ExTensor: An Accelerator for Sparse Tensor Algebra

Kartik Hegde, Hadi Asghari-Moghaddam, Michael Pellauer, Neal Crago, Aamer Jaleel, Edgar Solomonik, Joel Emer, and Christopher W. Fletcher. 2019. ExTensor: An Accelerator for Sparse Tensor Algebra.

Master: Intelligent Systems

Subject: SJK012 - High Performance Machine Learning

Author: Pablo Muñoz Alcaide & Vicent Santamarta Martínez

GENERALITAT VALENCIANA

valgrAI Valencian Graduate School and Research Network of Artificial Intelligence

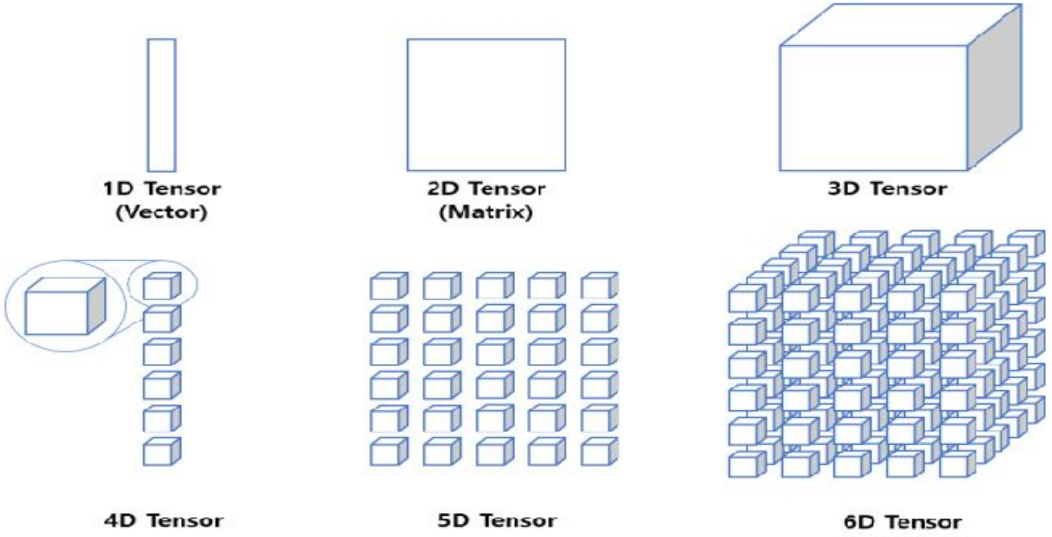
Table of Contents

01

Introduction & Motivation

What is a Tensor?

A tensor in machine learning is a multi-dimensional array of arbitrary order (dimensionality) used to represent data and model parameters. Basically, Is a generalisation of vectors and matrices to N-dimensions.



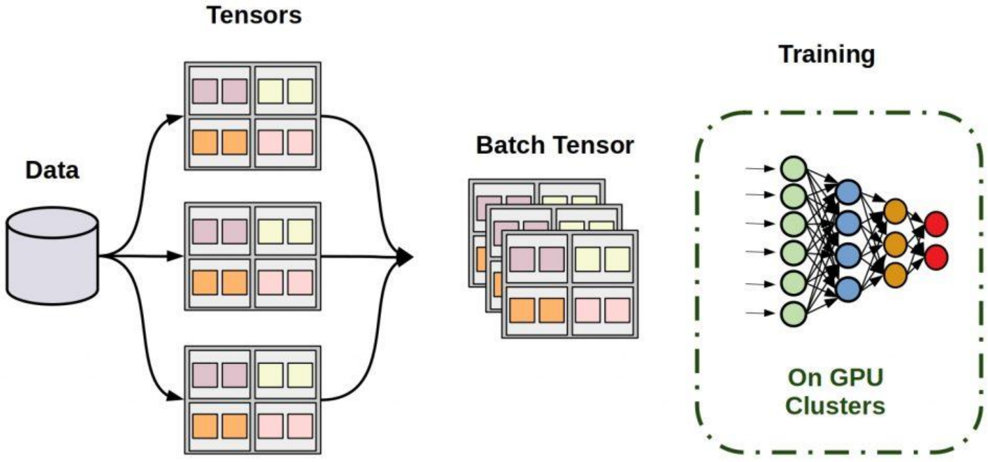
Tensor Algebra

Tensor algebra is the process of performing binary operations between tensors to produce new tensors.

