IA-SpGEMM: an Input-aware Auto-tuning Framework for Parallel Sparse Matrix-Matrix Multiplication

Zhen Xie†, Guangming Tan†, Weifeng Liu § ‡, Ninghui Sun†

†State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences
§ University of Chinese Academy of Sciences
‡Department of Computer Science and Technology, China University of Petroleum, Beijing

ICS ‘19, June 26-28, Phoenix AZ, USA

Faculty of Science

Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

Faculty of Science

Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

Sparse Matrix - Data Storage Format

- If most elements in a matrix are zeros, we can use sparse representations to store the matrix
- Different formats represent different space occupations and memory access sequences

$$A = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 0 & 3 & 4 & 0 \\ 0 & 0 & 5 & 6 \\ 0 & 0 & 0 & 7 \end{bmatrix}$$

Compressed Sparse Row (CSR)

$$\begin{aligned} \mathrm{ptr} &= [0\ 2\ 4\ 6\ 7] \\ \mathrm{col\_ind} &= [0\ 1\ 1\ 2\ 2\ 3\ 3] \\ \mathrm{data} &= [1\ 2\ 3\ 4\ 5\ 6\ 7] \end{aligned}$$

Coordinate (COO)

$$\begin{aligned} ptr &= [0\ 2\ 4\ 6\ 7] \\ \mathrm{rows} &= [0\ 0\ 1\ 1\ 2\ 2\ 3] \\ \mathrm{cols} &= [0\ 1\ 1\ 2\ 2\ 3\ 3] \\ \mathrm{data} &= [1\ 2\ 3\ 4\ 5\ 6\ 7] \end{aligned}$$

Diagonal (DIA)

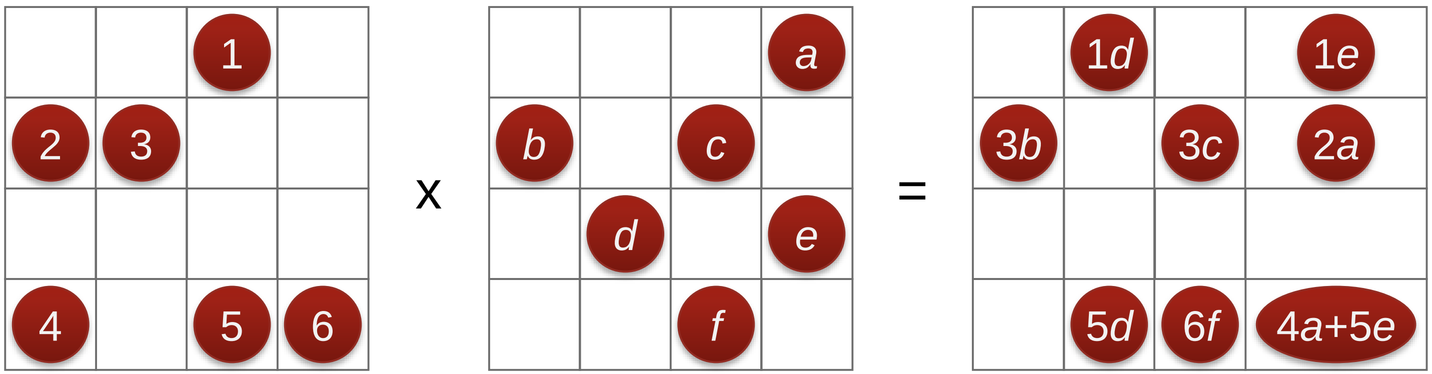
$$\begin{aligned} pos &= [-1\ -1\ -1\ 0\ 1\ -1\ -1] \\ \mathrm{offsets} &= [0\ 1] \\ \mathrm{data} &= [1\ 3\ 5\ 7\ 2\ 4\ 6\ *] \end{aligned}$$

ELLPACK (ELL)

$$\begin{aligned} nnz &= [2\ 2\ 2\ 1] \\ \mathrm{col\_ind} &= [0\ 1\ 1\ 2\ 2\ 3\ 3\ *] \\ \mathrm{data} &= [1\ 2\ 3\ 4\ 5\ 6\ 7\ *] \end{aligned}$$

Sparse GEMM (SpGEMM) - Basics

- Multiply a sparse matrix *A* by a sparse matrix *B*, obtain a result sparse matrix *C*.



