中國科學院
CHINESE ACADEMY OF SCIENCES

Register-Aware Optimizations for Parallel Sparse Matrix-Matrix Multiplication

Xin He, Guangming Tan, and Junmin Xiao

2019/02/28

高性能计算机研究中心
HIGH PERFORMANCE COMPUTER RESEARCH CENTER

Overview

- Sparse (SpGEMM) and its applications
- Challenges
- Our sparse accumulator optimization
- Experimental results
- Conclusions

Overview

- Sparse (SpGEMM) and its applications
- Challenges
- Our sparse accumulator optimization
- Experimental results
- Conclusions

Sparse matrix and its storage format

- Sparse matrix is a matrix with lots of zero elements.
- Compressed Sparse Row (CSR) format contains three arrays: (1) row pointer, (2) column index, and (3) value.

0	0	0	*a*
b	0	*c*	0
0	*d*	0	*e*
0	0	*f*	0

B
(4x4)
nnzB = 6

			a
b		*c*	
	d		*e*
		f	

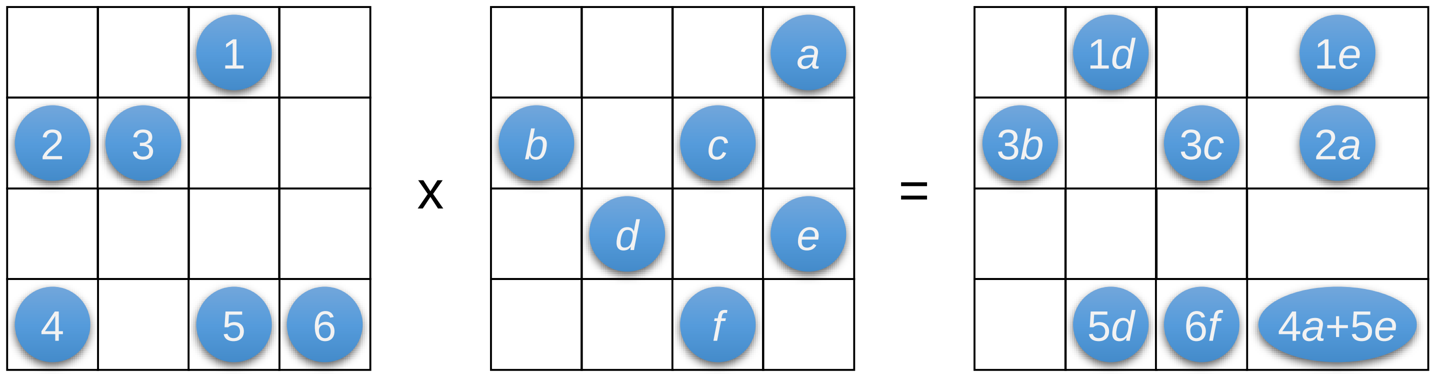
B
(4x4)
nnzB = 6

row_pointer =	0	1	3	5	6	
column_index =	3	0	2	1	3	2
value =	a	b	c	d	e	f

B in CSR-format

Sparse matrix-matrix multiplication

- Two sparse input matrices
- One sparse output matrix



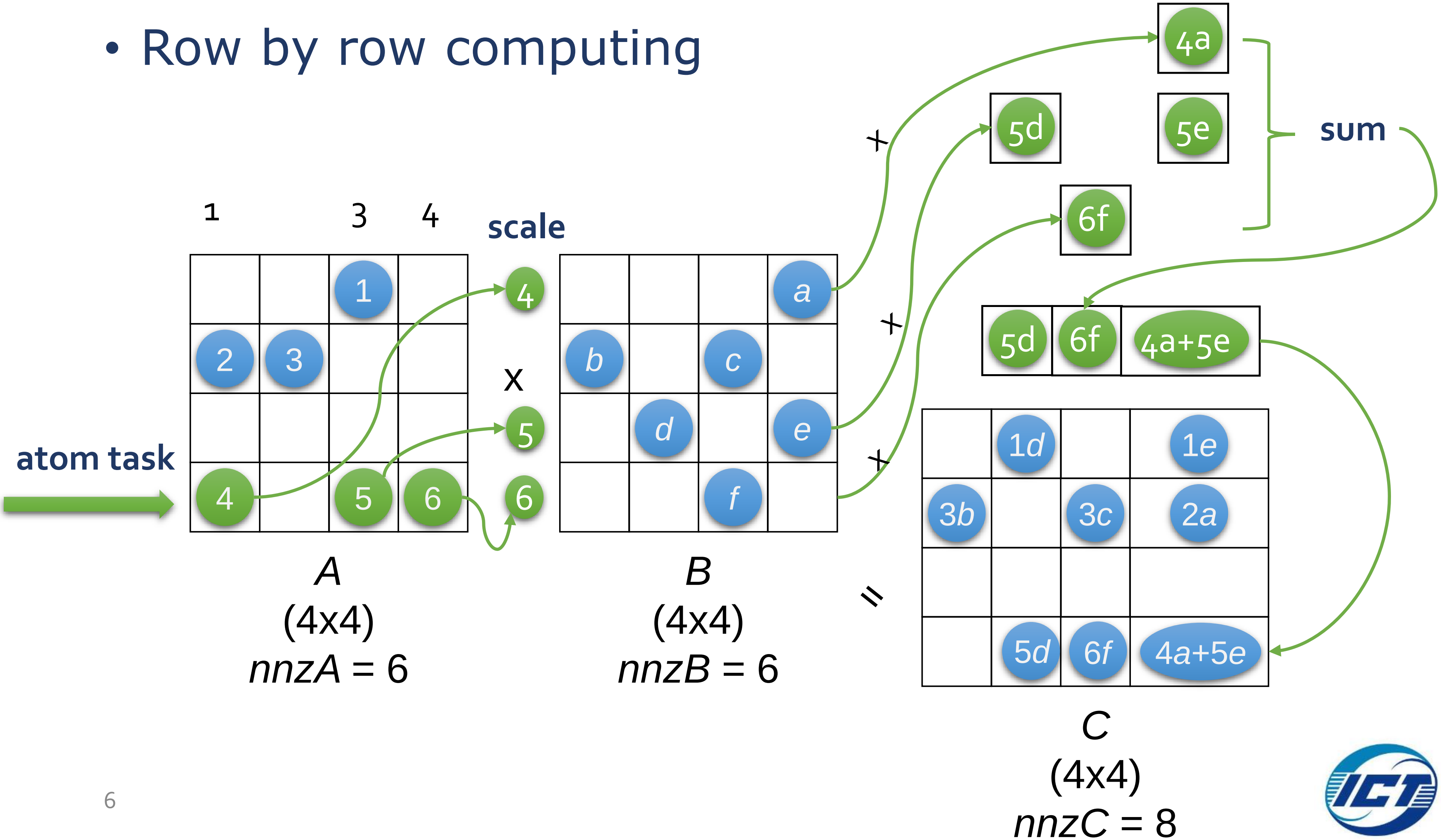
A (4x4) *nnzA* = 6

B (4x4) *nnzB* = 6

C (4x4) *nnzC* = 8

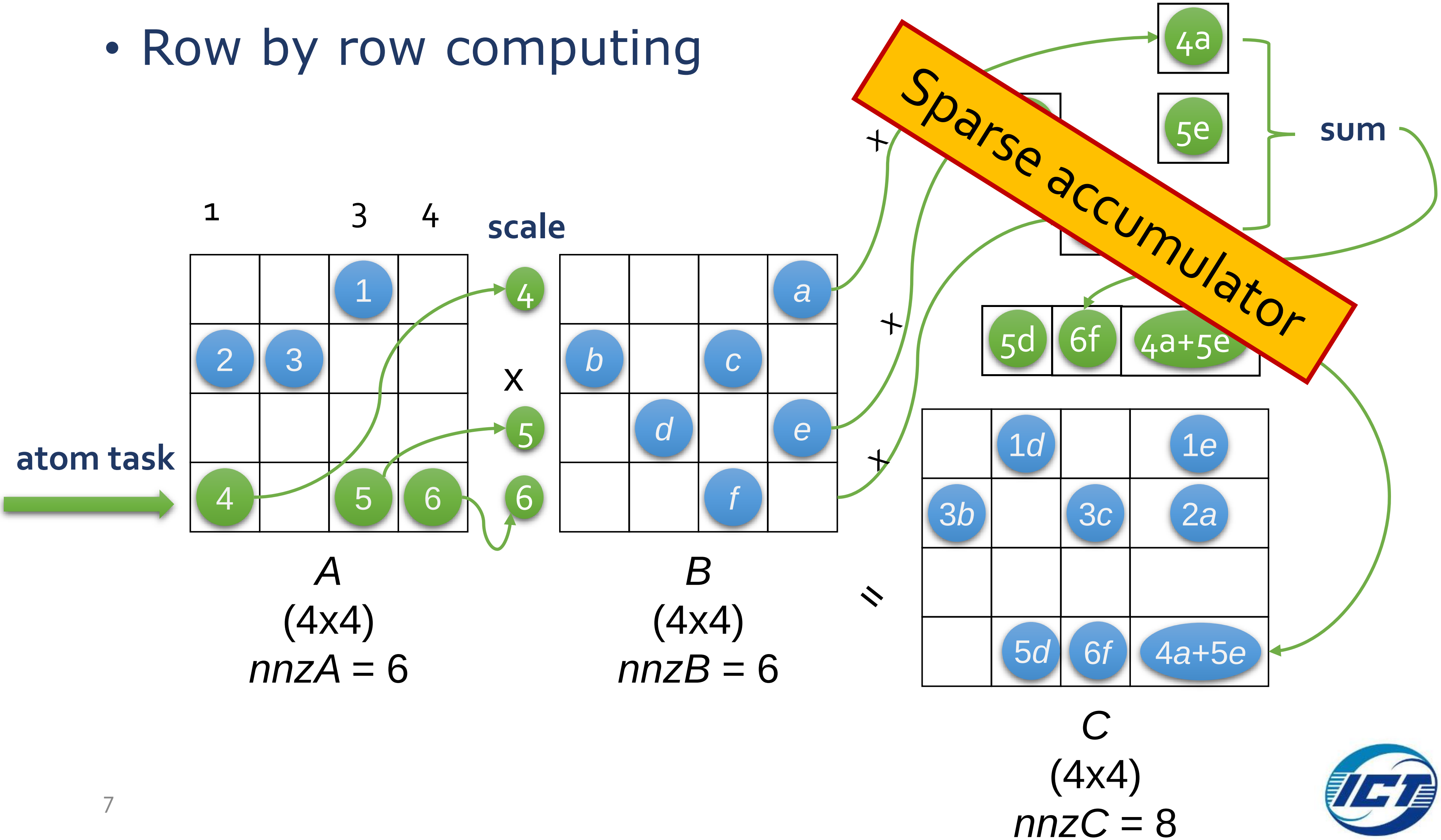
SpGEMM algorithm – basic

- Row by row computing



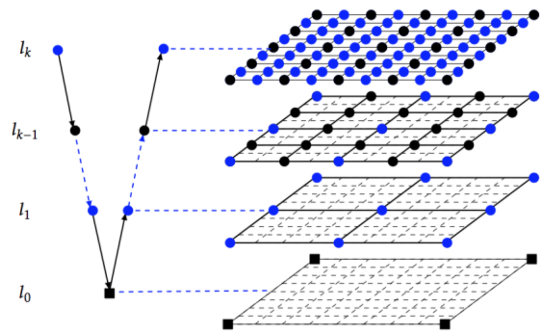
SpGEMM algorithm – basic

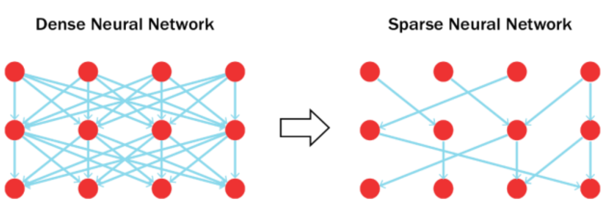
- Row by row computing

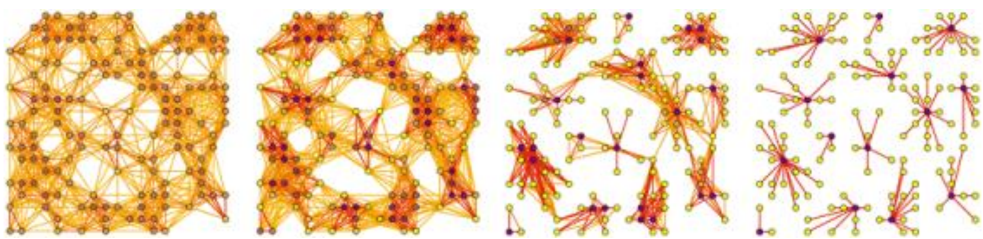


SpGEMM - Applications

- Algebraic multigrid method
- Breadth first search
- Shortest path
- Colored intersection
- Sub-graghs
- Sparse neural network
- …

