Parallel Tensor Compression for Large-Scale Scientific Data

Woody Austin, Grey Ballard, Tamara G. Kolda

April 14, 2016

SIAM Conference on Parallel Processing for Scientific Computing
MS 44/52: Parallel Algorithms for Tensor Computations
Summary

Natural five-way multiway structure of scientific data

We use a Tucker model to approximate dense tensor data
We introduce a parallel algorithm to compress large data sets

Compression rates as fidelity varies for 550 GB simulation dataset
Tucker Tensor Model

Tucker model generalizes the matrix SVD

\[ \mathcal{T} \approx \mathcal{G} \times_1 U \times_2 V \times_3 W \]

\( \mathcal{G} \) is core tensor \( U, V, W \) are factor matrices

Factor matrices usually tall and skinny with orthonormal columns

Notation conventions: scalar \( N \), vector \( v \), matrix \( M \), tensor \( \mathcal{T} \) [KB09]
Tensor-Times-Matrix (TTM)

Tensor version:

\[ Y = X \times_2 M \]

\[ Y \in \mathbb{R}^{I\times Q\times K} \quad X \in \mathbb{R}^{I\times J\times K} \quad M \in \mathbb{R}^{Q\times J} \]

Matrix version:

\[ Y_{(2)} = MX_{(2)} \]

\[ Y_{(2)} \in \mathbb{R}^{Q\times IK} \quad X_{(2)} \in \mathbb{R}^{J\times IK} \]

TTM is matrix multiplication with specified unfolding

Notation: \( T_{(n)} \) is unfolding of \( T \) in mode \( n \)
Fibers, Slices, and Unfoldings

A tensor can be reshaped into matrices called unfoldings

- columns correspond to fibers
- rows correspond to slices

Notation: $T_{(n)}$ is unfolding of $T$ in mode $n$
For fixed ranks, we want to solve

$$\min_{G, U, V, W} \|X - G \times_1 U \times_2 V \times_3 W\|,$$

which turns out to be equivalent to

$$\max_{U, V, W} \|G\| \text{ subject to } G = X \times_1 U^T \times_2 V^T \times_3 W^T,$$

a nonlinear, nonconvex optimization problem.
Fixing all but one factor matrix, we have a matrix problem:

$$\max_V \left\| \mathbf{X} \times_1 \hat{\mathbf{U}}^T \times_2 \mathbf{V}^T \times_3 \hat{\mathbf{W}}^T \right\|$$

or equivalently

$$\max_V \left\| \mathbf{V}^T \mathbf{Y}_{(2)} \right\|_F$$

where \( \mathbf{Y} = \mathbf{X} \times_1 \hat{\mathbf{U}}^T \times_3 \hat{\mathbf{W}}^T \), which can be solved with SVD of \( \mathbf{Y}_{(2)} \)

HOOI [DDV00, KD80] works by alternating over factor matrices, updating one at a time by computing leading left singular vectors.
Sequentially Truncated Higher-Order SVD

- HOOI needs accurate initialization

- Truncated Higher-Order SVD (T-HOSVD) typically used

- ST-HOSVD [VVM12] is more efficient than T-HOSVD, works by
  - initializing with identity matrices $U = I$, $V = I$, $W = I$
  - applying one iteration of HOOI
  - ranks can be chosen based on error tolerance
ST-HOSVD Algorithm

Input: $X$

1. $S^{(1)} \leftarrow X^{(1)}X^{(1)T}$
2. $U = \text{leading eigenvectors of } S^{(1)}$
3. $Y = X \times_1 U$
4. $S^{(2)} \leftarrow Y^{(2)}Y^{(2)T}$
5. $V = \text{leading eigenvectors of } S^{(2)}$
6. $Z = Y \times_2 V$
7. $S^{(3)} \leftarrow Z^{(3)}Z^{(3)T}$
8. $W = \text{leading eigenvectors of } S^{(3)}$
9. $G = Z \times_3 W$

Left singular vectors of $A$ computed as eigenvectors of $A^TA$
Data Distribution and Parallelization

Key data entities in ST-HOSVD are
- tensors: data tensor $X$, intermediates $Y$ and $Z$, core tensor $G$
- matrices: factor matrices $U, V, W$, Gram matrices $S^{(1)}, S^{(2)}, S^{(3)}$

