Optimizations for Symmetric Tensor Computations

Jonathon Cai

Dept. of Computer Science
Yale University

Muthu Baskaran, Benoit Meister, Richard Lethin

Reservoir Labs

Introduction

ENSIGN (Exascale Non-Stationary Graph Notation)

- Tool for hypergraph (or multi-link graph) analysis as tensor decompositions
  - Automatic source-to-source optimization tool
    - High-level specification → fast parallel target code
- Solve important graph and data analytic problems
- Offers performance and productivity benefits
  - Compiler technologies to improve and scale existing key tensor computations
    - Inter-operate with Reservoir Labs' auto-parallelizing compiler R-Stream
  - Quick turnaround time from prototyping to developing high-performance codes

Presentation Roadmap

General Background

Tensor Terminology

Optimization and Parallelization Techniques

Performance Evaluation

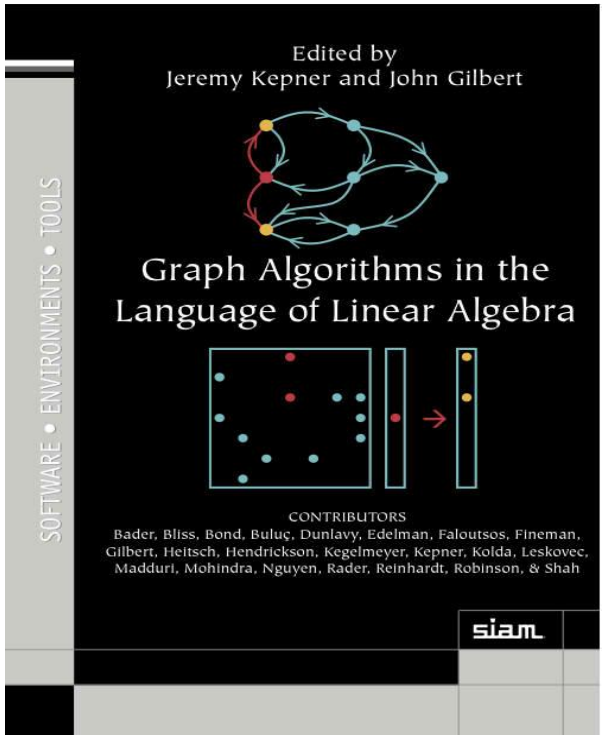
Summary & Forward Work

Linear Algebra and Graph Analysis

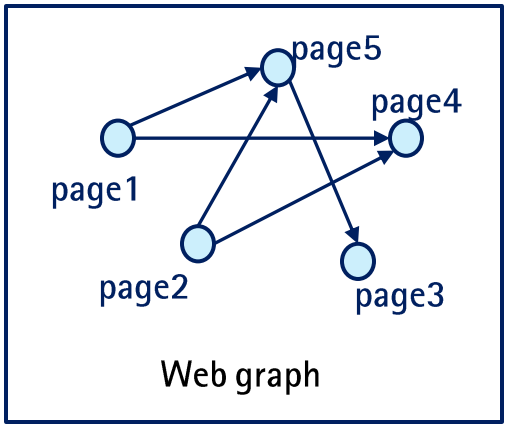
Popular graph problems use linear algebra formulations

- PageRank [Brin & Page, 1998]
  - Ranking web pages based on importance
  - Finding dominant eigenvectors
- HITS [Kleinberg, 1998]
  - Finding authoritative web pages
  - Finding principal singular vectors
- Common graph computations
  - Let A be the adjacency matrix,

    D the degree matrix.
    - Connected components: Zero eigenvectors of D – A.
    - Community detection: Least non-zero eigenvector of D – A.



