Communication Optimal Algorithms for Matrix and Tensor Computations

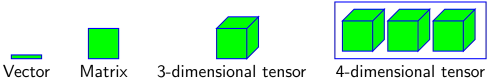
Suraj Kumar

ROMA team
Inria & ENS Lyon

Project committee meeting
Jan 17, 2025

Tensors and their uses

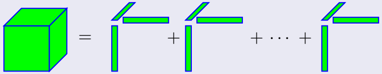
- **Neuroscience**: Neuron $\times$ Time $\times$ Trial
- **Media**: User x Movie x Time
- **Ecommerce**: User x Product x Time
- **Social-Network**: Person x Person x Time x Type



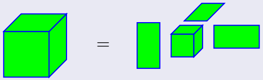
- High dimensional tensors: Neural network, Molecular simulation, Quantum computing

- People work with low dimensional structure (decomposition) of tensors

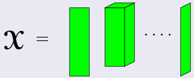
Canonical decomposition



Tucker decomposition



Tensor-train decomposition

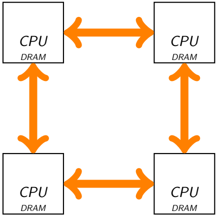


Importance of communication in high performance computing

- Gaps between computation and communication costs growing exponentially

	Annual improvements
Time-per-operation	59 %
Network-bandwidth	26 %
Network-latency	15 %

Source: Getting up to speed: The future of supercomputing (observed from 2004)



- **Goal**: Scalable and communication optimal tools for tensor computations

Communication Optimal Algorithms for Matrix and Tensor Computations

Hussam Al Daas[1], Grey Ballard[2], Laura Grigori[3], Suraj Kumar[4], Kathryn Rouse[5], and Mathieu Vérité[3]

[1]Rutherford Appleton Laboratory, UK

[2]Wake Forest University, USA

[3]EPFL, Switzerland

[4]Inria and ENS Lyon, France

[5]Inmar Intelligence, USA

Project committee meeting (Jan 17, 2025)

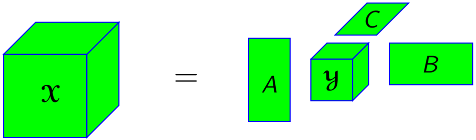
Outline

1. Parallel Multiple Tensor-Times-Matrix (Multi-TTM) computation
2. Parallel Nyström approximation with random matrices

Our approach:

- Obtain communication lower bounds for each computation
- Design algorithms based on the communication lower bounds

Higher-order SVD (HOSVD) to compute Tucker decomposition

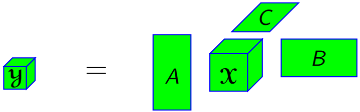


Algorithm 1 3-dimensional HOSVD Algorithm($\mathcal{X}$)

1: Obtain factor matrices A, B and C from the matrix representations of the input tensor $\mathcal{X}$
2: $\mathcal{Y} = \mathcal{X} \times_1 A^\mathsf{T} \times_2 B^\mathsf{T} \times_3 C^\mathsf{T}$
3: Return $\mathcal{Y}$, A, B, C

- $\mathcal{X}$, $\mathcal{Y}$: 3-dimensional input and output tensors (or arrays) & A, B, C: matrices
- $\times_i$: tensor contraction along the ith dimension (similar to matrix multiplication)
- Multiple Tensor-Times-Matrix (Multi-TTM) computation: $\mathcal{Y} = \mathcal{X} \times_1 A^\mathsf{T} \times_2 B^\mathsf{T} \times_3 C^\mathsf{T}$
- When A, B and C are obtained using randomized approaches, Multi-TTM becomes the bottleneck

Multi-TTM: $\mathcal{Y} = \mathcal{X} \times_1 A^\mathsf{T} \times_2 B^\mathsf{T} \times_3 C^\mathsf{T}$



- Focus on communication cost of Multi-TTM on a parallel homogeneous machine
- Multi-TTM is also the bottleneck computation for Sequentially Truncated HOSVD

Settings

- P number of processors
- Each processor performs (asymptotically) equal amount of operations
- One copy of data is in the system
- Focus on bandwidth cost (volume of data transfers)

H. Al Daas, G. Ballard, L. Grigori, S. Kumar, and K. Rouse, *Communication lower bounds and optimal algorithms for multiple tensor-times-matrix computation*, SIMAX, 2024.