A	*B*	*C*
(4x4)	(4x4)	(4x4)
nnzA = 6	*nnzB* = 6	*nnzC* = 8

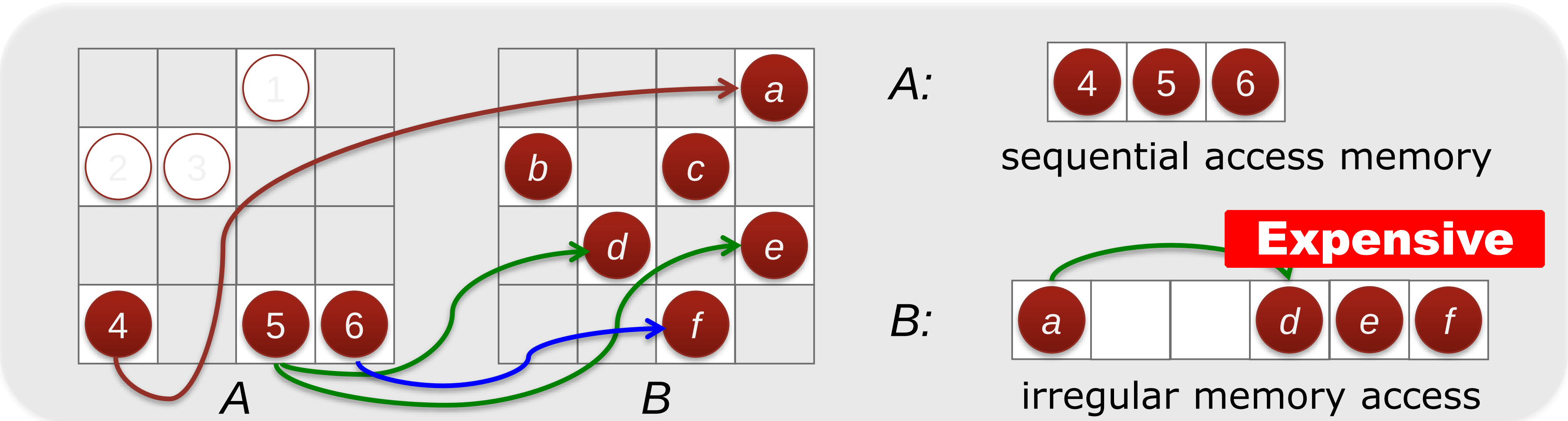
Faculty of Science

Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

SpGEMM Overhead 1 – Memory access

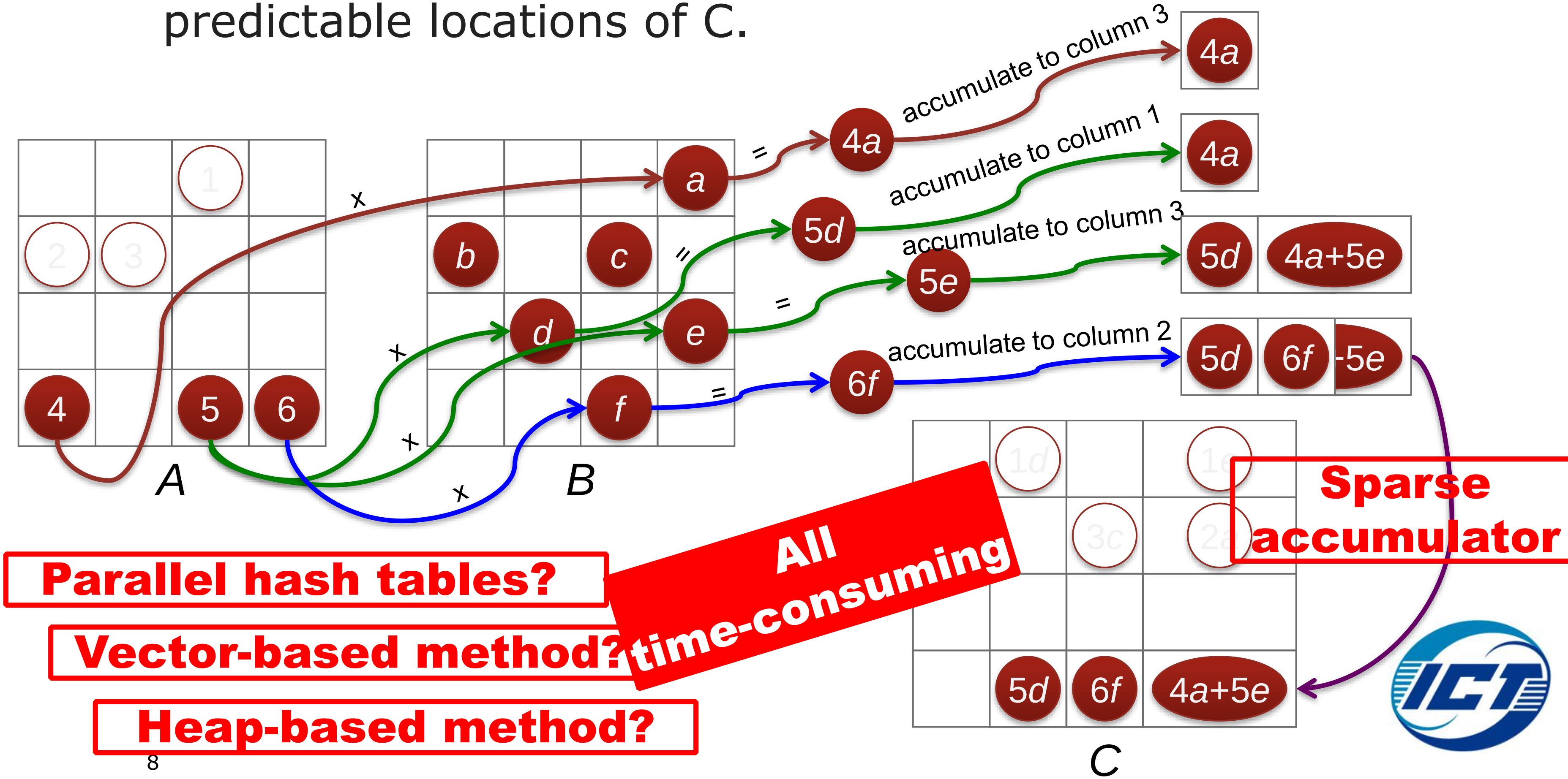
- Due to the *sparsity of A*, a large number of *irregular memory access occur*.



- *The memory access sequences of A & B using different formats are* hugely different.

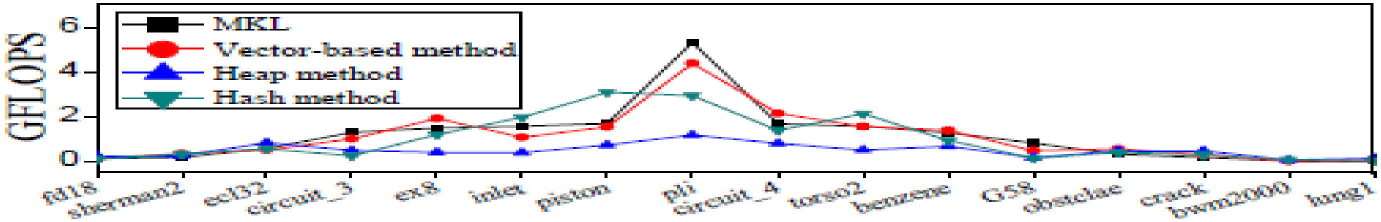
SpGEMM Overhead 2 – Sparse accumulation

- Because of the compressed sparse format, the result nonzeros are “accumulated” into C, but not “added” to predictable locations of C.