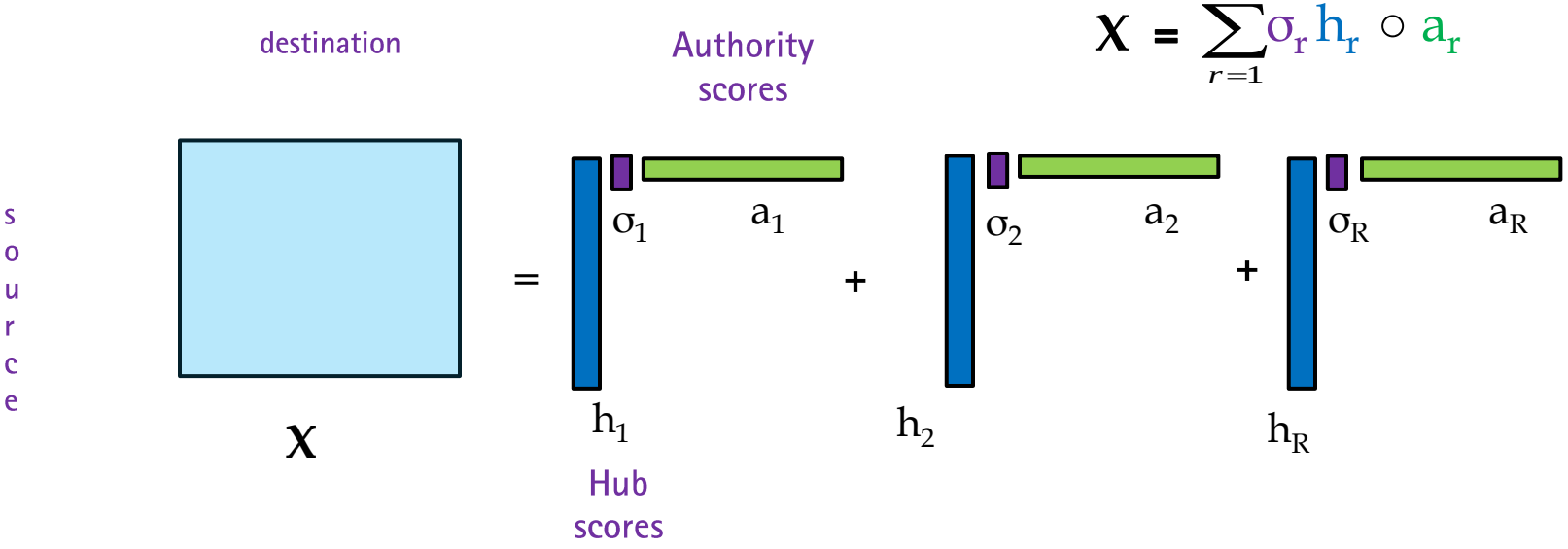
Web graph analysis using SVD



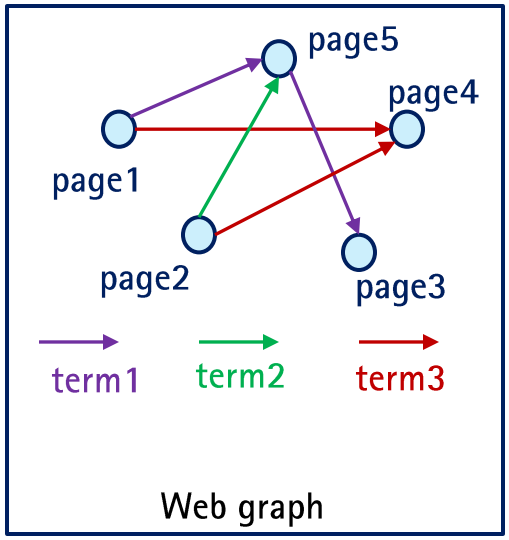
- Find authorities and hubs
  - Authorities: Pages that many important pages point to
  - Hubs: Pages that point to good authorities
- SVD on the "sparse" adjacency matrix of web graph