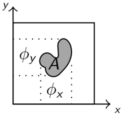
Outline

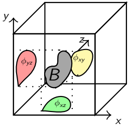
1. Parallel Multiple Tensor-Times-Matrix (Multi-TTM) computation
   - Our approach to compute communication lower bounds
     - For Traditional Matrix Matrix Multiplication
     - For Multi-TTM Computation
   - Communication Optimal Algorithm and Simulated Experiments
   - Conclusion

2. Parallel Nyström approximation with random matrices
   - Symmetric Nyström approximation
   - Conclusion

Approach to obtain communication lower bounds

- Loomis-Whitney inequalitiy: for $d-1$ dimensional projections
  - For the 2d object A, $\phi_x \phi_y \geq Area(A)$
  - For the 3d object B, $(\phi_{xy}\phi_{yz}\phi_{xz})^{\frac{1}{2}} \geq Volume(B)$





- Hölder-Brascamp-Lieb (HBL) inequality – generalization for arbitrary dimensional projections
  - Provide exponent for each projection

Constraints for parallel load balanced matrix matrix multiplication

- $C = AB$ with $A \in \mathbb{R}^{n_1 \times n_2}$, $B \in \mathbb{R}^{n_2 \times n_3}$, and $C \in \mathbb{R}^{n_1 \times n_3}$

$$\text{for } i = 1{:}n_1,\ \text{for } k = 1{:}n_2,\ \text{for } j = 1{:}n_3$$
$$C[i][j] += A[i][k] * B[k][j]$$

- ϕ_A, ϕ_B, ϕ_C: projections of computations on arrays A, B, C
- From Loomis-Whitney/HBL inequality: $\phi_A^{\frac{1}{2}}\phi_B^{\frac{1}{2}}\phi_C^{\frac{1}{2}} \geq$ number of multiplications per processor $= \frac{n_1 n_2 n_3}{P}$
- Extra constraints: $\frac{n_1 n_2}{P} \leq \phi_A \leq n_1 n_2$, $\frac{n_2 n_3}{P} \leq \phi_B \leq n_2 n_3$, $\frac{n_1 n_3}{P} \leq \phi_C \leq n_1 n_3$

Optimization problem and communication lower bounds

$$\begin{aligned}
&\textit{Minimize } \phi_A + \phi_B + \phi_C \quad \text{s.t.}\\
&\phi_A^{\frac{1}{2}} \phi_B^{\frac{1}{2}} \phi_C^{\frac{1}{2}} \geq \frac{n_1 n_2 n_3}{P}\\
&\frac{n_1 n_2}{P} \leq \phi_A \leq n_1 n_2\\
&\frac{n_2 n_3}{P} \leq \phi_B \leq n_2 n_3\\
&\frac{n_1 n_3}{P} \leq \phi_C \leq n_1 n_3
\end{aligned}$$

Amount of array accesses $= \phi_A + \phi_B + \phi_C$

- Estimate the solution and prove optimality by showing Karush–Kuhn–Tucker (KKT) conditions are satisfied
- For $n_1 \leq n_2 \leq n_3$,

Axis P, with points 1, $\frac{n_3}{n_2}$, $\frac{n_2 n_3}{n_1^2}$:

- Between 1 and $\frac{n_3}{n_2}$: $\phi_A = n_1 n_2$, $\phi_B = \frac{n_1 n_3}{P}$, $\phi_C = \frac{n_2 n_3}{P}$
- Between $\frac{n_3}{n_2}$ and $\frac{n_2 n_3}{n_1^2}$: $\phi_A = \phi_B = (\frac{n_1^2 n_2 n_3}{P})^{1/2}$, $\phi_C = \frac{n_2 n_3}{P}$
- Beyond $\frac{n_2 n_3}{n_1^2}$: $\phi_A = \phi_B = \phi_C = (\frac{n_1 n_2 n_3}{P})^{2/3}$

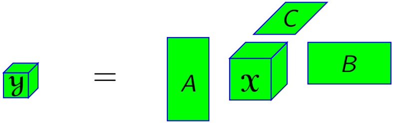
- Communication lower bound $= \phi_A + \phi_B + \phi_C -$ data owned by the processor $= \phi_A + \phi_B + \phi_C - \frac{n_1 n_2 + n_2 n_3 + n_1 n_3}{P}$

Outline

1. Parallel Multiple Tensor-Times-Matrix (Multi-TTM) computation
   - Our approach to compute communication lower bounds
     - For Traditional Matrix Matrix Multiplication
     - For Multi-TTM Computation
   - Communication Optimal Algorithm and Simulated Experiments
   - Conclusion

2. Parallel Nyström approximation with random matrices
   - Symmetric Nyström approximation
   - Conclusion

3-dimensional Multi-TTM computation



- $\mathcal{Y} = \mathcal{X} \times_1 A^\mathsf{T} \times_2 B^\mathsf{T} \times_3 C^\mathsf{T}$
- $\mathcal{X}$, $\mathcal{Y}$: 3-dimensional input and output tensors
- A, B, C: matrices
- $\times_i$: analogous to matrix multiplication

- TTM-in-Sequence approach (used in TuckerMPI library): $\mathcal{Y} = \left(\left(\mathcal{X} \times_1 A^\mathsf{T}\right) \times_2 B^\mathsf{T}\right) \times_3 C^\mathsf{T}$

- Our All-at-Once definition with $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\mathcal{Y} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$, $A \in \mathbb{R}^{n_1 \times r_1}$, $B \in \mathbb{R}^{n_2 \times r_2}$, $C \in \mathbb{R}^{n_3 \times r_3}$

$$\text{for } \{n_1', n_2', n_3', r_1', r_2', r_3'\} = 1{:}\{n_1, n_2, n_3, r_1, r_2, r_3\}$$
$$\mathcal{Y}(r_1', r_2', r_3') {+}{=} \mathcal{X}(n_1', n_2', n_3') \cdot A(n_1', r_1') \cdot B(n_2', r_2') \cdot C(n_3', r_3')$$

Solving optimization problems to compute lower bounds

- Each processor performs $\frac{n_1 r_1 n_2 r_2 n_3 r_3}{P}$ amount of $4 - array$ operations
- After applying lower and upper bounds for each projection, we need to solve the following optimization problem

$$
\begin{aligned}
Minimize\ \phi_{\mathcal{X}} + \phi_{\mathcal{Y}} + \phi_1 + \phi_2 + \phi_3 \quad \text{s.t.} \\
\phi_{\mathcal{X}}^{1-a} \phi_{\mathcal{Y}}^{1-a} \phi_1^a \phi_2^a \phi_3^a \geq \frac{n_1 r_1 n_2 r_2 n_3 r_3}{P} \\
\frac{n_1 n_2 n_3}{P} \leq \phi_{\mathcal{X}} \leq n_1 n_2 n_3 \\
\frac{r_1 r_2 r_3}{P} \leq \phi_{\mathcal{Y}} \leq r_1 r_2 r_3 \\
\frac{n_1 r_1}{P} \leq \phi_1 \leq n_1 r_1 \\
\frac{n_2 r_2}{P} \leq \phi_2 \leq n_2 r_2 \\
\frac{n_3 r_3}{P} \leq \phi_3 \leq n_3 r_3 \\
0 \leq a \leq 1
\end{aligned}
$$

