A Medium-Grained Algorithm for Distributed Sparse Tensor Factorization

Shaden Smith George Karypis

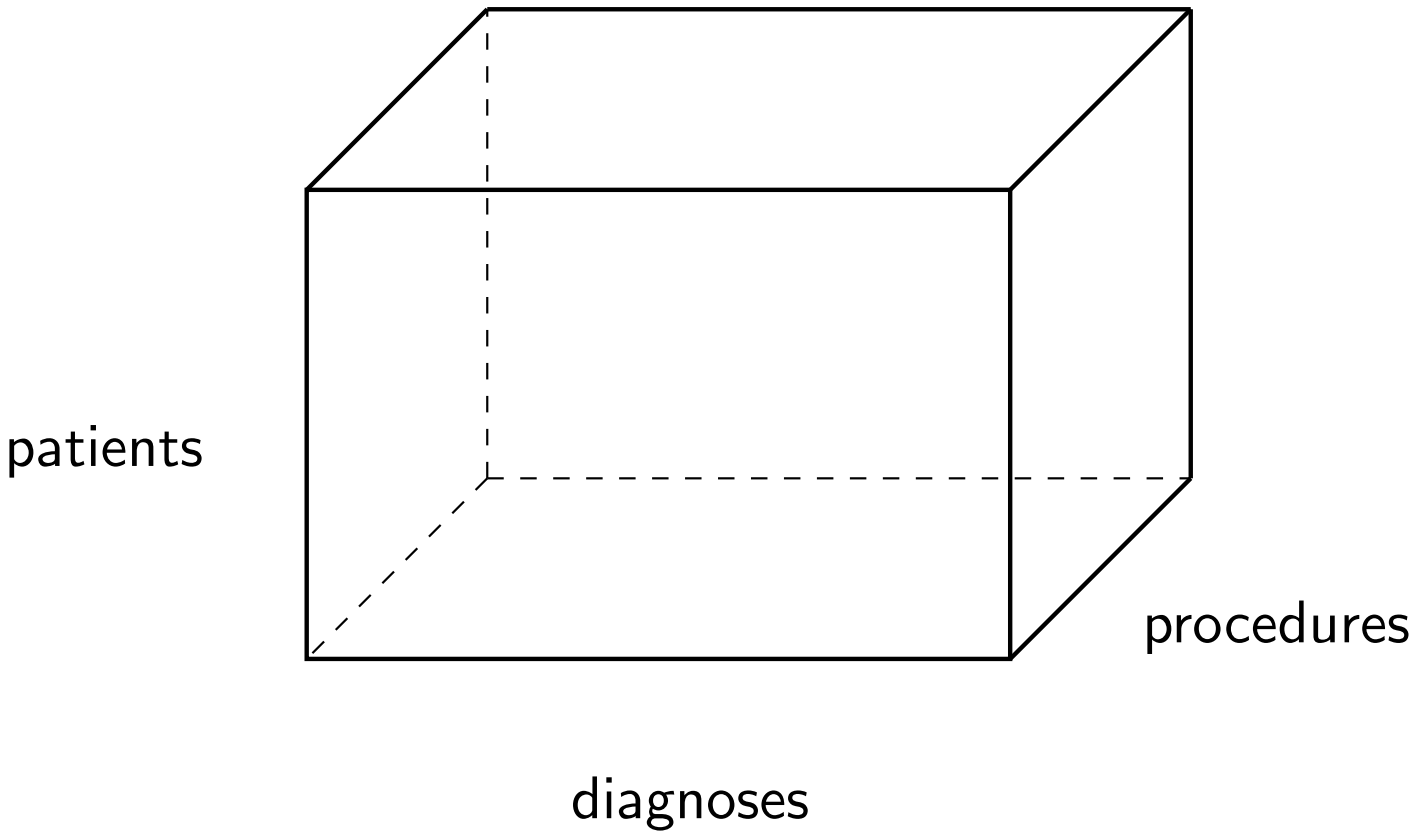
University of Minnesota
Department of Computer Science & Engineering
shaden@cs.umn.edu

Table of Contents

1. Preliminaries
2. Related Work: Coarse- and Fine-Grained Algorithms
3. A Medium-Grained Algorithm
4. Experiments
5. Conclusions

