A. P. Liavas, G. Kostoulas, G. Lourakis, K. Huang, and N. D. Sidiropoulos, *Nesterov-based alternating optimization for nonnegative tensor factorization: Algorithm and parallel implementation*, IEEE Transactions on Signal Processing, (2017), https://doi.org/10.1109/TSP.2017.2777399, http://ieeexplore.ieee.org/document/8119874/.

[14] A.-H. Phan, P. Tichavsky, and A. Cichocki, *Fast alternating LS algorithms for high order CANDECOMP/PARAFAC tensor factorizations*, IEEE Transactions on Signal Processing, 61 (2013), pp. 4834–4846, https://doi.org/10.1109/TSP.2013.2269903.

[15] C. Simon and L. Blume, *Mathematics for Economists*, vol. 7, Norton, New York, 1994.

[16] S. Smith and G. Karypis, *A medium-grained algorithm for distributed sparse tensor factorization*, in IEEE 30th International Parallel and Distributed Processing Symposium, May 2016, pp. 902–911, https://doi.org/10.1109/IPDPS.2016.113.

[17] N. Vervliet and L. D. Lathauwer, *Numerical optimization-based algorithms for data fusion*, in Data Fusion Methodology and Applications, M. Cocchi, ed., vol. 31 of Data Handling in Science and Technology, Elsevier, 2019, pp. 81 – 128, https://doi.org/https://doi.org/10.1016/B978-0-444-63984-4.00004-1, http://www.sciencedirect.com/science/article/pii/B9780444639844000041.