Name	Tensor index notation
GEMV	$Z_i = \alpha \sum_k A_k B_{ki} + \beta C_i$
GEMM	$Z_{ij} = \alpha \sum_k A_{ik} B_{kj} + \beta C_{ij}$
TTV	$Z_{ij} = \sum_k A_{ijk} B_k$
TTM	$Z_{ijk} = \sum_l A_{ijl} B_{kl}$
SDDMM	$Z_{ij} = C_{ij} \sum_k A_{ik} B_{kj}$
MTTKRP	$Z_{ij} = \sum_{kl} A_{ikl} B_{kj} C_{lj}$
2D Conv	$O_{xy} = \sum_{rs} I_{(x+r)(y+s)} F_{rs}$
CNN layer	$O_{zuxy} = \sum_{crs} I_{zc(\gamma x+r)(\gamma y+s)} F_{ucrs}$

Why Tensors are important in ML?

Tensors enables complex data representation and efficient computation in diverse Machine Learning applications.



Challenge: Tensor Sparsity

The variety of tensor kernels, their extreme sparsity (percentage of data which is non-zero), and their compressed representations make tensor algebra challenging on today's platforms.

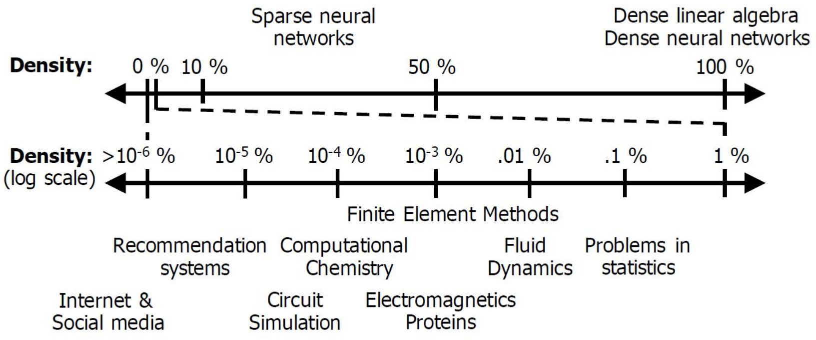


Figure 1: Tensor sparsity by workload domain.

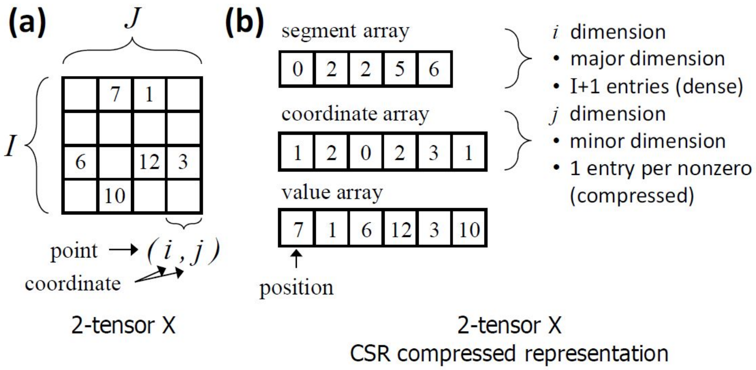


Figure 2: Tensor terminology & example compression using CSR.

Main Motivation

The main opportunity provided by sparsity in tensor operations is the potential to exploit axiom $0 \cdot x = 0$. But, How…?

- Some platforms exploit this axiom in scalars, avoiding delivering *x* to the staging buffers.
- In higher-order tensor algebra this opportunity applies even when x is not a scalar.
- *x,* might be a tile or an un-evaluated tensor.

Recognizing that the other operand is 0 means we don’t have to transfer data (or metadata) for the entire tile.

02

ExTensor Solution

ExTensor

Extensor is an accelerator architecture built around the idea of locating non-zero data to eliminate ineffectual computation. The main contributions of ExTensor are:

1) First accelerator for general, sparse tensor algebra.
2) General abstraction -based on intersections coordinates of non-zero data- for describe the opportunities to skip the work due to sparsity, in different granularities.
3) Hardware mechanism and optimization, for perform this intersections at multiple levels of an accelerator memory hierarchy.
4) Improves speedups in a lot of kernels.

03

Why is ExTensor Possible?

Intersection Opportunities (i)

Intersections opportunities are where the coordinates of non-zero elements from two tensors overlap, i.e., intersect.