$$\mathbf{X} = \mathrm{H}\Sigma\mathrm{A}^{\mathrm{T}}$$

$$\mathbf{X} = \sum_{r=1}^{R} \sigma_r \, h_r \circ a_r$$

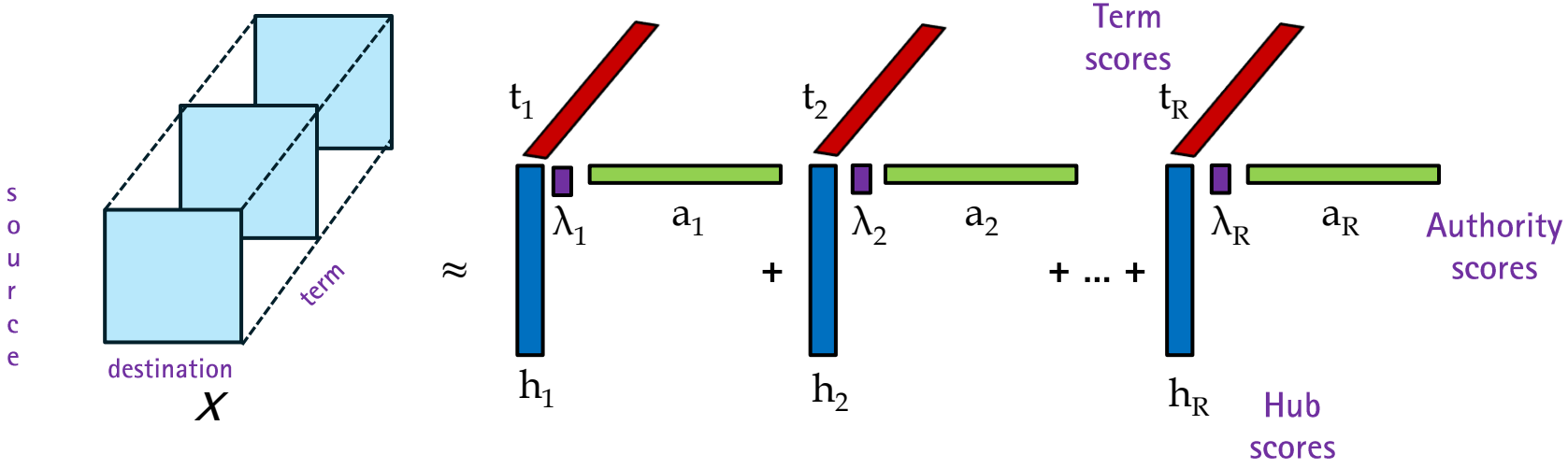


Tensor analogues of web graph analysis



- 3D view of the web graph with topical information added
  - 3D adjacency "tensor"
- TOPHITS [Kolda et al., 2005]
- Tensor decomposition of the adjacency tensor

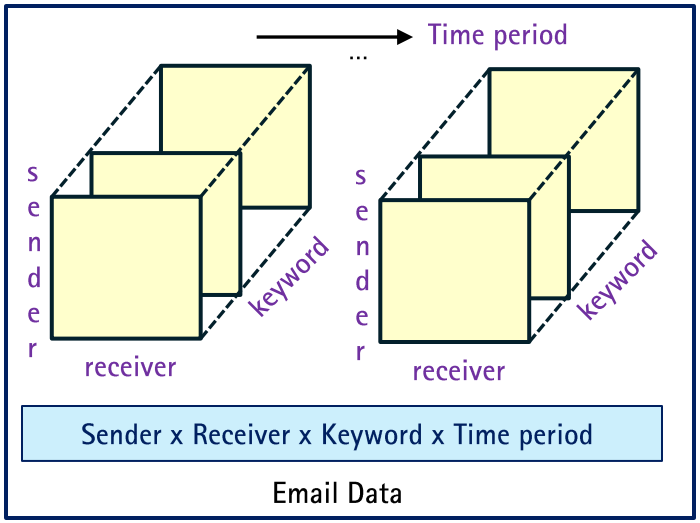
$$\mathcal{X} \approx \sum_{r=1}^{R} \lambda_r \, h_r \circ a_r \circ t_r$$