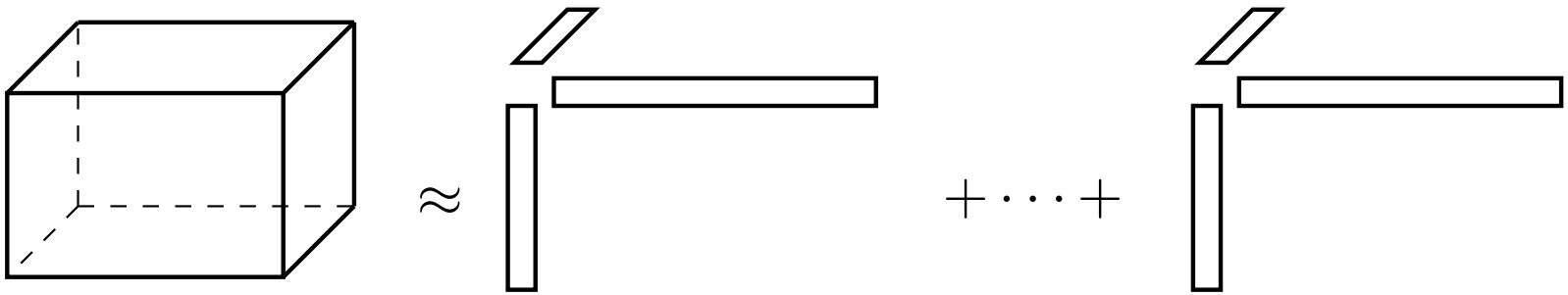
Tensor Introduction

- Tensors are the generalization of matrices to $\geq 3D$
- Tensors have m dimensions (or *modes*) and are $I_1 \times \ldots \times I_m$.
  - We'll stick to $m = 3$ in this talk and call dimensions I, J, K



Canonical Polyadic Decomposition (CPD)

- We compute matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, each with F columns
  - F is assumed to be small, on the order of 10 or 50



- Usually computed via *alternating least squares* (ALS)
- As a result, computations are *mode-centric*

CPD-ALS

Algorithm 1 CPD-ALS

1: **while** not converged **do**
2: $\quad \mathbf{A}^\intercal = \left(\mathbf{C}^\intercal\mathbf{C} * \mathbf{B}^\intercal\mathbf{B}\right)^{-1}\left(\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})\right)^\intercal$
3: $\quad \mathbf{B}^\intercal = \left(\mathbf{C}^\intercal\mathbf{C} * \mathbf{A}^\intercal\mathbf{A}\right)^{-1}\left(\mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A})\right)^\intercal$
4: $\quad \mathbf{C}^\intercal = \left(\mathbf{B}^\intercal\mathbf{B} * \mathbf{A}^\intercal\mathbf{A}\right)^{-1}\left(\mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A})\right)^\intercal$
5: **end while**

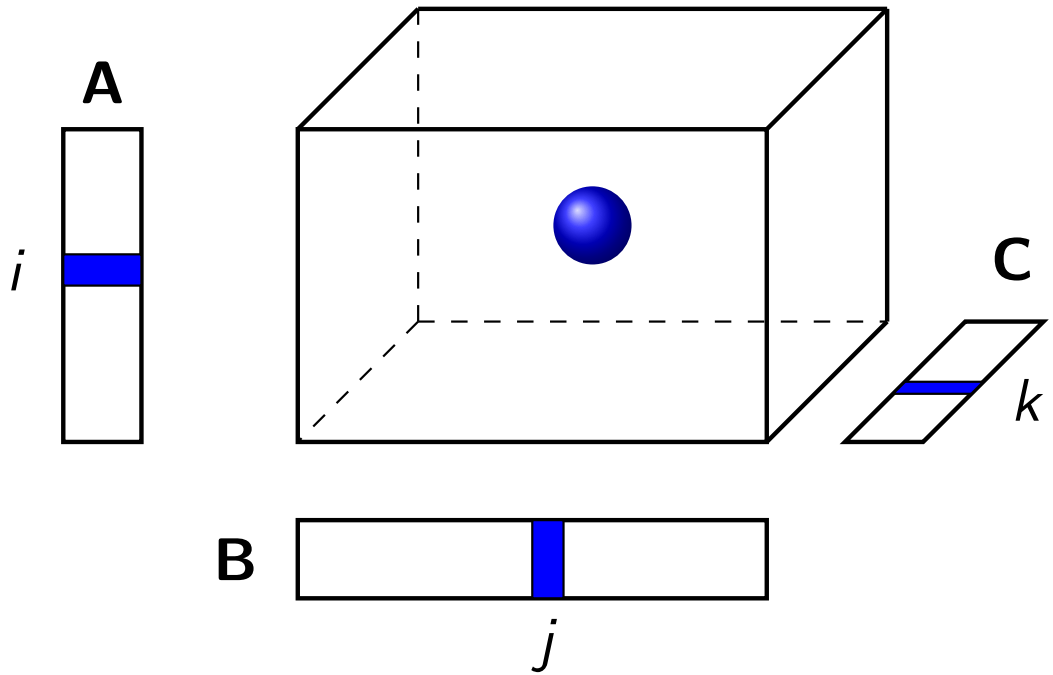
A Closer Look...