Divide the problem into two parts

Matrix part

$$\textit{Minimize } \phi_1 + \phi_2 + \phi_3 \text{ s.t.}$$

$$\phi_1\phi_2\phi_3 \geq \frac{n_1 r_1 n_2 r_2 n_3 r_3}{P}$$

$$\frac{n_1 r_1}{P} \leq \phi_1 \leq n_1 r_1$$

$$\frac{n_2 r_2}{P} \leq \phi_2 \leq n_2 r_2$$

$$\frac{n_3 r_3}{P} \leq \phi_3 \leq n_3 r_3$$

Tensor part

$$\textit{Minimize } \phi_{\mathcal{X}} + \phi_{\mathcal{Y}} \text{ s.t.}$$

$$\phi_{\mathcal{X}}\phi_{\mathcal{Y}} \geq \frac{n_1 r_1 n_2 r_2 n_3 r_3}{P}$$

$$\frac{n_1 n_2 n_3}{P} \leq \phi_{\mathcal{X}} \leq n_1 n_2 n_3$$

$$\frac{r_1 r_2 r_3}{P} \leq \phi_{\mathcal{Y}} \leq r_1 r_2 r_3$$

Amount of accesses and lower bounds

- We assume $n_1 r_1 \leq n_2 r_2 \leq n_3 r_3$ and $r_1 r_2 r_3 \leq n_1 n_2 n_3$
- Estimate the solutions and prove optimality by showing KKT conditions are satisfied

Amount of accesses $= \phi_x + \phi_y + \phi_1 + \phi_2 + \phi_3$

Axis from 1 to P:

- From 1 to $\frac{n_3 r_3}{n_2 r_2}$: $\phi_1 = n_1 r_1$, $\phi_2 = n_2 r_2$, $\phi_3 = \frac{n_3 r_3}{P}$
- From $\frac{n_3 r_3}{n_2 r_2}$ to $\frac{n_2 r_2 n_3 r_3}{n_1^2 r_1^2}$: $\phi_1 = n_1 r_1$, $\phi_2 = \phi_3 = \left(\frac{n_2 r_2 n_3 r_3}{P}\right)^{1/2}$
- From $\frac{n_2 r_2 n_3 r_3}{n_1^2 r_1^2}$ to P: $\phi_1 = \phi_2 = \phi_3 = \left(\frac{n_1 r_1 n_2 r_2 n_3 r_3}{P}\right)^{1/3}$

Axis from 1 to P:

- From 1 to $\frac{n_1 n_2 n_3}{r_1 r_2 r_3}$: $\phi_y = r_1 r_2 r_3$, $\phi_x = \frac{n_1 n_2 n_3}{P}$
- From $\frac{n_1 n_2 n_3}{r_1 r_2 r_3}$ to P: $\phi_x = \phi_y = \left(\frac{n_1 r_1 n_2 r_2 n_3 r_3}{P}\right)^{1/2}$

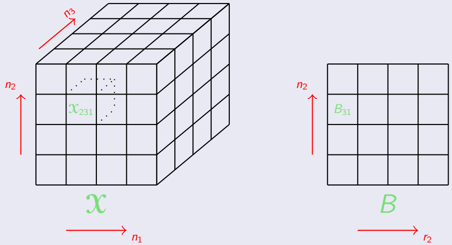
Communication lower bound $= \phi_x + \phi_y + \phi_1 + \phi_2 + \phi_3 - \frac{n_1 n_2 n_3 + r_1 r_2 r_3 + n_1 r_1 + n_2 r_2 + n_3 r_3}{P}$

Outline

Design of communication optimal algorithms

Data Distribution (P is organized into a $p_1 \times p_2 \times p_3 \times q_1 \times q_2 \times q_3$ grid)