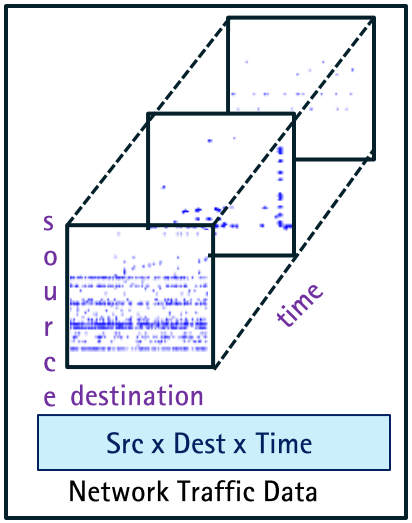


Multi-link Graphs or Hypergraphs and Multi-aspect Data

Real world Data

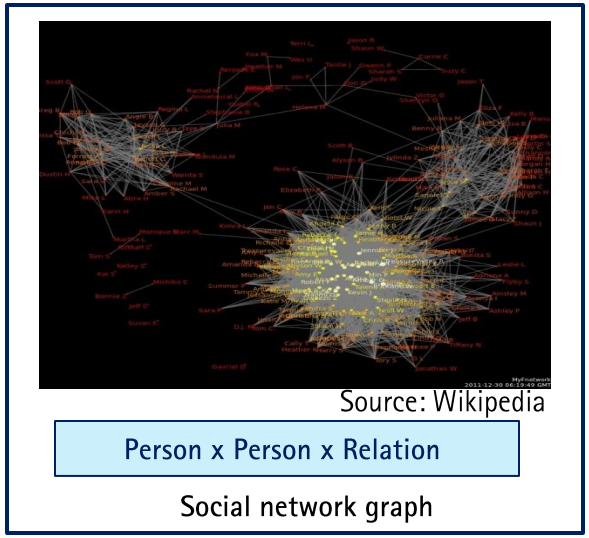
- Multi-dimensional
- Multi-aspect
- Large
- Sparse
- *Symmetric*





Symmetric Tensor Examples

- Time-evolving graph
- Higher order derivatives (generalization of Hessian)
- Cumulants of random vectors
- Many more



Presentation Roadmap

General Background

Tensor Terminology

Optimization and Parallelization Techniques

Performance Evaluation

Summary & Forward Work

Basic Tensor Terminology

- Order refers to number of dimensions and mode refers to particular dimension

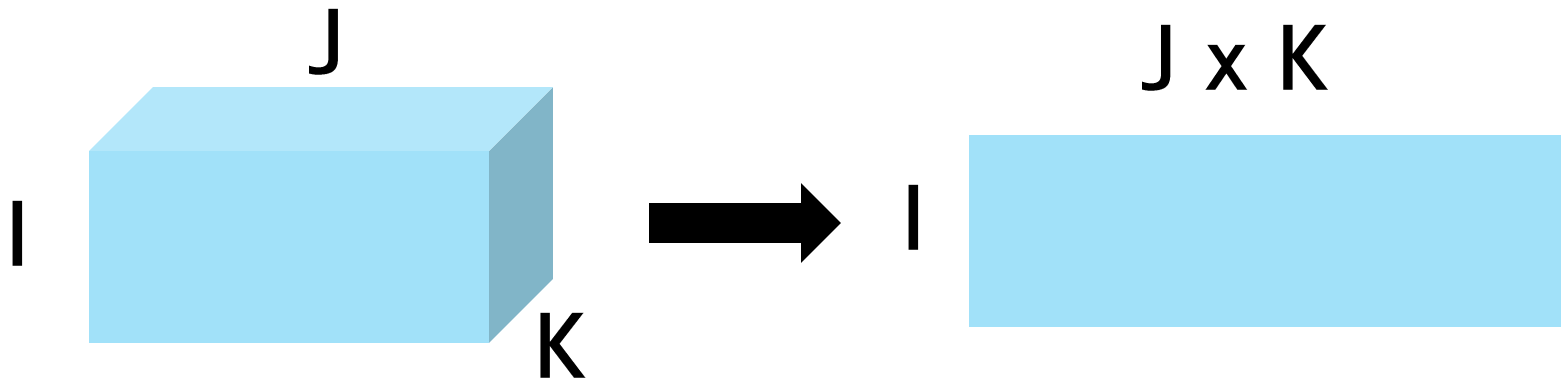
- A tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times I}$ is supersymmetric if:

$$x_{ijk} = x_{ikj} = x_{jik} = x_{jki} = x_{kij} = x_{kji}$$

- A tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times K}$ is partially symmetric in the first two modes if:

$$x_{ijk} = x_{jik}$$

Tensor Flattening



$$\mathcal{X}(:,:,1) = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \mathcal{X}(:,:,2) = \begin{bmatrix} 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

$$\mathbf{X}_{(1)} = \begin{bmatrix} 1 & 2 & 3 & 7 & 8 & 9 \\ 4 & 5 & 6 & 10 & 11 & 12 \end{bmatrix}$$

$$\mathbf{X}_{(2)} = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}$$

$$\mathbf{X}_{(3)} = \begin{bmatrix} 1 & 4 & 2 & 5 & 3 & 6 \\ 7 & 10 & 8 & 11 & 9 & 12 \end{bmatrix}$$

Unfolding of tensor $\mathcal{X} \in \mathbb{R}^{2\times3\times2}$

Matricized Tensor Times Khatri Rao Product

- Khatri Rao Product:

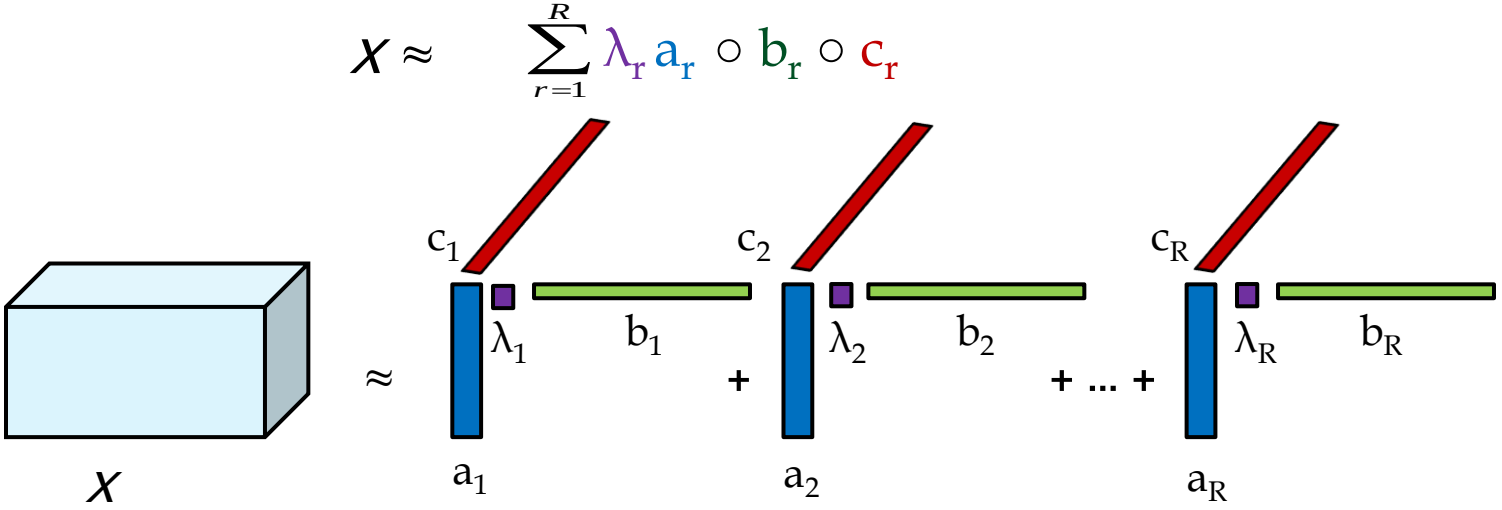
$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \dots & a_{1K}b_{1K} \\ a_{11}b_{21} & a_{12}b_{22} & \dots & a_{1K}b_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ a_{11}b_{J1} & a_{12}b_{J2} & \dots & a_{1K}b_{JK} \\ a_{21}b_{11} & a_{22}b_{12} & \dots & a_{2K}b_{1K} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{J1} & a_{I2}b_{J2} & \dots & a_{IK}b_{JK} \end{bmatrix}$$