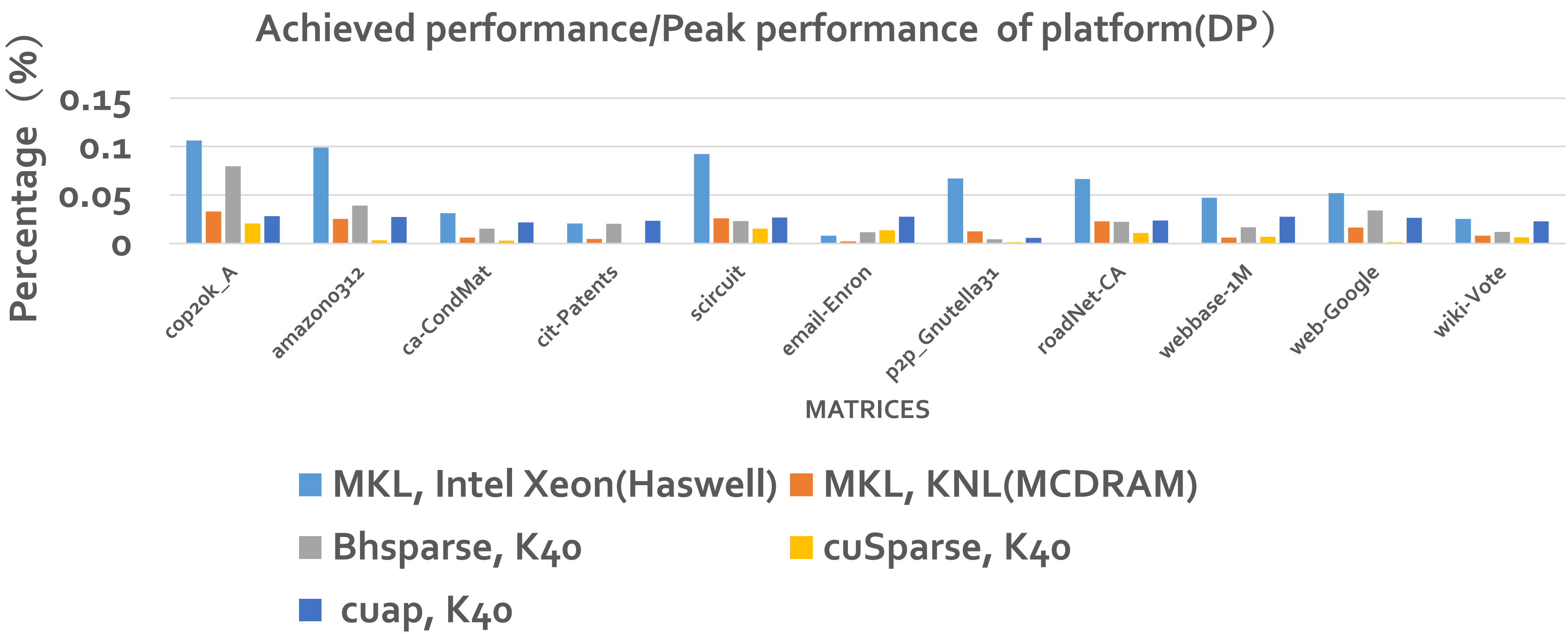






Sparse matrix floating efficiency

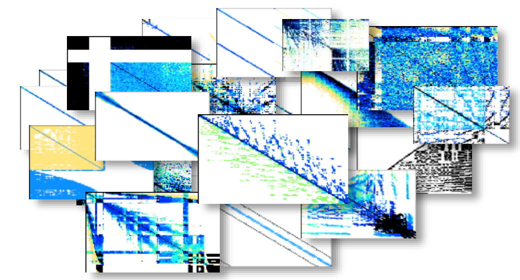
– Iregular sparse matrices



Overview

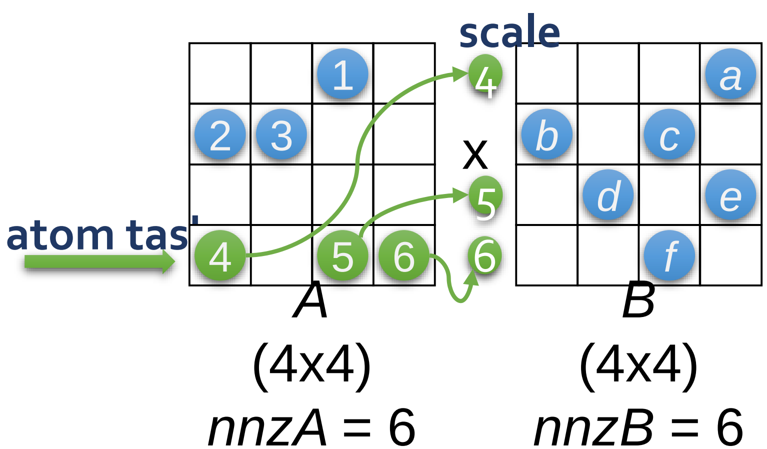
- Sparse (SpGEMM) and its applications
- Challenges
- Our sparse accumulator optimization
- Experimental results
- Conclusions

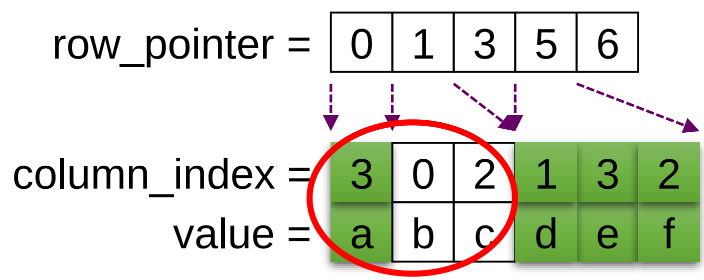
Challenges



- **The number of nonzeros of the output is unknown in advance**
- **Irregular memory access**
- **Poor data locality**
- **Load imbalance problem**

Increasing data access latency



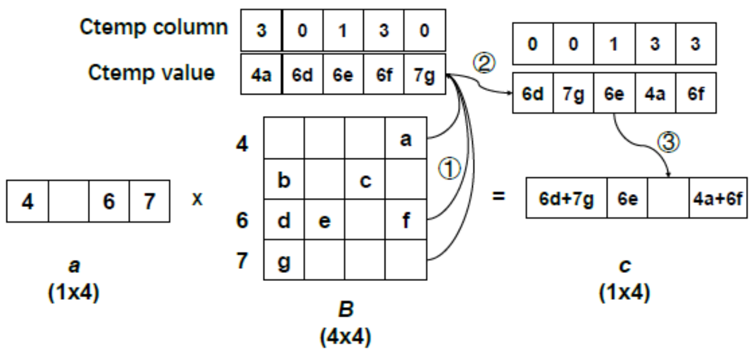


B in CSR-format

Overview

- Sparse (SpGEMM) and its applications
- Challenges
- Our sparse accumulator optimization
- Experimental results
- Conclusions