- p_1, p_2, p_3, q_1, q_2, and q_3 evenly distribute n_1, n_2, n_3, r_1, r_2, and r_3
- Each processor has $\frac{1}{P}$th amount of input and output variables
- Subtensor $\mathcal{X}_{231} = \mathcal{X}(\frac{n_1}{p_1} + 1 : 2\frac{n_1}{p_1}, 2\frac{n_2}{p_2} + 1 : 3\frac{n_2}{p_2}, 1 : \frac{n_3}{p_3})$ is distributed evenly among processors $(2, 3, 1, *, *, *)$
- Submatrix $B_{31} = B(2\frac{n_2}{p_2} + 1 : 3\frac{n_2}{p_2}, 1 : \frac{r_2}{q_2})$ is distributed evenly among processors $(*, 3, *, *, 1, *)$



6-dimensional algorithm to compute Multi-TTM

Algorithm 1 3-dimensional Parallel Atomic Multi-TTM

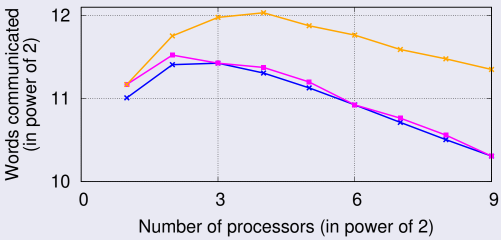
Require: $\mathcal{X}$, A, B, C, $p_1 \times p_2 \times p_3 \times q_1 \times q_2 \times q_3$ logical processor grid

Ensure: $\mathcal{Y}$ such that $\mathcal{Y} = \mathcal{X} \times_1 A^\mathsf{T} \times_2 B^\mathsf{T} \times_3 C^\mathsf{T}$

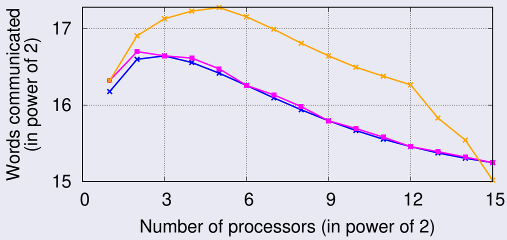
$(p_1', p_2', p_3', q_1', q_2', q_3')$ is my processor id
//All-gather input tensor and matrices
$\mathcal{X}_{p_1' p_2' p_3'} =$ All-Gather$(\mathcal{X}, (p_1', p_2', p_3', *, *, *))$
$A_{p_1' q_1'} =$ All-Gather$(A, (p_1', *, *, q_1', *, *))$
$B_{p_2' q_2'} =$ All-Gather$(B, (*, p_2', *, *, q_2', *))$
$C_{p_3' q_3'} =$ All-Gather$(C, (*, *, p_3', *, *, q_3'))$
//Perform local Multi-TTM computation in a temporary tensor $\mathcal{T}$
$\mathcal{T} =$ Local-Multi-TTM$(\mathcal{X}_{p_1' p_2' p_3'}, A_{p_1' q_1'}, B_{p_2' q_2'}, C_{p_3' q_3'})$
//Reduce-scatter the output tensor in $\mathcal{Y}_{q_1' q_2' q_3'}$
Reduce-Scatter$(\mathcal{Y}_{q_1' q_2' q_3'}, \mathcal{T}, (*, *, *, q_1', q_2', q_3'))$

Performance comparison (simulated experiments)