Algorithm 2 One mode of CPD-ALS

1: $\hat{\mathbf{A}} \leftarrow \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})$ ▷ $\mathcal{O}(F \cdot \text{nnz}(\mathcal{X}))$
2: $\mathbf{L}\mathbf{L}^\mathsf{T} \leftarrow \text{Cholesky}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{B}^\mathsf{T}\mathbf{B})$ ▷ $\mathcal{O}(F^3)$
3: $\mathbf{A}^\mathsf{T} = (\mathbf{L}\mathbf{L}^\mathsf{T})^{-1}\hat{\mathbf{A}}^\mathsf{T}$ ▷ $\mathcal{O}(IF^2)$
4: Compute $\mathbf{A}^\mathsf{T}\mathbf{A}$ ▷ $\mathcal{O}(IF^2)$

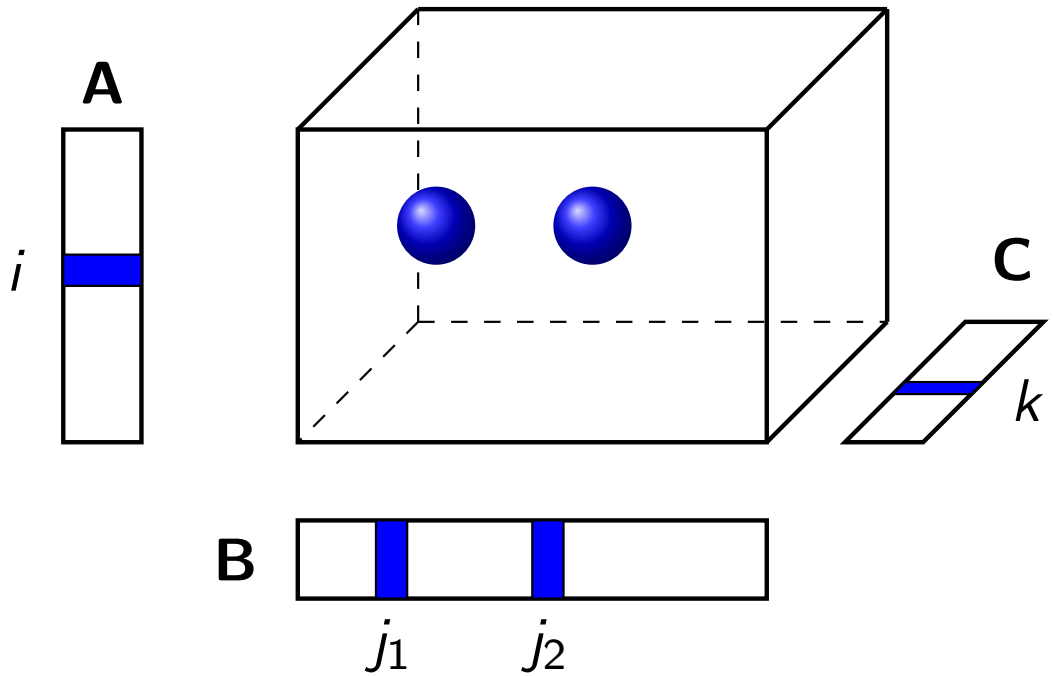
- Step 1 is the most expensive and the focus of this talk

Matricized Tensor Times Khatri-Rao Product (MTTKRP)



$$\hat{\mathbf{A}}(i,:) \leftarrow \hat{\mathbf{A}}(i,:) + \mathcal{X}(i,j,k)\,[\mathbf{B}(j,:) * \mathbf{C}(k,:)]$$