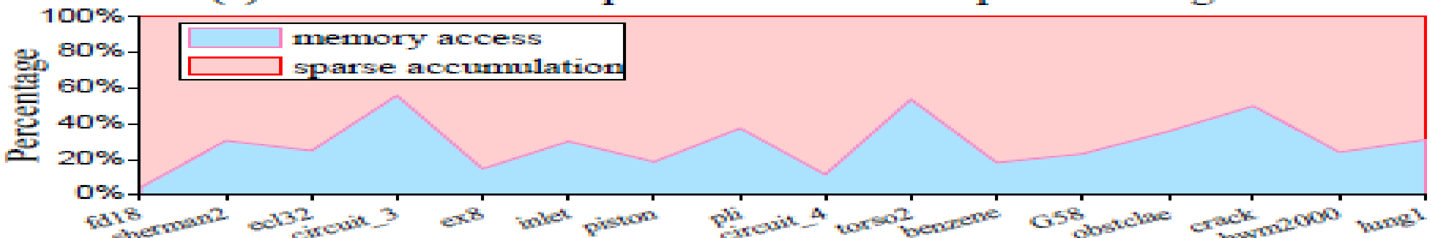


Three motivations

- These libraries are sensitive to the input sparse matrices and show huge performance differences.
- To some extent, SpGEMM is similar to sparse matrix-vector multiplication (SpMV) for irregular and indirect memory access pattern, many auto-tunning on SpMV have been dedicated. (OSKI,SPARITY)
- Memory access still occupies a certain amount of execution time.



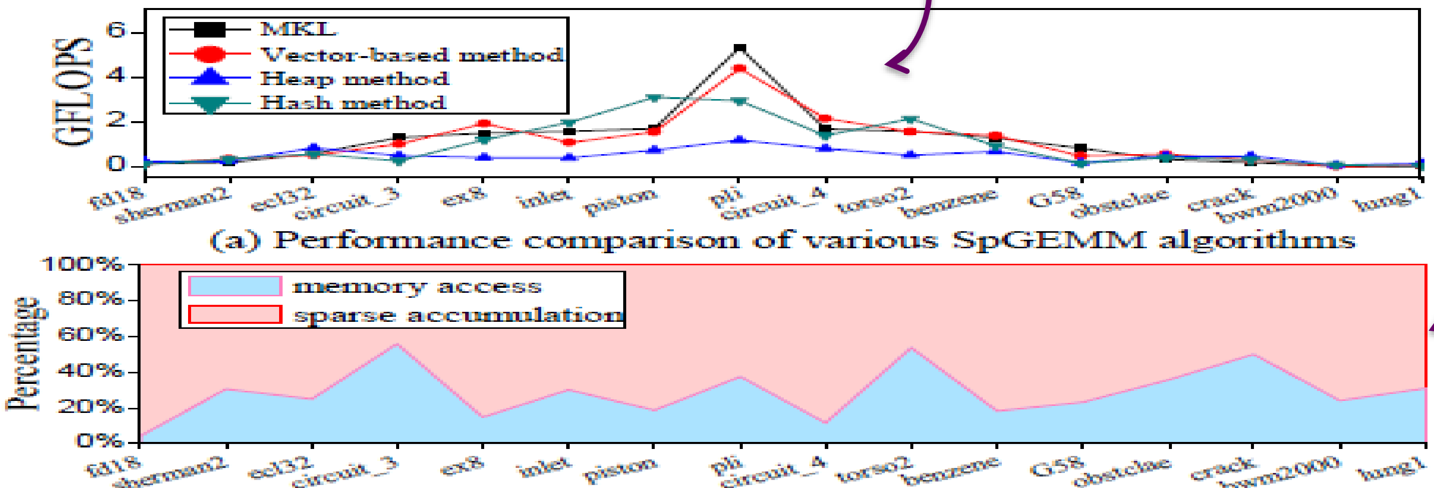
(a) Performance comparison of various SpGEMM algorithms



(b) Time overhead of memory access and sparse accumulation

Two solutions

- Reducing memory requirements or accelerating memory access on vector architecture by using classic storage formats, such as DIA, COO and ELL.
- Developing an autotuning framework to determine the optimal SpGEMM algorithm.