$n_1 = n_2 = n_3 = 2^8, r_1 = r_2 = r_3 = 2^3$



$n_1 = n_2 = n_3 = 2^{11}, r_1 = r_2 = r_3 = 2^5$

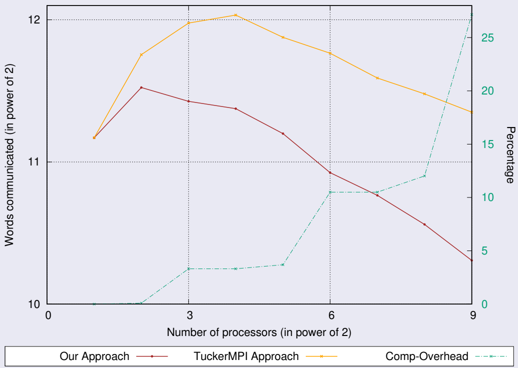


- Typical scenarios in data compression problems
- For small P, our approach communicates much less than TuckerMPI approach

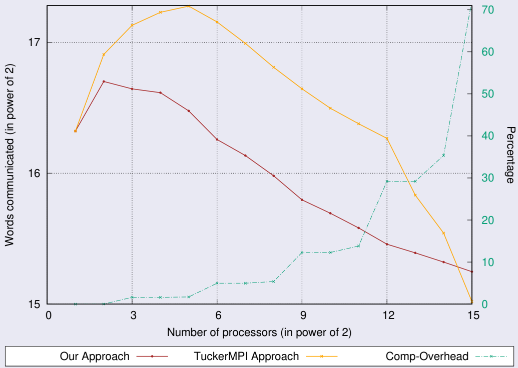
Implementation of the proposed approach: R. Minster, Z. Li, and G. Ballard, *Parallel Randomized Tucker Decomposition Algorithms*, SISC, 2024.

Extra computation in our approach

$n_1 = n_2 = n_3 = 2^8, r_1 = r_2 = r_3 = 2^3$



$n_1 = n_2 = n_3 = 2^{11}, r_1 = r_2 = r_3 = 2^5$



Outline

1. Parallel Multiple Tensor-Times-Matrix (Multi-TTM) computation
   - Our approach to compute communication lower bounds
     - For Traditional Matrix Matrix Multiplication
     - For Multi-TTM Computation
   - Communication Optimal Algorithm and Simulated Experiments
   - Conclusion
2. Parallel Nyström approximation with random matrices
   - Symmetric Nyström approximation
   - Conclusion

Conclusion and future work

Conclusion

- Communication lower bounds and optimal algorithms for All-at-Once Multi-TTM
- Our algorithm communicates much less data than TTM-in-Sequence for small P

Future Work

- Detailed study of what scenarios are favorable for our approach
- Combine both All-at-Once and TTM-in-Sequence approaches

Outline

1. Parallel Multiple Tensor-Times-Matrix (Multi-TTM) computation
2. Parallel Nyström approximation with random matrices

Nonsymmetric Nyström approximation

$\tilde{A} = (AV)(U^T AV)^{\dagger}(A^T U)^T$ with $A \in \mathbb{R}^{n_1 \times n_2}$, $U \in \mathbb{R}^{n_1 \times r_1}$, and $V \in \mathbb{R}^{n_2 \times r_2}$.

- U and V are random matrices
  - can be generated on any processor without any extra communication costs

- Need to focus on $D = U^T A$, $B = AV$ and $C = U^T AV$ computations together

We will focus mainly on symmetric Nyström approximation today.

Outline

1. Parallel Multiple Tensor-Times-Matrix (Multi-TTM) computation
   - Our approach to compute communication lower bounds
     - For Traditional Matrix Matrix Multiplication
     - For Multi-TTM Computation
   - Communication Optimal Algorithm and Simulated Experiments
   - Conclusion

2. Parallel Nyström approximation with random matrices
   - Symmetric Nyström approximation
   - Conclusion

Symmetric Nyström approximation

$\tilde{A} = (AV)(V^TAV)^\dagger(AV)^T$ with $A \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{n\times r}$.

- V is a random matrix and $A = A^T$
- Need to focus on $B = AV$ and $C = V^TAV$ computations together

A naive way to minimize communication cost for $B = AV$

$$\begin{pmatrix} B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \cdot V$$

Structure of the algorithm for each processor i:

- owns A_i row block ($1/P$ th portion of A), generates random matrix V, and performs $B_i = A_i \cdot V$

When P is small, communication cost of the algorithm is 0 (Optimal).