MTTKRP Communication



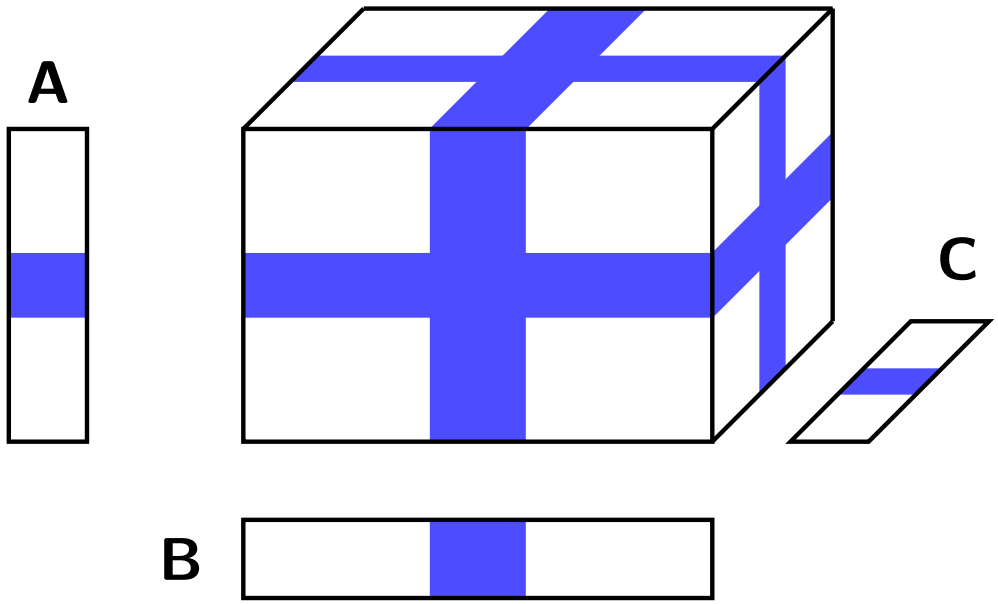
$$\hat{\mathbf{A}}(i,:) \leftarrow \hat{\mathbf{A}}(i,:) + \mathcal{X}(i,j_1,k)\,[\mathbf{B}(j_1,:) * \mathbf{C}(k,:)]$$

$$\hat{\mathbf{A}}(i,:) \leftarrow \hat{\mathbf{A}}(i,:) + \mathcal{X}(i,j_2,k)\,[\mathbf{B}(j_2,:) * \mathbf{C}(k,:)]$$

Table of Contents

1. Preliminaries
2. Related Work: Coarse- and Fine-Grained Algorithms
3. A Medium-Grained Algorithm
4. Experiments
5. Conclusions

Coarse-Grained Decomposition



[Choi & Vishwanathan 2014, Shin & Kang 2014]

- Processes own complete slices of $\mathcal{X}$ and aligned factor rows
- I/p rows communicated to $p-1$ processes after each update

Fine-Grained Decomposition

[Kaya & Uçar 2015]

- Most flexible: non-zeros individually assigned to processes
- Two communication steps
  1. Aggregate partial computations after MTTKRP
  2. Exchange new factor values
- Factors can be assigned to minimize communication

Finding a Fine-Grained Decomposition

Some options:

- Random assignment
- Hypergraph partitioning
- Multi-constraint hypergraph partitioning