- Matricized tensor times Khatri Rao product (mttkrp):

$$\mathbf{Z}_{(n)}(\mathbf{A}^{(N)} \odot \ldots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \ldots \odot \mathbf{A}^{(1)})$$

CP Tensor Decomposition

- CANDECOMP/PARAFAC (CP) tensor decomposition
  - Factorizes a tensor into a sum of component rank-one tensors



  - Decomposition algorithms
    - CP-ALS, CP-APR, CP-OPT

INDSCAL Decomposition Algorithm (IND-OPT)

- INDSCAL decomposition is a special version of CANDECOMP/PARAFAC (CP) decomposition, with first two modes equivalent

$$\mathcal{X} \approx \sum_{r=1}^{R} a_r \circ a_r \circ c_r$$

- Implemented IND-OPT, version of CP-OPT algorithm optimized for symmetric tensors
- IND-OPT uses a L-BFGS algorithm with line search
- Optimize least squares fit function: $\frac{1}{2}\| \mathcal{Z} - [\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}] \|_2$

- Compute gradient function many times

$$\frac{\partial f}{\partial \mathbf{A}^{(n)}} = -\mathbf{Z}_{(n)}(\mathbf{A}^{(N)} \odot \ldots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \ldots \odot \mathbf{A}^{(1)}) + \mathbf{A}^{(n)}\Gamma^{(n)}$$

  - Dominant operation is mttkrp

Presentation Roadmap

General Background

Tensor Terminology

Optimization and Parallelization Techniques

Performance Evaluation

Summary & Forward Work

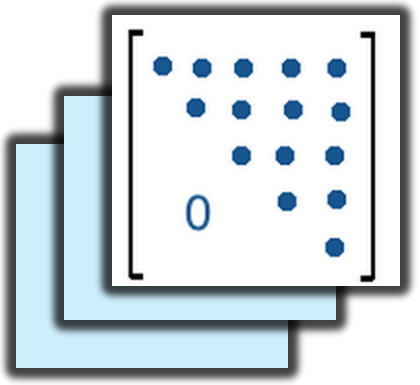
Focus of this Work

Challenge: Optimizing symmetric tensor computations

- Need to implement a storage format for improved data locality and facilitating parallelism
- Need to efficiently compute the mttkrp, exploiting symmetry
- For this talk, we consider a tensor $\mathcal{Z} \in \mathcal{R}^{I_n \times I_n \times I_k}$ partially symmetric in the first two modes

Storage Format: Upper Hypertriangle of Tensor

Challenge: Optimizing storage



- Stacks of upper triangular slices

- For the partially symmteric tensor $\mathcal{Z} \in \mathcal{R}^{I_n \times I_n \times I_k}$ storage reduction from $I_n \times I_n \times I_k$ to $\frac{I_n \times (I_n + 1)}{2} \times I_k$

Optimized Algorithms for mttkrp: High-level Ideas

Challenge: Optimizing symmetric tensor mttkrp

- Do not construct entire Khatri Rao product, a large dense matrix, in storage; compute intermediate results and then use appropriately on the fly
- Do not compute entire Khatri Rao product and multiply by flattened tensor; fuse computations
- Two cases: symmetric and non-symmetric

$$\mathbf{Z}_{(2)}(\mathbf{B} \odot \mathbf{A}) \quad \mathbf{Z}_{(3)}(\mathbf{A} \odot \mathbf{A})$$