Key computations in ST-HOSVD are
- Gram: computing $S^{(2)} = X_{(2)} X_{(2)}^T$
- Eigenvectors: computing leading eigenvectors of $S^{(2)}$
- TTM: computing $Y_{(2)} = V^T X_{(2)}$
Parallel Block Tensor Distribution

For $N$-mode tensor, use logical $N$-mode processor grid

Proc. grid: $P_I \times P_J \times P_K = 3 \times 5 \times 2$

Local tensors have dimensions $\frac{I}{P_I} \times \frac{J}{P_J} \times \frac{K}{P_K}$
Key idea: each unfolded matrix is 2D block distributed

Proc. grid: $P_I \times P_J \times P_K = 3 \times 5 \times 2$

Logical mode-2 2D processor grid: $P_J \times P_I P_K$
Local unfolded matrices have dimensions $\frac{J}{P_J} \times \frac{IK}{P_I P_K}$
Key idea: store factor matrix redundantly across processor fibers

Proc. grid: $P_I \times P_J \times P_K = 3 \times 5 \times 2$

$V$ stored in 1D layout across each fiber of $P_J = 5$ processors

$P_I P_K = 6$ copies of $V$ stored
Data distributions match TTM computation perfectly
- fibers work independently
- communication consists of reduce-scatter(s) within fiber
- output distributed to match tensor block distribution
ST-HOSVD Algorithm (again)

Input: $X$

1. $S^{(1)} \leftarrow X_{(1)}X_{(1)}^T$
2. $U = \text{leading eigenvectors of } S^{(1)}$
3. $Y = X \times_1 U$
4. $S^{(2)} \leftarrow Y_{(2)}Y_{(2)}^T$
5. $V = \text{leading eigenvectors of } S^{(2)}$
6. $Z = Y \times_2 V$
7. $S^{(3)} \leftarrow Z_{(3)}Z_{(3)}^T$
8. $W = \text{leading eigenvectors of } S^{(3)}$
9. $G = Z \times_3 W$

$\leftarrow$ Gram must also be parallelized
$\leftarrow$ done redundantly
$\leftarrow$ TTM makes next step smaller
Given a tensor $X$ and number of processors $P$, 

- ST-HOSVD can be performed in any mode order
  - affecting both computation and communication costs
  - yielding (slightly) different results

- $P$ can be logically decomposed into many processor grids
  - having large effects on communication cost
Varying mode order for tensor of size $25 \times 250 \times 250 \times 250$ with reduced size $10 \times 10 \times 100 \times 100$. Varying processor grid for tensor of size $384 \times 384 \times 384 \times 384$ with reduced size of $96 \times 96 \times 96 \times 96$. 
Parallel Scaling

**Weak scaling for**

$200k \times 200k \times 200k \times 200k$

tensor with reduced size

$20k \times 20k \times 20k \times 20k$,

using $k^4$ nodes for $1 \leq k \leq 6$.

**Strong scaling for**

$200 \times 200 \times 200 \times 200$

tensor with reduced size

$20 \times 20 \times 20 \times 20$,

using $2^k$ nodes for $0 \leq k \leq 9$. 
Compression of Scientific Simulation Data

We applied ST-HOSVD to compress multidimensional data from numerical simulations of combustion, including the following data sets:

- **HCCI:**
  - Dimensions: $672 \times 672 \times 33 \times 627$
  - $672 \times 672$ spatial grid, 33 variables over 627 time steps
  - Total size: 70 GB

- **TJLR:**
  - Dimensions: $460 \times 700 \times 360 \times 35 \times 16$
  - $460 \times 700 \times 360$ spatial grid, 35 variables over 16 time steps
  - Total size: 520 GB

- **SP:**
  - Dimensions: $500 \times 500 \times 500 \times 11 \times 50$
  - $500 \times 500 \times 500$ spatial grid, 11 variables over 50 time steps
  - Total size: 550 GB
Compression ratio: \[ \frac{IJK}{PQR+IP+JQ+KR} \]

Relative Normwise Error: \[ \frac{\|x - \hat{x}\|}{\|x\|} \]
Summary

- Tucker is a powerful tool for multidimensional compression

- Large data sets require efficient parallel algorithms

- We propose an algorithm that performs and scales well for dense data sets
Parallel Tensor Compression for Large-Scale Scientific Data
Woody Austin, Grey Ballard, and Tamara G. Kolda
International Parallel and Distributed Processing Symposium 2016
http://arxiv.org/abs/1510.06689
References