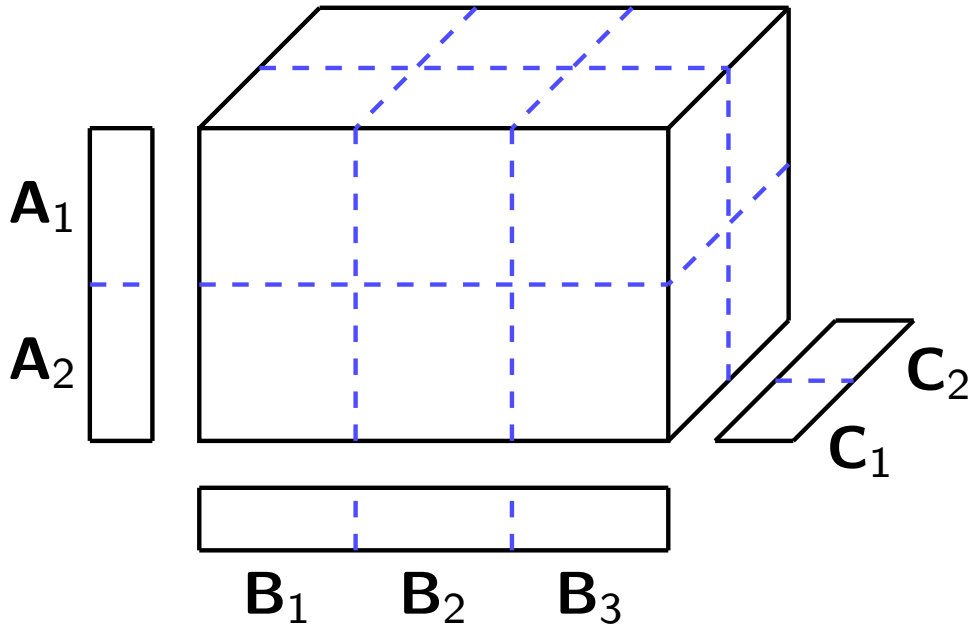
In Practice: Hypergraph Model

- nnz($\mathcal{X}$) vertices and $I+J+K$ hyperedges
- Tight approximation of communication and load balance
  - Distribution of factors must be considered: in practice a greedy solution works well

Table of Contents

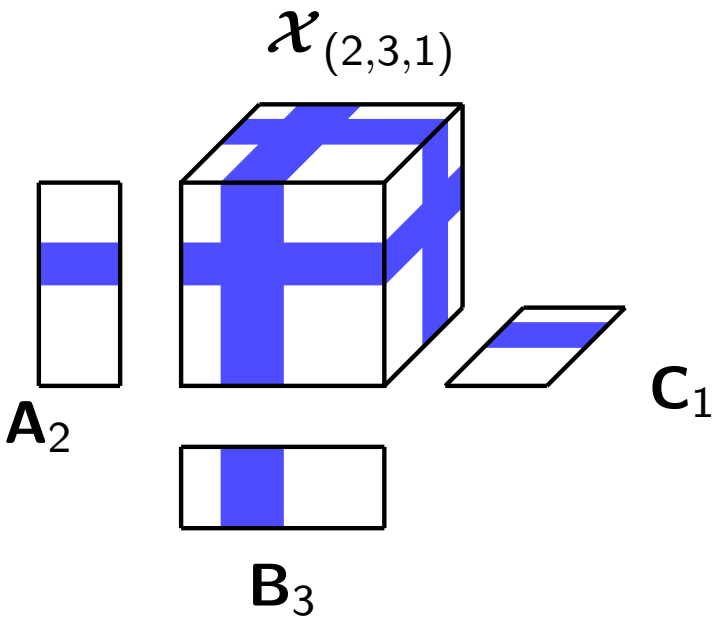
1. Preliminaries
2. Related Work: Coarse- and Fine-Grained Algorithms
3. A Medium-Grained Algorithm
4. Experiments
5. Conclusions

Medium-Grained Decomposition



- Distribute over a grid of $p = q \times r \times s$ partitions
- $r \times s$ processes divide each $\mathbf{A}_1, \ldots, \mathbf{A}_q$
- Two communication steps like fine-grained
  - $\mathcal{O}(I/p)$ rows communicated to $r \times s$ processes

Medium-Grained Decomposition



- Each process owns roughly I/p rows of each factor
- Like before, a greedy algorithm works well

Finding a Medium-Grained Decomposition

Greedy Algorithm

1. Apply a random relabeling to modes of $\mathcal{X}$
2. Choose a decomposition dimension (algorithm in paper)
3. Compute 1D partitionings of each mode
   - Greedily chosen with load balance objective
4. Intersect!
5. Distribute factors with objective of reducing communication

Table of Contents

1. Preliminaries
2. Related Work: Coarse- and Fine-Grained Algorithms
3. A Medium-Grained Algorithm
4. Experiments
5. Conclusions