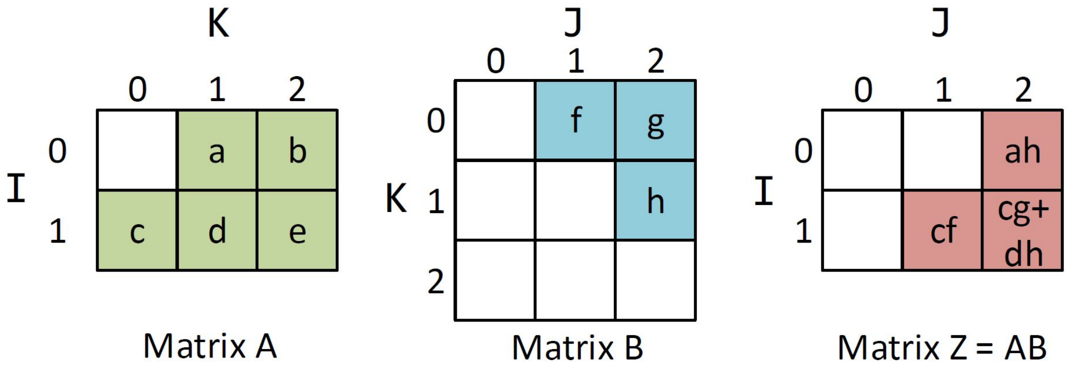
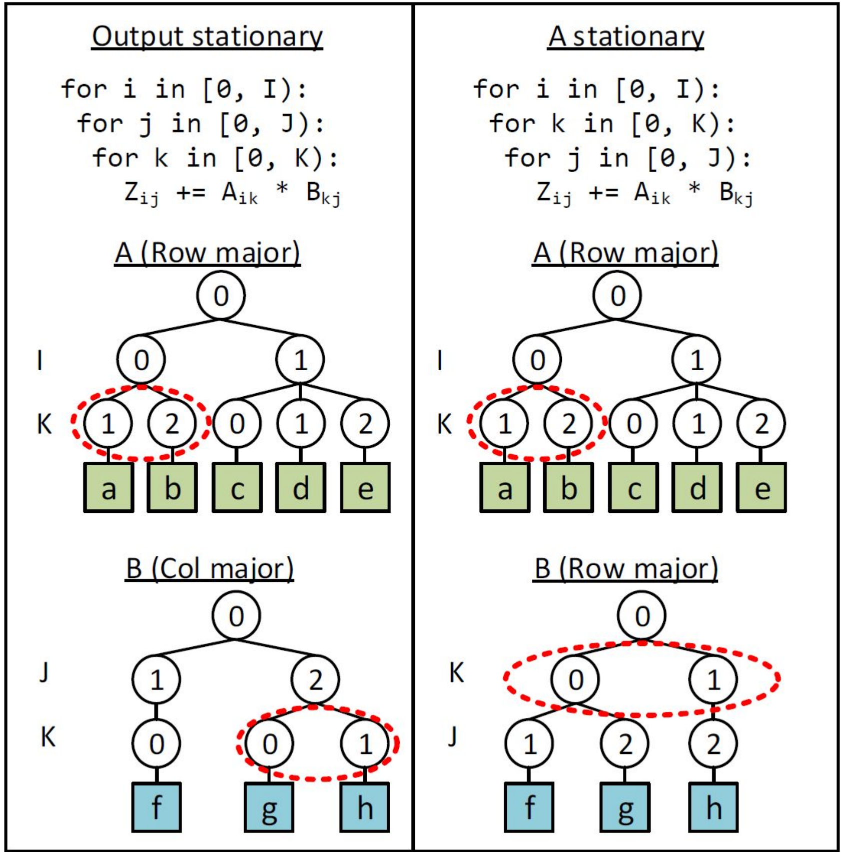


Figure 3: Example matrices. White space indicates zero value. Numbers along each dimension are coordinates for that dimension.

Intersection Opportunities (ii)

- N - levels trees are a graphical representations of the compression formats.
- Allows to identify the intersections coordinates at different levels.
- This intersections are the multiplication of non-zero data.



04

ExTensor Architecture

Scanner Hardware (i)

- The scanner hardware is the unit that stores individual coordinate streams.
- Metadata storage interfaces with a FSM hardware, that iterates through the storage.
- Scanner outputs coordinates stored in the metadata, in increasing order by coordinate.

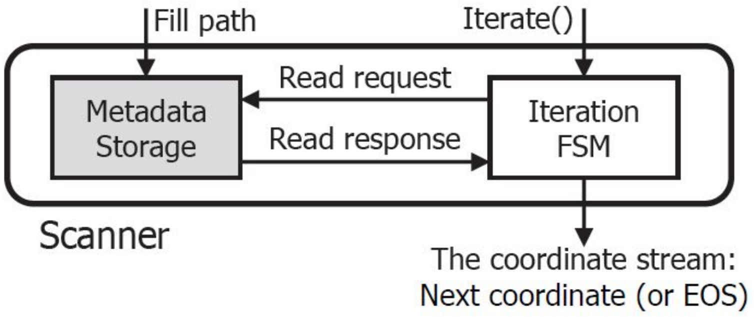


Figure 5: Scanner hardware. Storage is shaded.