~~$$\mathbf{Z}_{(1)}(\mathbf{B} \odot \mathbf{A})$$~~

Case 1: Non-symmetric mttkrp kernel $\mathbf{Z}_{(3)}(\mathbf{A} \odot \mathbf{A})$

- Recognize that KRP has redundant elements – only need to compute rows corresponding to the upper triangular indices
- Compute each row of KRP and multiply with appropriate elements in flattened tensor, where there are also redundancies due to partial symmetry of tensor
- KRP storage reduction: $(I_n)^2 \times R$ to R

Case 2: Symmetric mttkrp kernel $\mathbf{Z}_{(2)}(\mathbf{B} \odot \mathbf{A})$

- Observe tensor elements with common third mode are multiplied by the same matrix elements in $\mathbf{B}$
- For parallel case, compute an intermediate of size $\frac{I_n \times (I_n + 1)}{2} \times R$ where the r-th element in each row represents the fused partial product $\sum_k \mathcal{Z}_{ijk} \mathbf{B}(k, r)$
  - intermediate can be reduced to size R but by sacrificing parallelism
- KRP storage for Parallel: $(I_n)^2 \times R$ to $\frac{I_n \times (I_n + 1)}{2} \times R$
- KRP storage for Sequential: $(I_n)^2 \times R$ to R

Summary: Optimized Algorithms for mttkrp

- By exploiting redundancies, total number of computations is reduced compared to original
- mttkrp is parallelized
- By using special storage format, data locality is improved

Presentation Roadmap

Background

Tensor Terminology

Optimization and Parallelization Techniques

Performance Evaluation

Summary & Forward Work

Performance Evaluation

Benchmarked using IND-OPT method

- Evaluated using
  - Four synthetically generated tensor data sets
  - One real data set (Facebook graph)
  - Intel Xeon E5-4620 2.2 GHz (Quad socket 8-core)

Reservoir "Tensorstation"

Mttkrp kernel proportion time in IND-OPT

Data Set	mttkrp % time	Remaining % Time
Syn-120	99.45	0.55
Syn-1000	99.93	0.07
Syn-500	99.99	0.01
Syn-200	99.93	0.07
facebook-m	99.99	0.01

Percentage of time spent in mttkrp kernel

Sequential Performance Evaluation

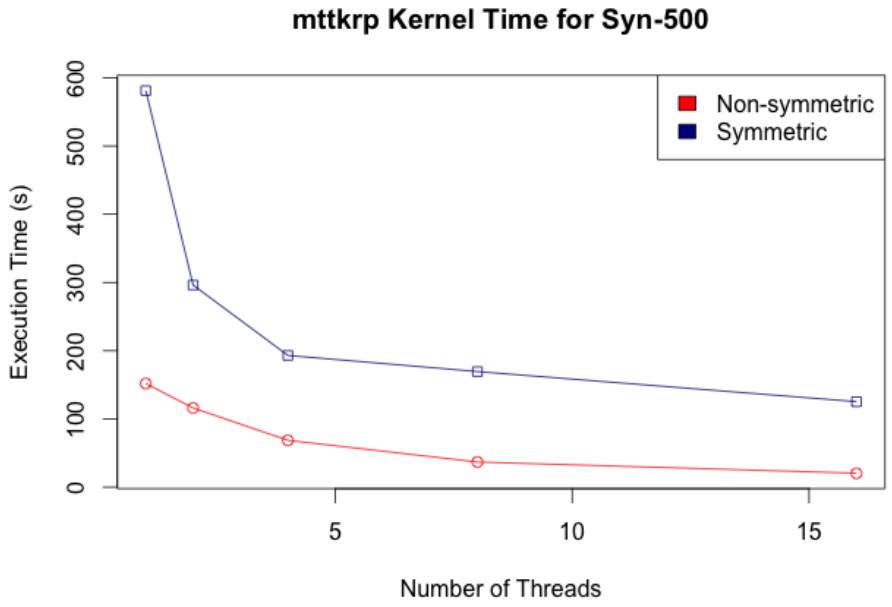
Data Set	IND-OPT time (s)	INDSCAL Time (s)
Syn-120	8.27	176.98
Syn-1000	771.84	22347.10
Syn-500	250.61	8023.86
Syn-200	308.21	12173.50
facebook-m	50.43	2101.46