Datasets

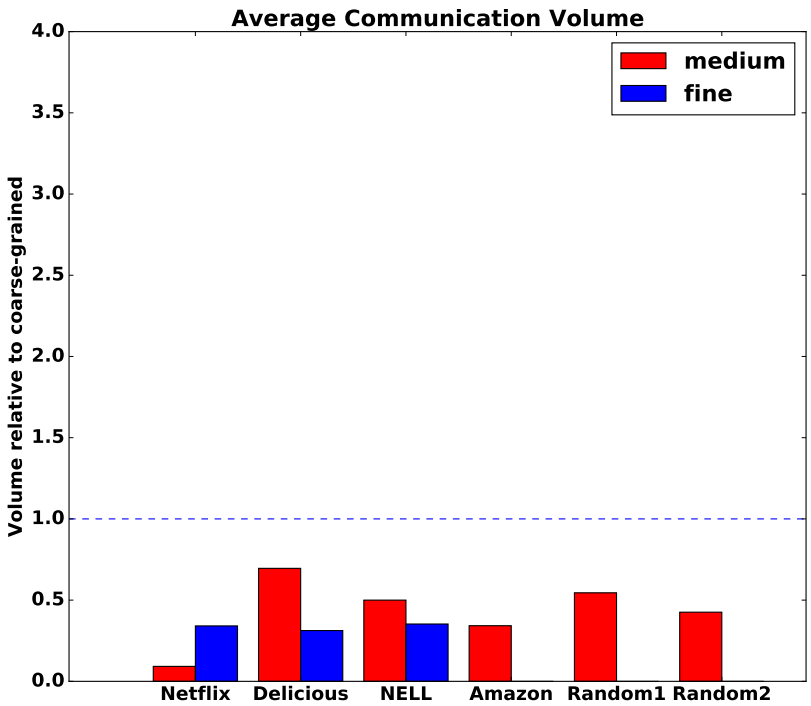
Dataset	I	J	K	nnz
Netflix	480K	18K	2K	100M
Delicious	532K	17M	3M	140M
NELL	3M	2M	25M	143M
Amazon	5M	18M	2M	1.7B
Random1	20M	20M	20M	1.0B
Random2	50M	5M	5M	1.0B

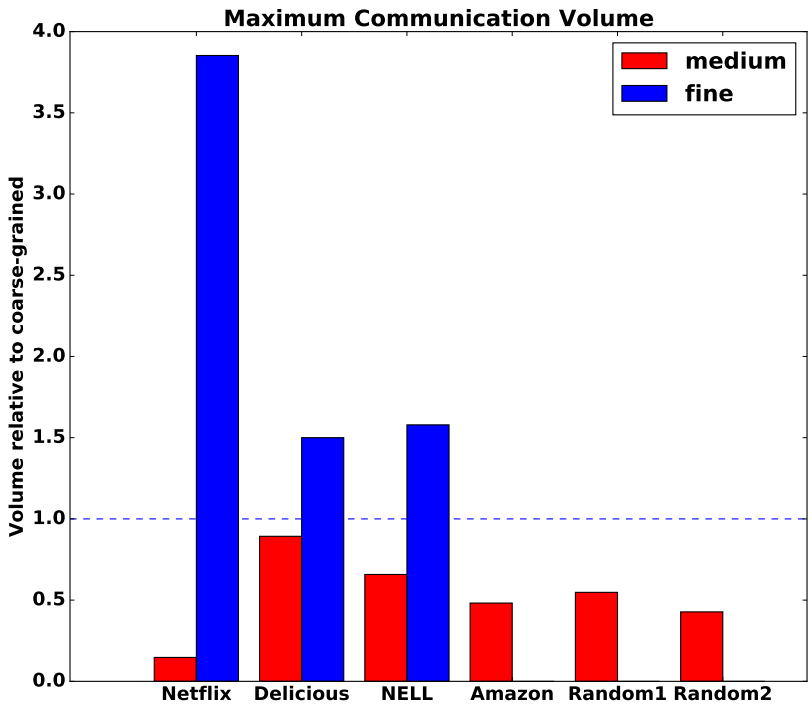
Load Balance

Table: Load imbalance with 64 and 128 processes.