(a) Performance comparison of various SpGEMM algorithms

(b) Time overhead of memory access and sparse accumulation

Faculty of Science

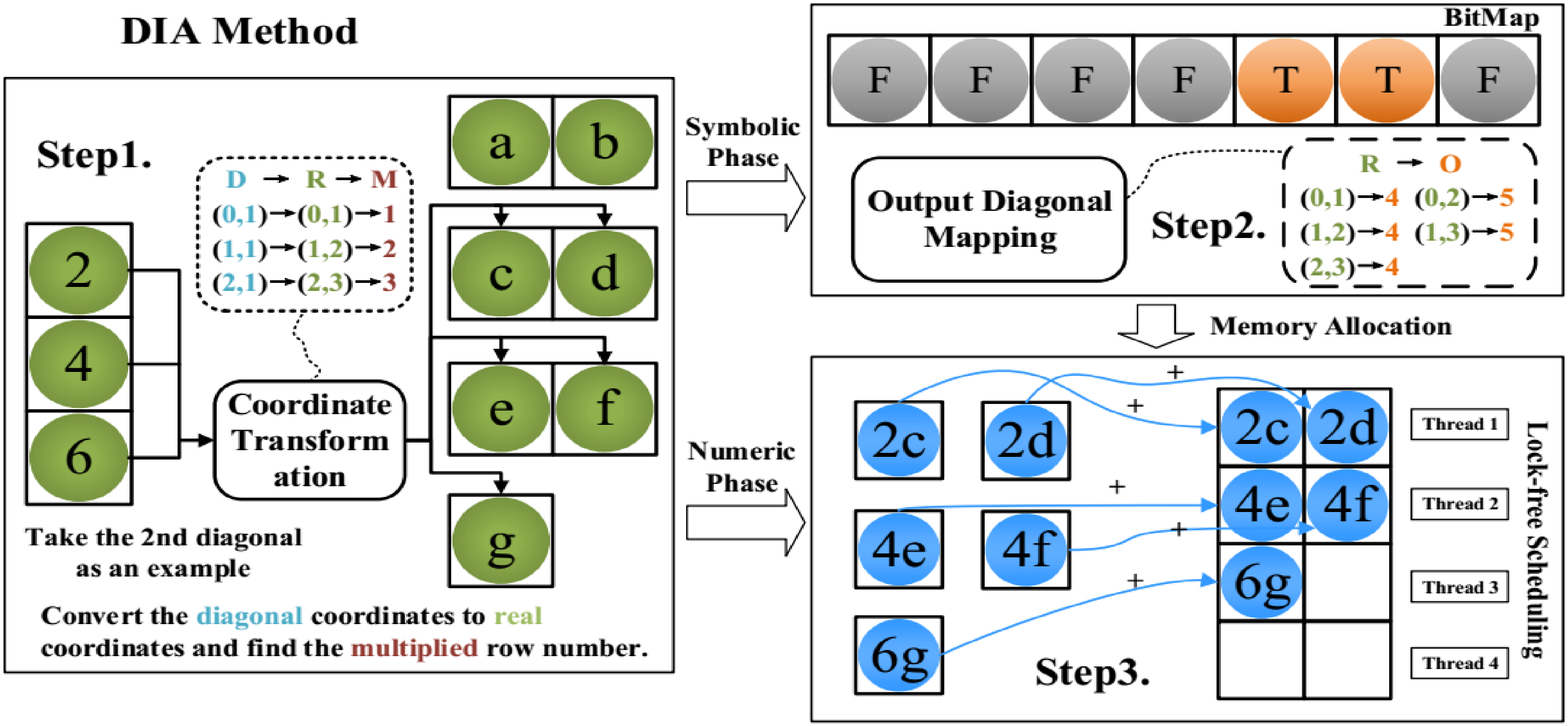
Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

Three format-special SpGEMM algorithms

DIA method:

- Step 1: coordinate transformation before multiplication.
- Step 2: output mapping after multiplication.



Analysis of DIA method

Pros:

- Greatly reduces the overhead of memory access for diagonal matrix.
- Directly accumulates intermediate results to the target address without extra memory consumption.

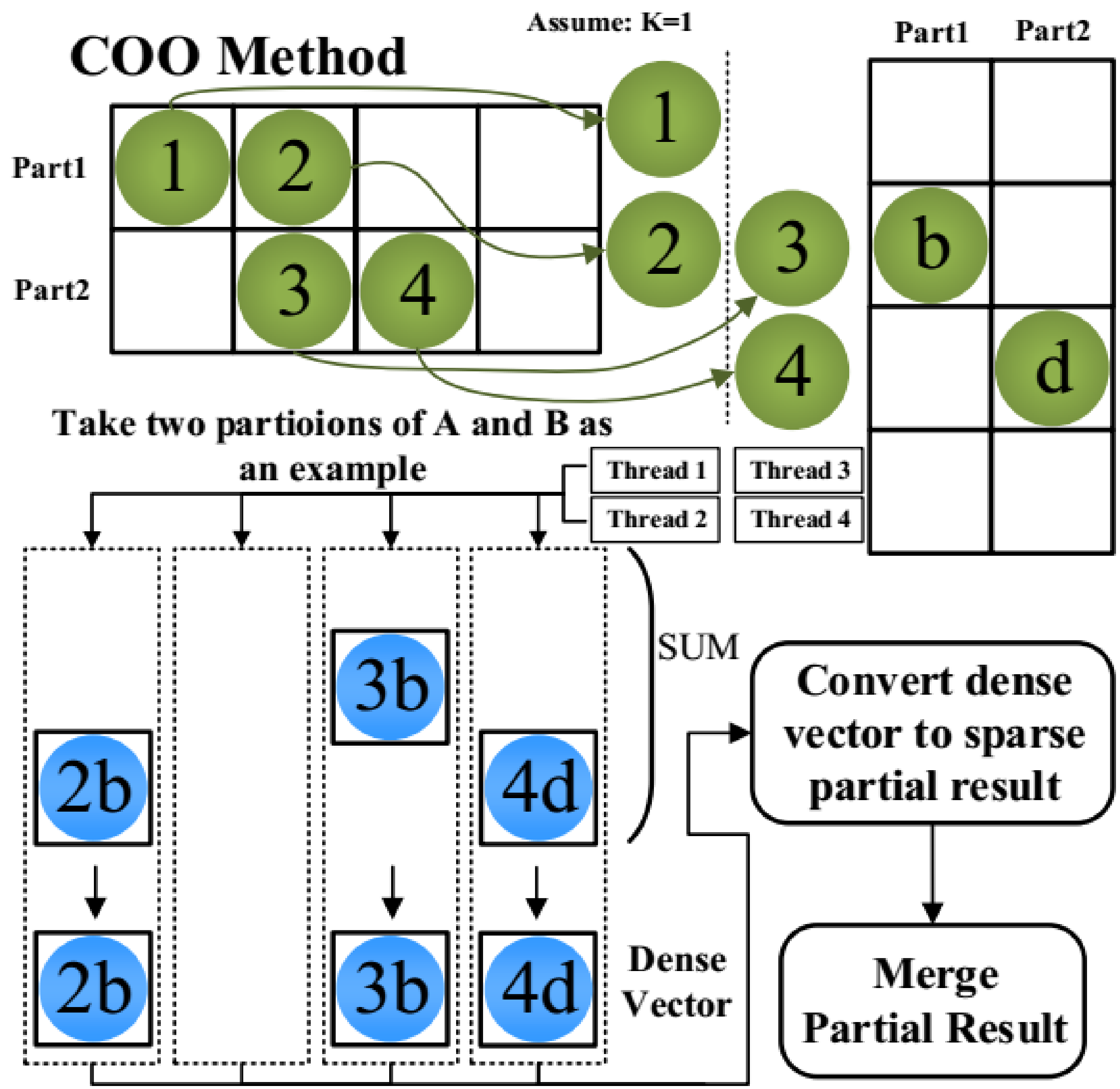
Cons:

- Extra format conversion overhead.
- Not suitable for non-diagonal matrices.

Three format-special SpGEMM algorithms

COO method:

- Step 1: divides matrix A into k parts by row and matrix B into k parts by column before multiplication.
- Step 2: all the partial results are merged after multiplication.



Analysis of COO method

Pros:

- It greatly reduces the length of dense vector than that of the SPA method.

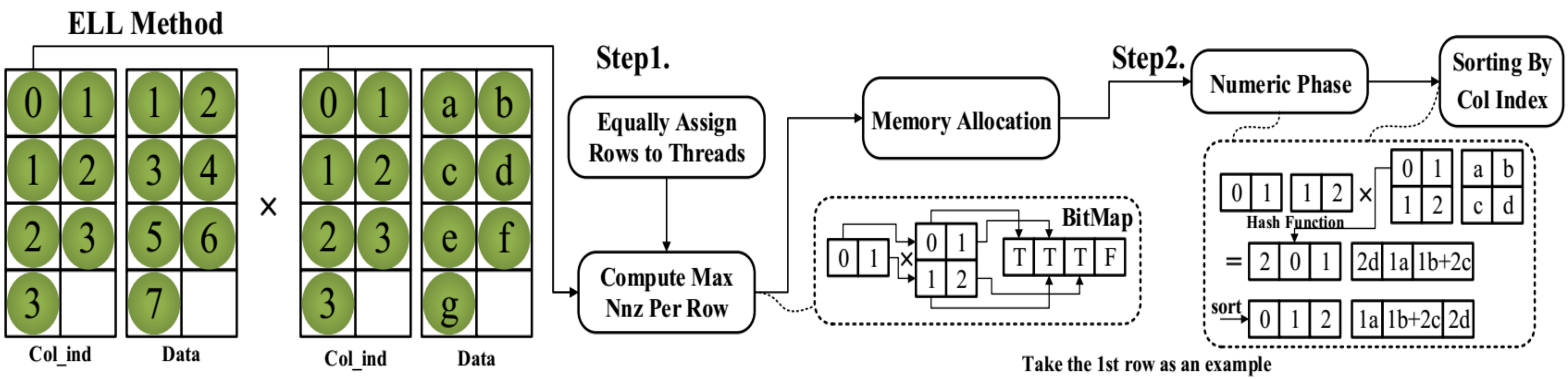
Cons:

- It brings additional overhead for partitioning and merging matrices.

Three format-special SpGEMM algorithms

ELL method:

- Step 1: Since each line of ELL format contains same non-zero number, so this format makes it possible for reducing the overhead of symbol phase.
- Step 2: The allocated memory space of C is used as hash table to accumulate intermediate results.



Analysis of ELL method

Pros:

- The symbolic phase can make full use of the SIMD instructions to speed up the efficiency of loading and assigning data.
- In numeric phase, the allocated memory space of C benefits the advantage of hash table without memory consumption.

Cons:

- Need to strictly meet the ELL format requirements.

Performance improvements

- Observation 1: Huge performance differences by various inputs, formats, algorithms and platforms.
- Observation 2: No single algorithm can constantly contribute the best performance
- Observation 3: It is difficult to obtain high prediction accuracy by using traditional machine learning algorithm.

	Method	Dominance		Percentage		Average Speedup	Speedup by "Ideal Tool"
		Best of all	Over BL.	Best of all	Over BL.		
Intel CPU	MKL (Baseline)	2874	-	35.07%	-	-	8.94x
	DIA method	491	1107	5.99%	13.51%	72.04x	
	COO method	283	506	3.45%	6.17%	7.63x	
	ELL method	1496	2879	18.26%	35.13%	9.92x	
	SPA vector-based	259	748	3.16%	9.13%	1.31x	
	Hash-based	2150	4307	26.24%	52.56%	6.37x	
	Heap-based	642	1951	7.83%	23.81%	6.21x	
AMD CPU	MKL (Baseline)	1708	-	20.96%	-	-	46.16x
	DIA method	745	1544	9.14%	18.94%	346.0x	
	COO method	342	586	4.20%	7.19%	8.20x	
	ELL method	1989	3044	24.40%	37.35%	32.96x	
	SPA vector-based	830	2529	10.18%	31.03%	1.58x	
	Hash-based	1757	2363	21.56%	28.99%	21.40x	
	Heap-based	779	1015	9.56%	12.45%	12.18x	
NVIDIA GPU	cuSPARSE(Baseline)	3827	-	51.07%	-	-	2.40x
	CUSP	208	769	2.78%	10.26%	6.27X	
	NSPARSE	3459	3525	46.16%	47.04%	3.71X	