Sequential performance of IND-OPT vs INDSCAL

- From 21x for Syn-120 up to 39x for Syn-200

Synthetic Tensor Performance Evaluation

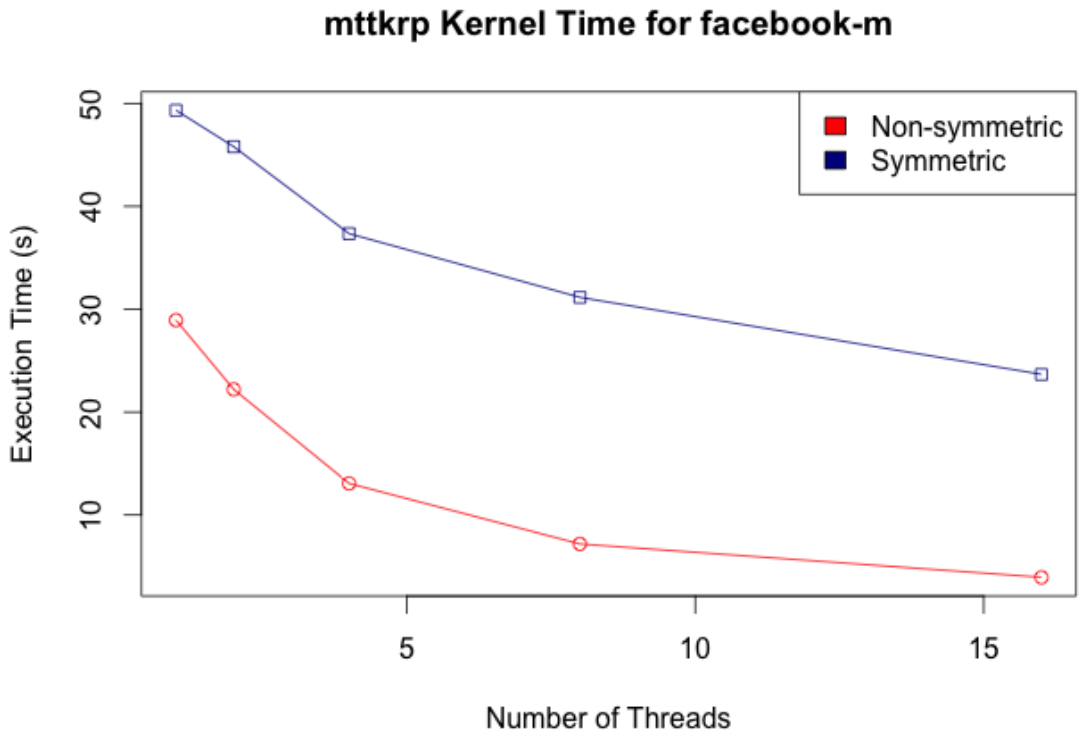
Tensor	Size	Rank
Syn-500	500 x 500 x 300	6



- 1.96x for 2 threads, 4.64x for 16 threads, non-symmetric
- 1.31x for 2 threads, 7.49x for 16 threads, symmetric

Real Tensor Performance Evaluation

Tensor	Size	Rank
facebook-m	500 x 500 x 300	6



Presentation Roadmap

Background

Tensor Terminology

Optimization and Parallelization Techniques

Performance Evaluation

Summary & Forward Work

Summary & Forward Work

What we did

- Developed techniques to effectively parallelize and optimize symmetric tensor computations
  - Focused on the common matricized tensor times Khatri Rao product (mttkrp) operation,
  - Demonstrated performance improvements on synthetic and real-world data sets

What we plan to do

- Automate optimizations using R-Stream compiler