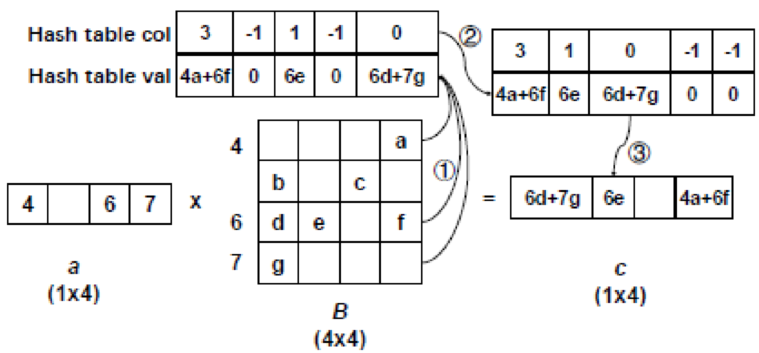
Basic Sparse Accumulators



$c=aB$

① expanding ② sorting ③compressing

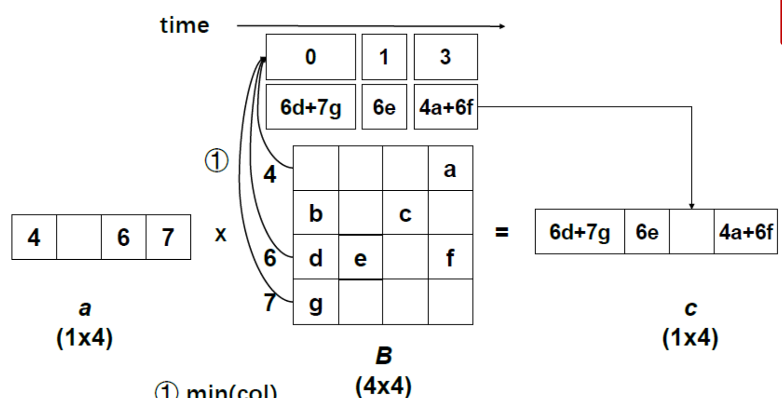
sort



$c=aB$

① hash: key = col ② shrinking ③sorting