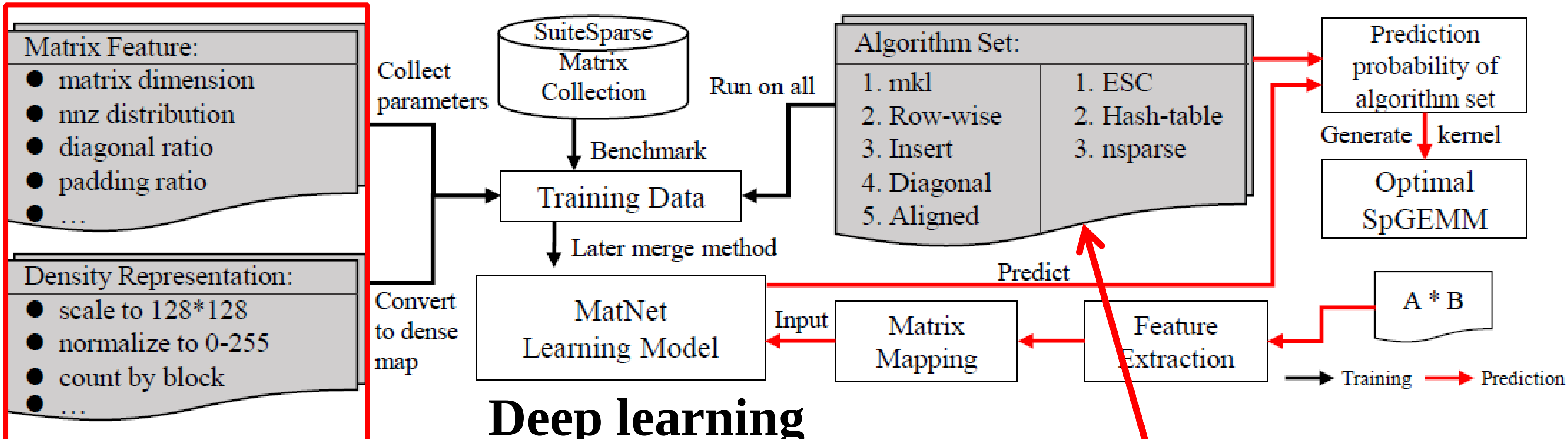
Faculty of Science

Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

IA-SpGEMM overview

The collection phase includes extracting two patterns of input and executing all the algorithms. The training phase generates the MatNet model by two-way strategy.



Deep learning model

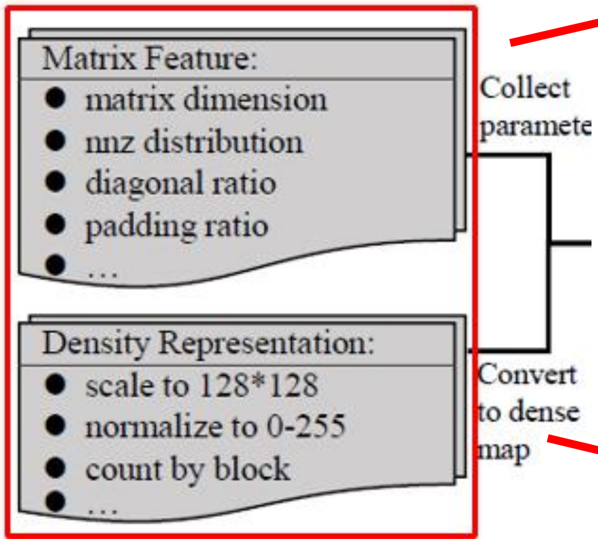
Two respective inputs：
1. fine-grained: sparse features
2. coarse-grained: density representation

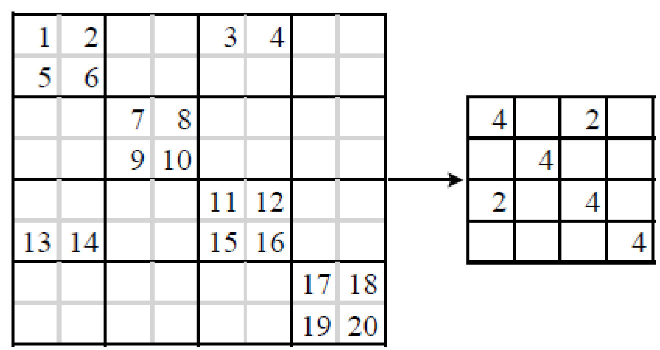
CPU	Intel MKL (CSR)
	Fine tuning row-wise method (CSR)
	Insert method (COO)
	Transform method (DIA)
	Aligned method (ELL)
GPU	Fine tuning ESC method based on CUSP (COO)
	cuSparse (CSR)
	nsparse (CSR)

Input features

Sparse features relate to the distribution and characteristic of four formats.

Feature	Description
row, col, nnz	the number of rows, columns and non-zero elements
nnz_ratio	the ratio of non-zero elements in CSR format
max, min, average	the maximun, minimun and average of non-zero elements
VAR	the variance of non-zero elements
dia_num	the number of diagonals in DIA format
dia_ratio	the number of diagonals divided by all the diagonals
dia_pad, ell_pad	the ratio of padding data in DIA and ELL format
CV	the coefficient of variation of non-zero elements