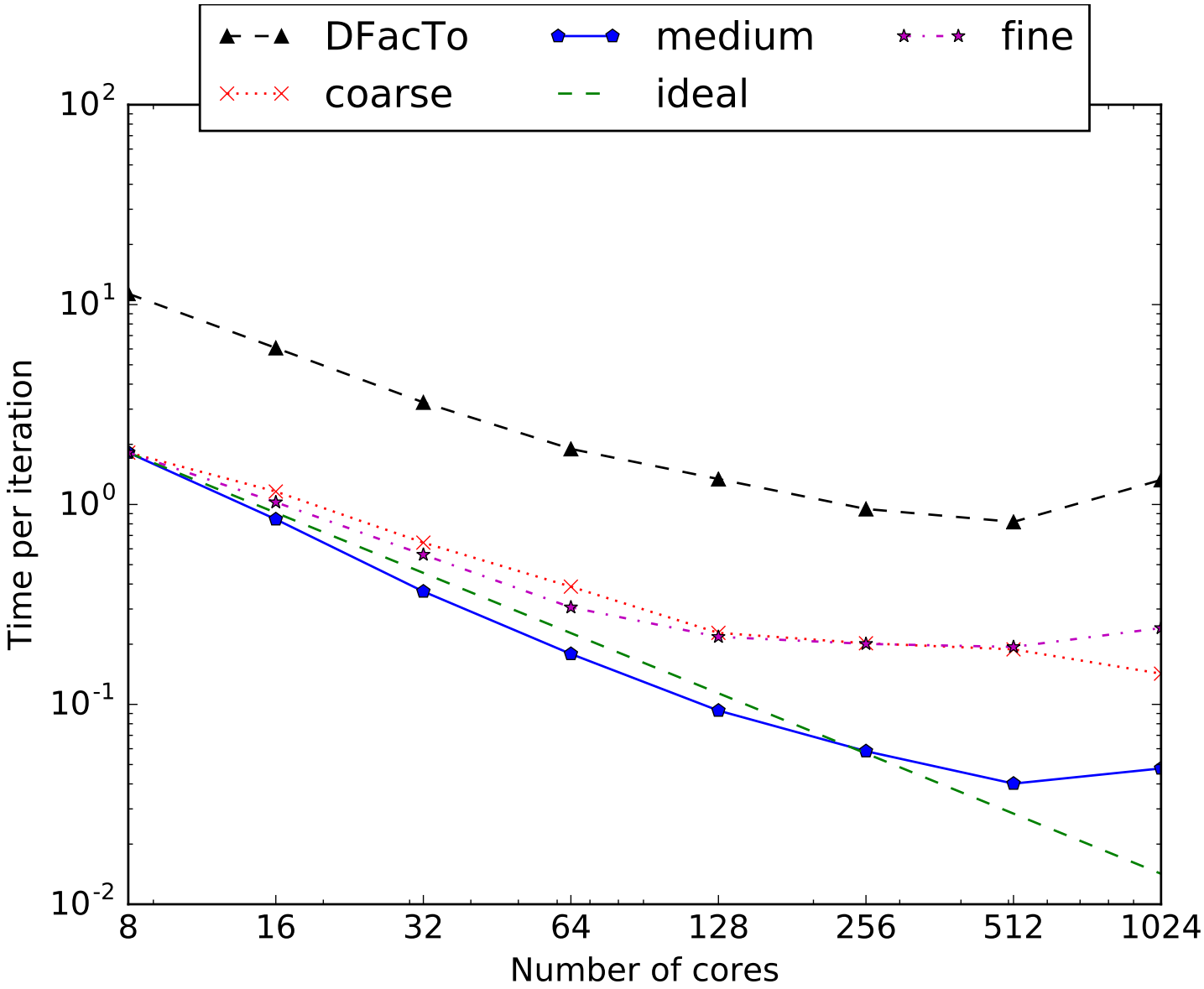
Dataset	coarse		medium		fine	
	64	**128**	**64**	**128**	**64**	**128**
Netflix	1.03	1.18	1.00	1.00	1.00	1.00
Delicious	1.21	1.41	1.01	1.06	1.00	1.05
NELL	1.12	1.29	1.01	1.01	1.00	1.00
Amazon	2.17	3.86	1.08	1.08	-	-