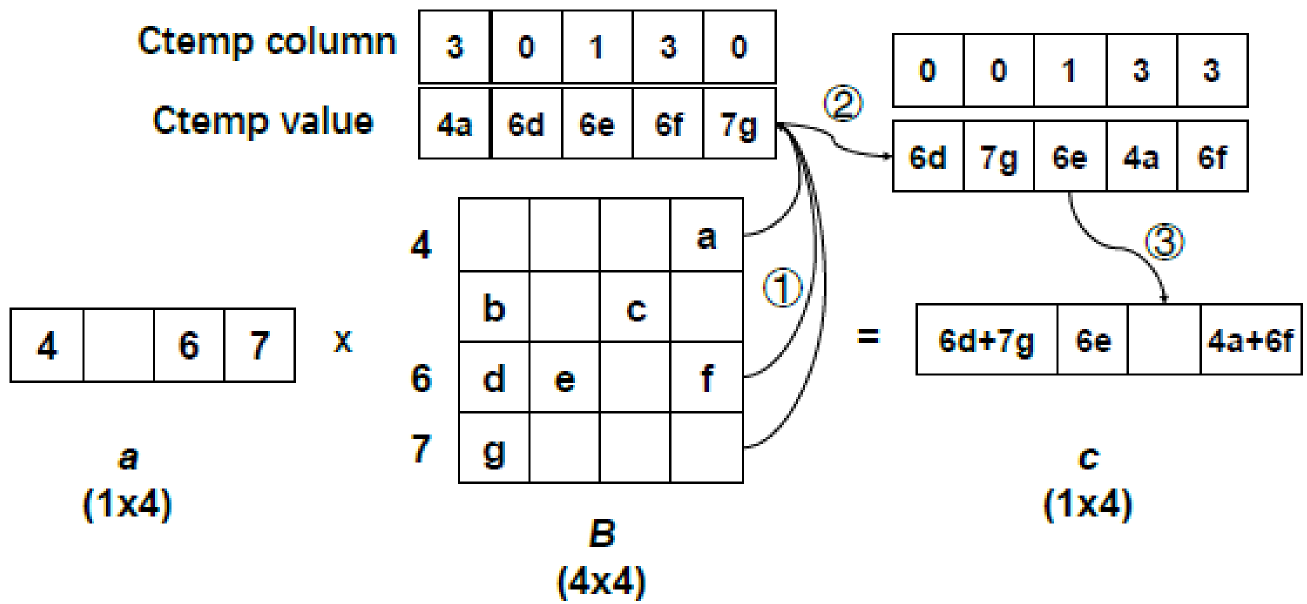
hash



$c=aB$

merge

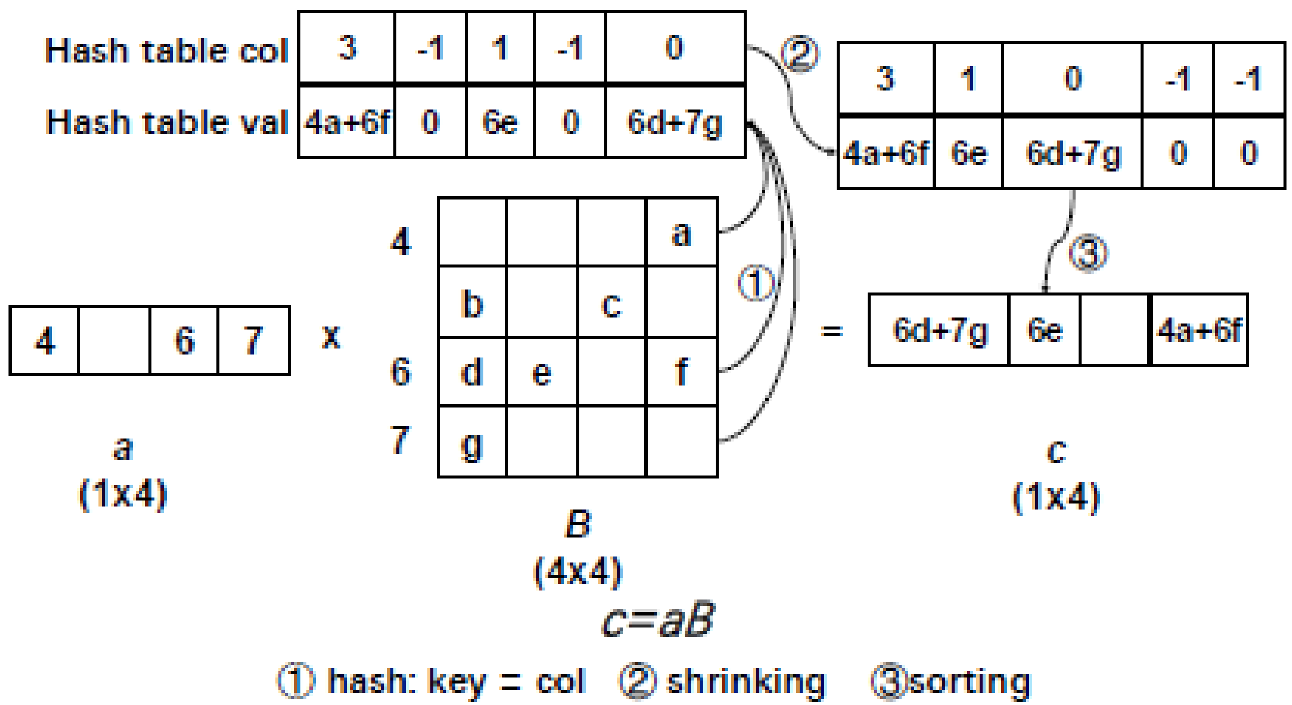
Sort-based Sparse Accumulator



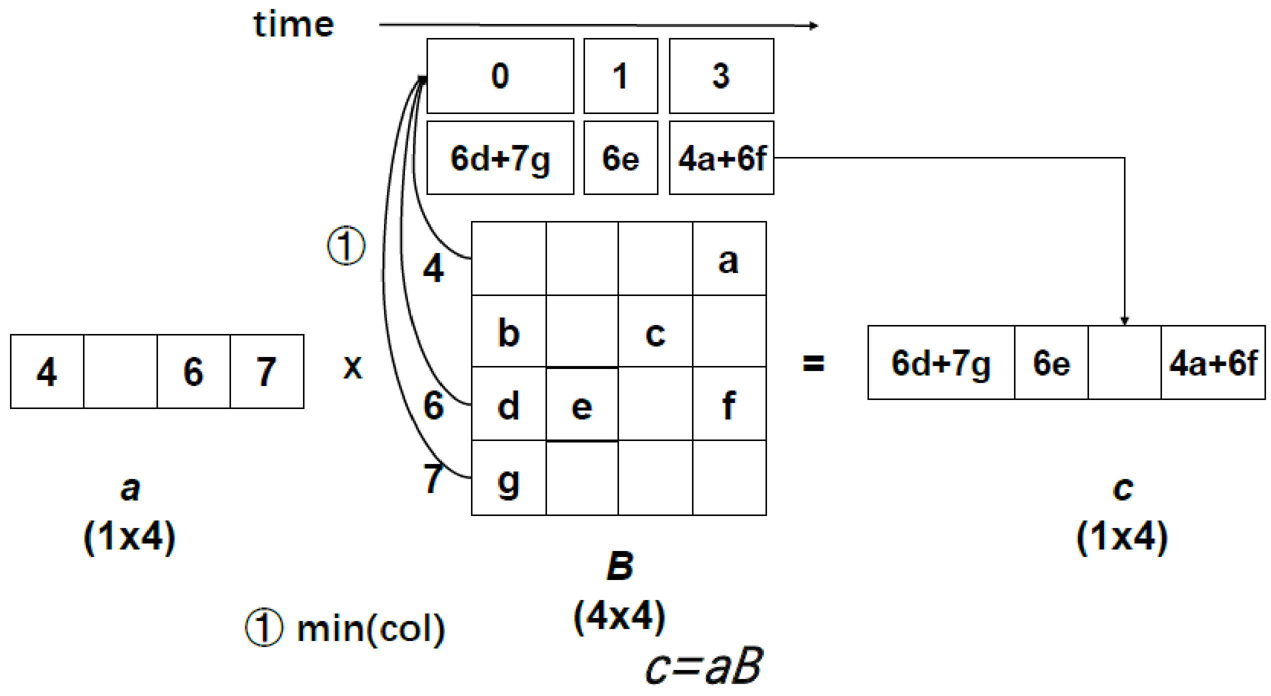
c=aB

① expanding ② sorting ③compressing

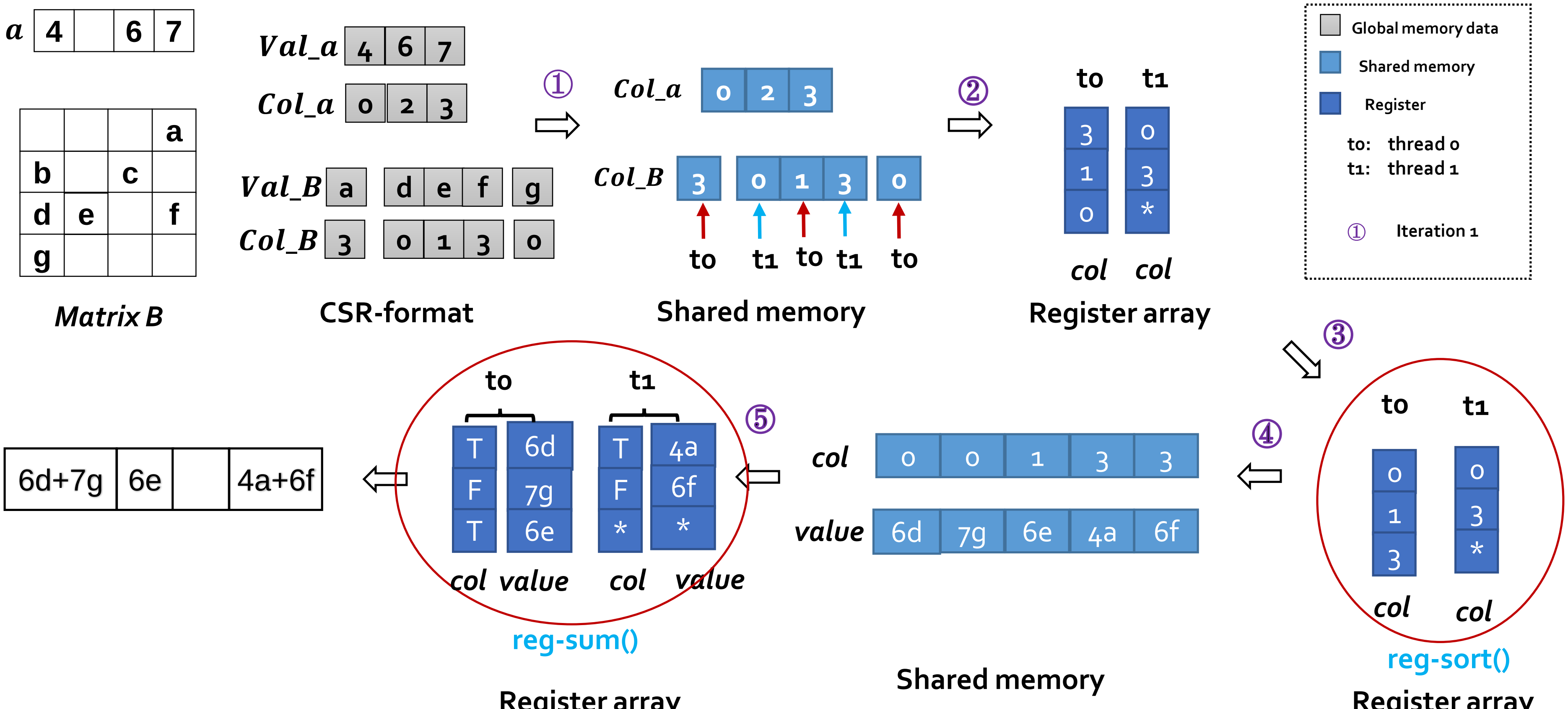