What about when P is large?

Solving optimization problems to compute lower bounds for $B = AV$

- Each processor performs $\frac{n^2 r}{P}$ multiplications
- After applying lower and upper bounds for each projection, we need to solve the following optimization problem

$$Minimize\ \phi_A + \phi_B \text{ s.t.}$$

$$\phi_A \phi_B \geq ncols(\phi_A)\phi_B \geq \frac{n^2 r}{P}$$

$$\phi_A^{1/2} \phi_B^{1/2} \phi_V^{1/2} \geq \frac{n^2 r}{P}$$

$$\frac{n^2}{P} \leq \phi_A \leq n^2,\ \frac{nr}{P} \leq \phi_B \leq nr,\ \frac{nr}{P} \leq \phi_V \leq nr,\ r \leq n$$

Communication lower bound $= \phi_A + \phi_B - \frac{nr+n^2}{P}$

P: 1 — n — $\frac{n^2}{r}$ —

$1 \le P \le n$: $\phi_A = n^2/P$, $\phi_B = nr/P$

$n \le P \le \frac{n^2}{r}$: $\phi_A = n^2/P$, $\phi_B = r$

$P \ge \frac{n^2}{r}$: $\phi_A = \phi_B = \left(\frac{n^2 r}{P}\right)^{1/2}$

Our algorithm to compute $B = AV$ computation

Algorithm 2 $B = AV$ computation

Require: A, $p_1 \times p_2 \times p_3$ logical processor grid

Ensure: B such that $B = AV$

1: (p_1', p_2', p_3') is my processor id
2: $A_{p_1'p_2'} = \text{All-Gather}(A, (p_1', p_2', *))$ //All-gather input matrix A
3: $V_{p_2'p_3'} = \text{GenerateRequiredRandomMatrix}()$ //Generate random matrix of size $\frac{n}{p_2} \times \frac{r}{p_3}$
4: //Perform local computation in a temporary matrix T
5: $T = A_{p_1'p_2'} \cdot V_{p_2'p_3'}$
6: $\text{Reduce-Scatter}(B_{p_1'p_3'}, T, (p_1', *, p_3'))$ //Reduce-scatter the output matrix in $B_{p_1'p_3'}$

Our algorithm is communication optimal when p_1, p_2 & p_3 are chosen based on lower bounds.

Compute lower bounds for $B = AV$ and $C = V^T AV$

$$B = AV \qquad \text{and} \qquad C = V^T B$$

- Each processor performs $\frac{n^2 r}{P}$ multiplications to compute B and $\frac{nr^2}{P}$ multiplications to compute C
- Assume that same entries of B are accessed on a processor for both computations
- After combining iteration spaces of both computations in a 6-dimensional lattice, we need to solve the following optimization problem

$$\mathit{Minimize}\ \phi_A + \phi_B + \phi_C \ \text{ s.t.}$$

$$\phi_A \phi_B \phi_C \geq \frac{n^2 r}{P} \cdot \frac{nr^2}{P}$$

$$\frac{n^2}{P} \leq \phi_A \leq n^2,\ \frac{nr}{P} \leq \phi_B \leq nr,\ \frac{r^2}{P} \leq \phi_C \leq r^2,\ r \leq n$$

- Can obtain communication lower bounds by solving the above optimization problem
- Similar to the previous algorithms, we can design communication optimal algorithms for a $p_1 \times p_2 \times p_3$ logical processor grid

Outline

Conclusion and future work

Conclusion

- Communication lower bounds and optimal algorithms for the computations of symmetric Nyström approximation
- An approach to obtain communication lower bounds for a set of computations

Future Work

- Implementation of the proposed algorithms for real datasets
- Simplify and refine the lower bound computations

Thank You!