Communication Volume





Strong Scaling: Netflix



Strong Scaling: Amazon

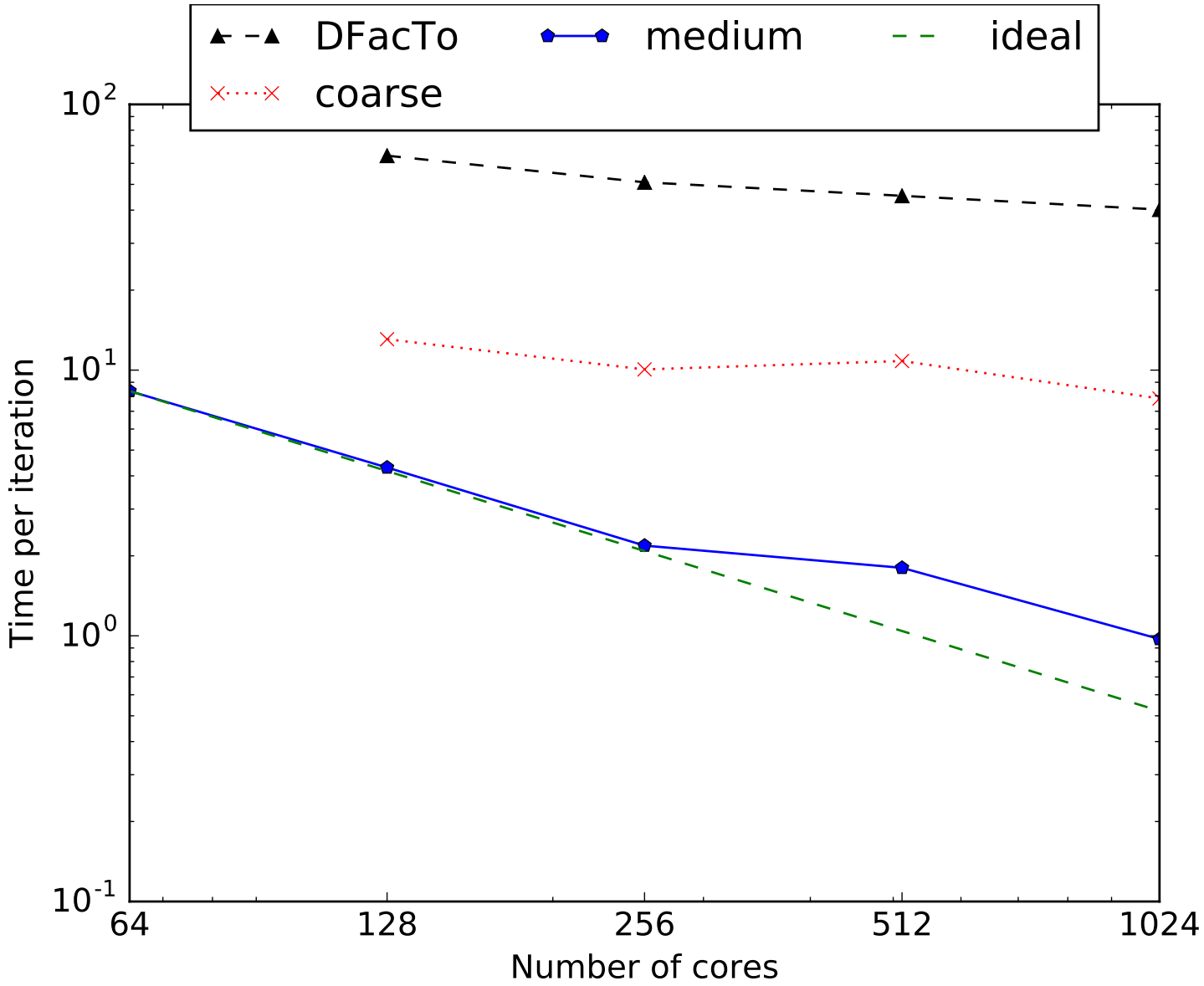


Table of Contents

1. Preliminaries
2. Related Work: Coarse- and Fine-Grained Algorithms
3. A Medium-Grained Algorithm
4. Experiments
5. Conclusions

Wrapping Up...

- Medium-grained decompositions are a good middle-ground
- 1.5× to 5× faster than fine-grained decompositions with hypergraph partitioning
- DMS is 40× to 80× faster than DFacTo, the fastest publicly available software

`http://cs.umn.edu/~splatt/`

Choosing the Shape of the Decomposition

Objective

- We need to find q, r, s such that $q \times r \times s = p$
- Tensors modes are often very skewed (480k Netflix users vs 2k days)
  - We want to assign processes proportionally
  - 1D decompositions actually work well for many tensors

Algorithm

1. Start with a $1 \times 1 \times 1$ shape
2. Compute the prime factorization of p
3. For each prime factor f, starting from the largest, multiply the most imbalanced mode by f