Hash-based Sparse Accumulator



Merge-based Sparse Accumulator

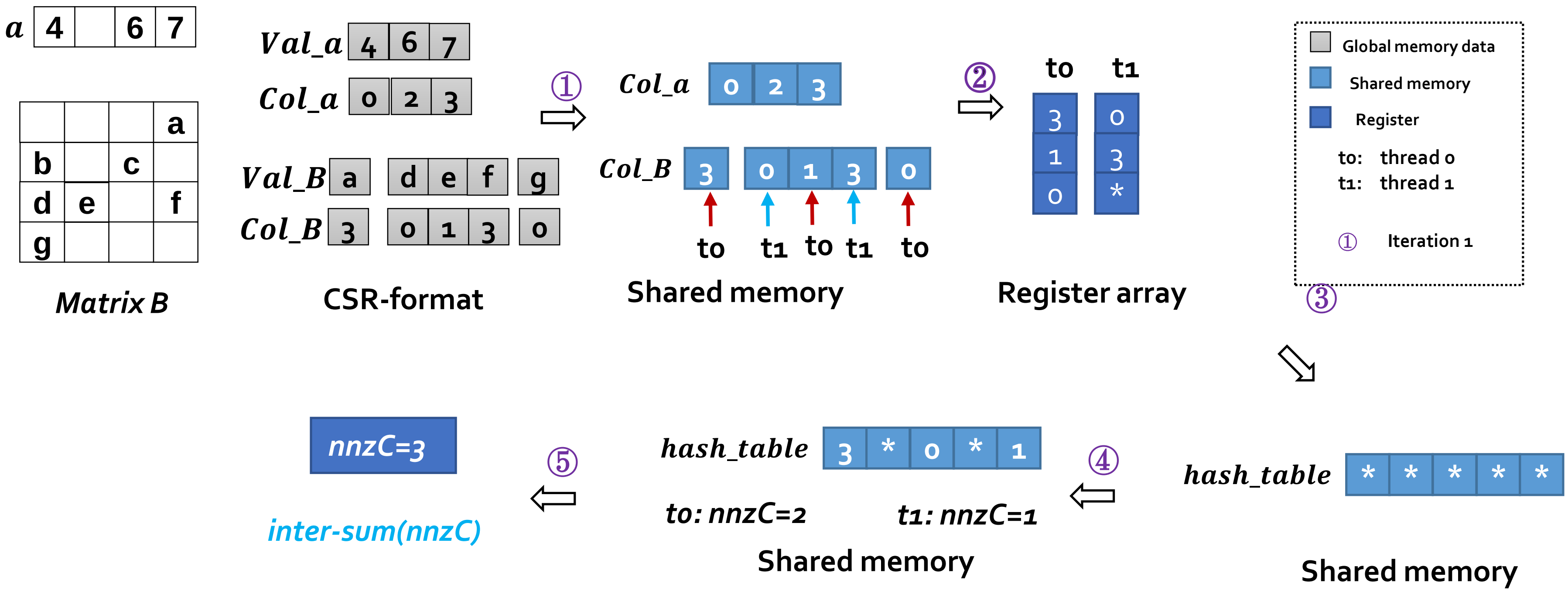


Reg-sort Sparse Accumulator



There is no need to use shared memory for heavy computation and data movement.