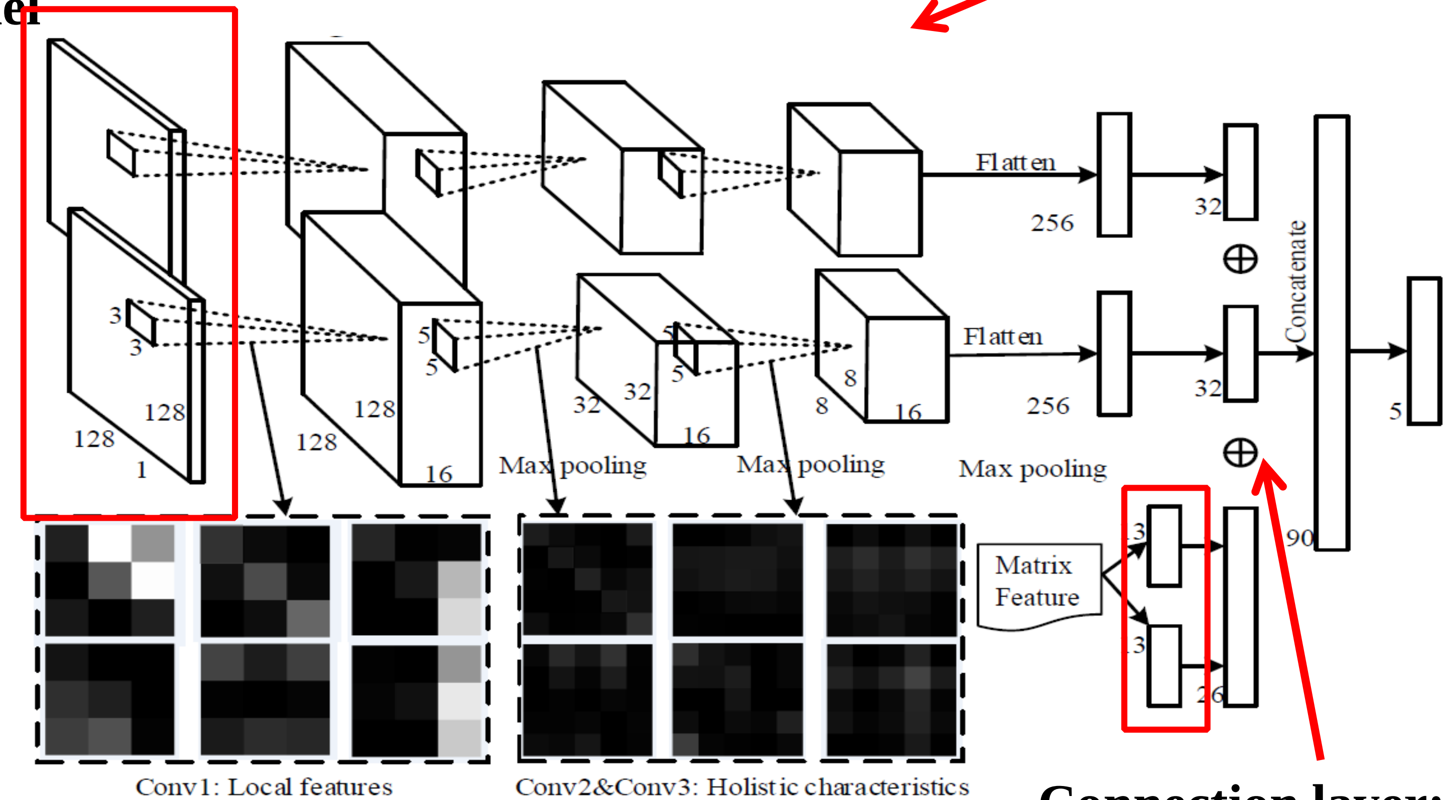
Density representation as primary image input of CNN represents snapshot matrix which abstracts most of matrix features.

MatNet design

Two respective inputs: The red box represents the four input data of the model

MatNet Learning Model

Deep learning model



Output: Predicted probability of various formats and algorithms

Connection layer: Connecting coarse-grained and fine-grained patterns

Faculty of Science

Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

Experiment setup

Platform:

- Intel Xeon E5-2690 v4, 2 processors, 28 cores @2.60 GHz, 35 MB LLC, 2*4 channel memory, 136.6 GB/s Bandwidth. MKL v19.0.
- AMD EPYC 7501, 2 processors, 64 cores@2.00 GHz, 64 MB LLC, 2*8 channel memory, 341 GB/s Bandwidth.
- nVidia Tesla P100, 56 SMs@1328 MHz, 4096 KB L2, 732 GB/s bandwidth. cuSPARSE v8.0.61, CUSP v0.5.1.

Dataset:

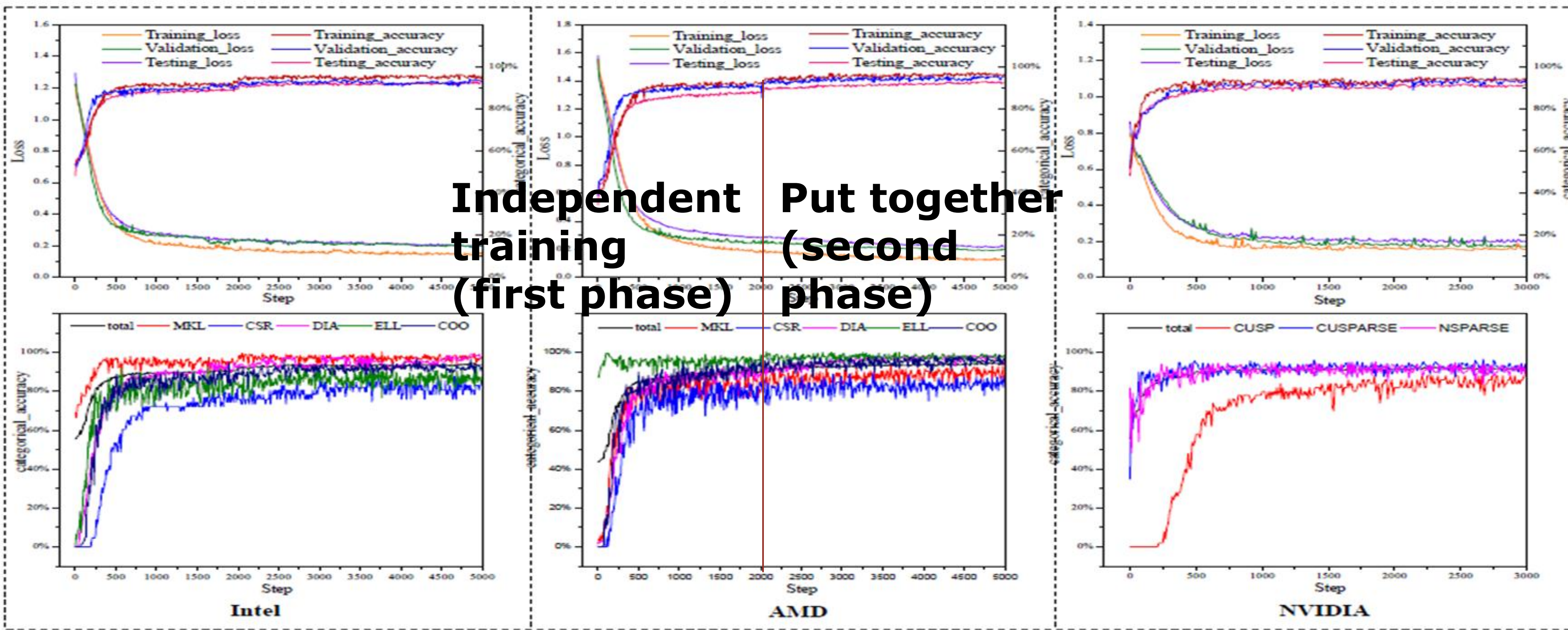
- 2726 matrices from the SuiteSparse matrix collection are used to randomly construct 8195 matrix pairs for evaluation by over 220 GB total size

Baseline:

- CPU: Intel MKL v19.0.0.117 and hash-based method
- GPU: NVIDIA cuSPARSE v8.0.61 and NSPARSE.

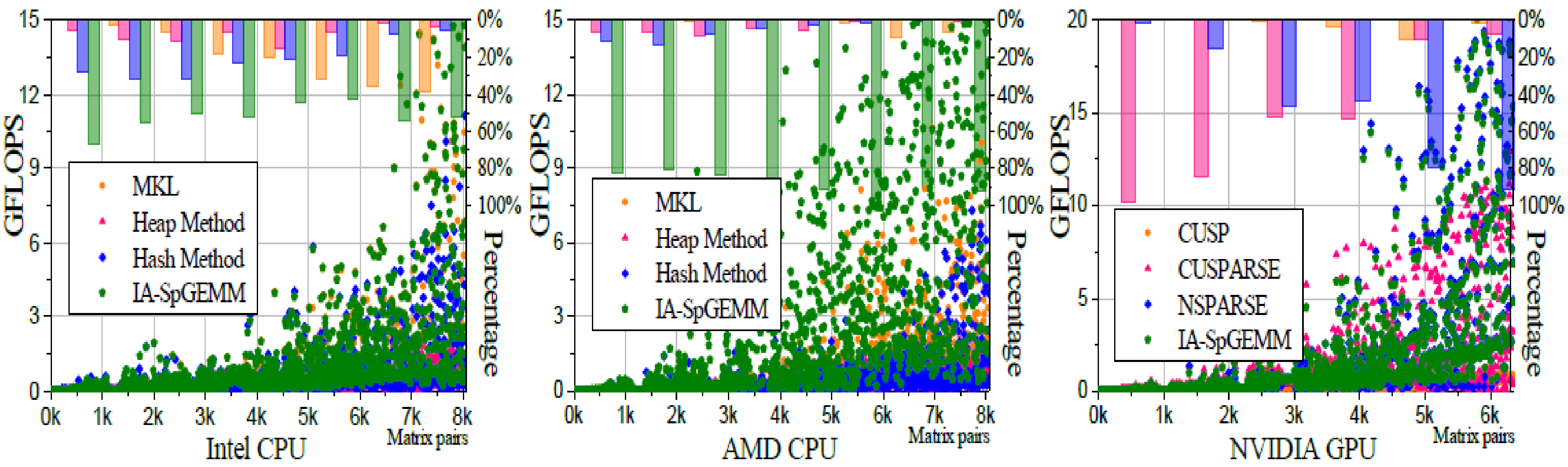
Loss and accuracy

- With the increase of training times, prediction accuracy increases and loss decreases.
- The training process of various formats and algorithms is related to amount of the training data.
- The prediction accuracy of three platforms can reach more than 93%.



Speedup results

- DIA format brings the greatest performance improvement, up to 346.0x
- Overall performance improvement: 3.27x, 13.17x and 2.23x (with overhead)



Faculty of Science

Overview

- Sparse GEMM (SpGEMM) overview
- Two overheads and motivation
- Our SpGEMM algorithms
  - Algorithms: SpGEMM in DIA, COO and ELL formats
  - Performance improvements
- IA-SpGEMM: Input-aware Auto-tuning Framework
  - Input features (sparse features and density representation)
  - MatNet design
- Experimental results
- Conclusion

Conclusion

1. We propose a variety of SpGEMM algorithms for DIA, COO and ELL format, and compare performance of various algorithms.

2. We present an Input-aware Auto-tuning Framework for SpGEMM (IA-SpGEMM), which could automatically determine the best format and algorithm for any sparse matrix pairs.

3. The results show that IA-SpGEMM yields better performance than four other state-of-the-art libraries. And we also expect more sparse and input-sensitive algorithms can be inspired by our method.

Faculty of Science

Thanks!

Any Questions?