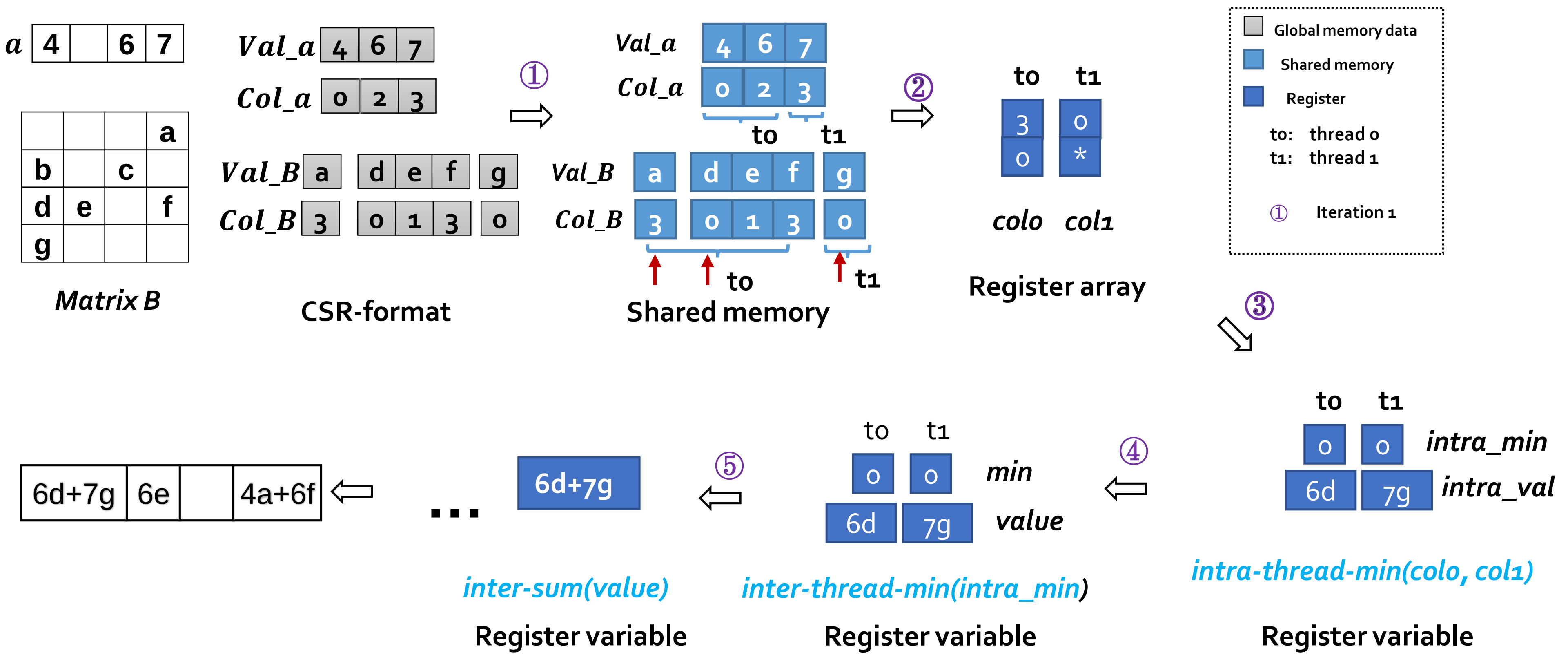
Reg-hash Sparse Accumulator



Decrease the total number of shared memory hash operations. (3 vs 4 iterations)

The intermediate products are well organized in a load balanced way.

Reg-merge Sparse Accumulator



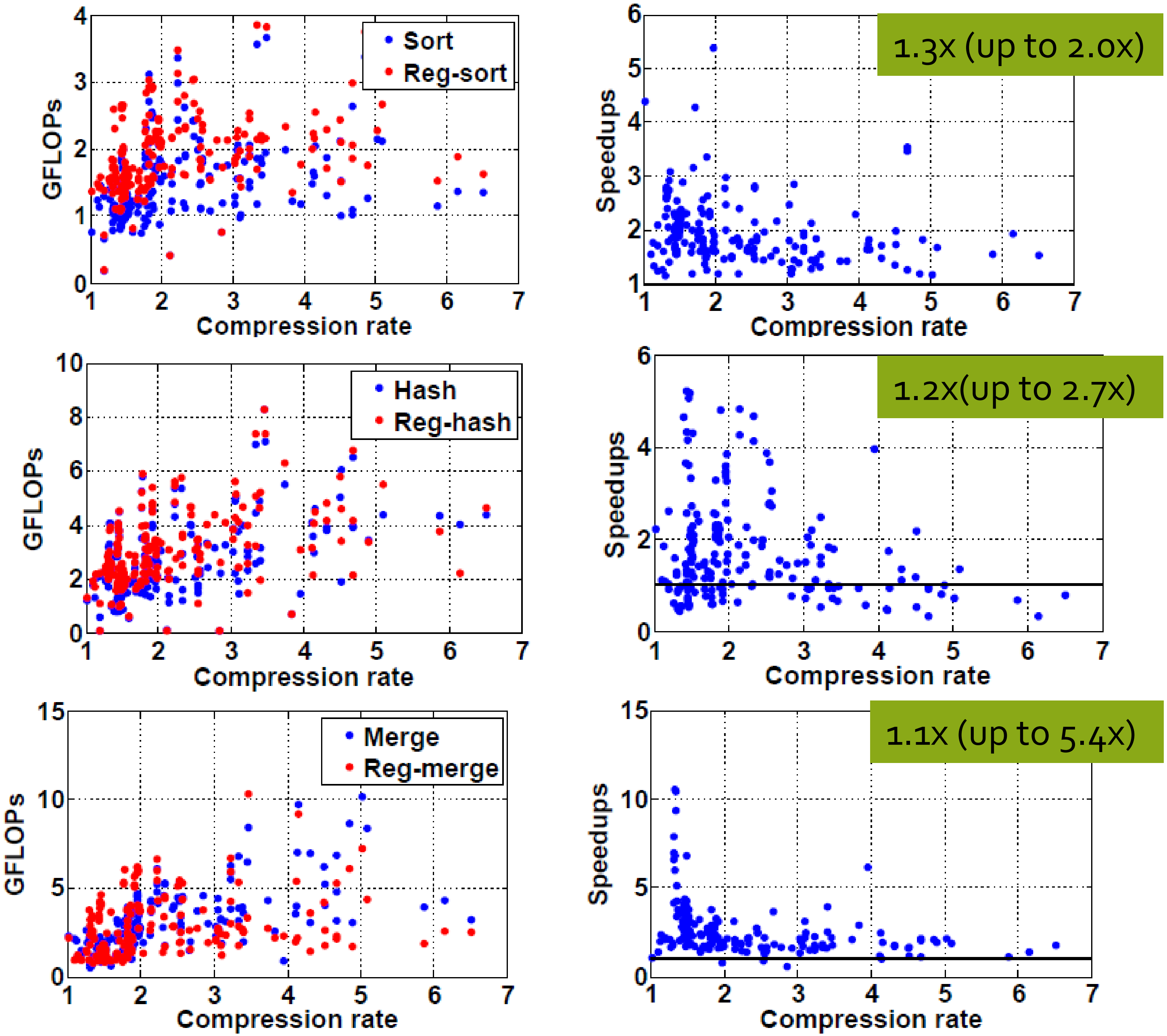
The elements are all computed inside registers instead of the global memory.

Overview

- Sparse (SpGEMM) and its applications
- Challenges
- Our sparse accumulator optimization
- Experimental results
- Conclusions

Experiments

- Platforms
  - Nvidia Pascal P100 GPU (3584 CUDA cores and 16GB HBM2 memory)
  - CUDA v8.0 and Intel C/C++ compiler v18.
- Benchmark：205 matrices from SuiteSparse Matrix Collection



(a) Overall performance. (b) Kernel speedups.

Overview

- Sparse (SpGEMM) and its applications
- Challenges
- Our sparse accumulator optimization
- Experimental results
- Conclusions

Conclusions

- This work has proposed three register-aware optimization methods to improve the performance of SpGEMM.
- The three new sparse accumulators have covered the parallel primitives, such as, sort, hash and merge.
- Numerical results demonstrates the significant performance improvement using the new methods.

Thanks !

Any Ques?