Scanner Hardware (ii)

- Scanners iterates in parallel to find intersections.
- If two coordinates match an intersection is found and the matched coordinate is processed.
- If the coordinate from Scanner A is less than from Scanner B. Scanner A moves to its next coordinate, discarding the current one.
- Same with Scanner B.
- Intersect block process the matches.

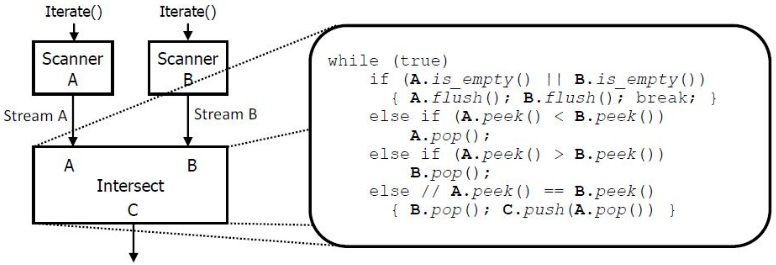
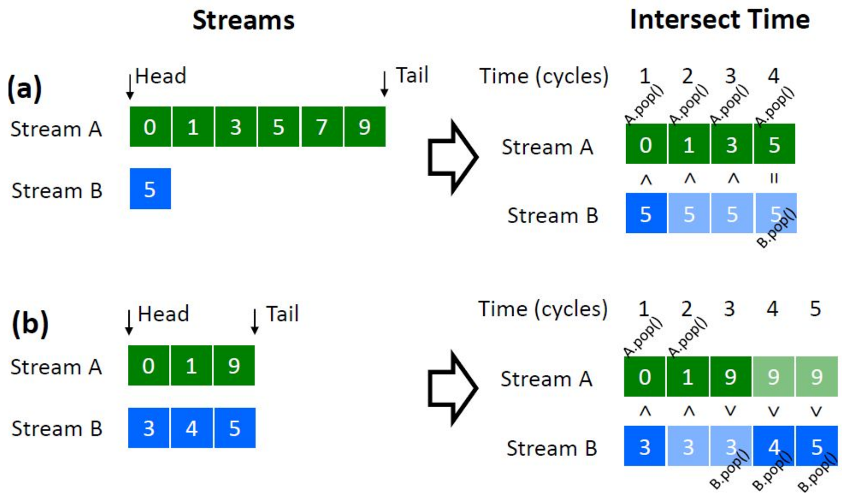


Figure 6: Basic intersection hardware and algorithm.

Problems of Scanner

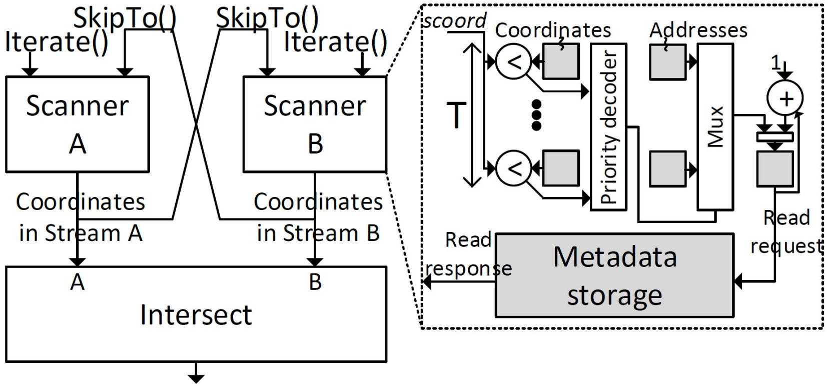
The scanner has efficiency problem because completes an intersection in *$O(Stream_A \cup Stream_B)$ cycles.* That is, having to step through many elements that do not result in productive matches.



Optimization of Scanner

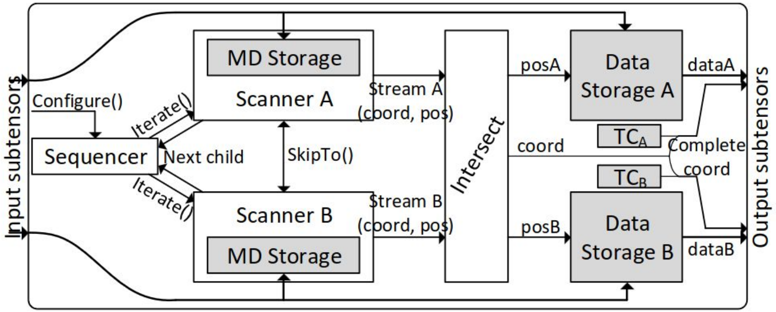
Two ways of optimization:

- Skip Mechanism design.
- Content Addressable lookup.



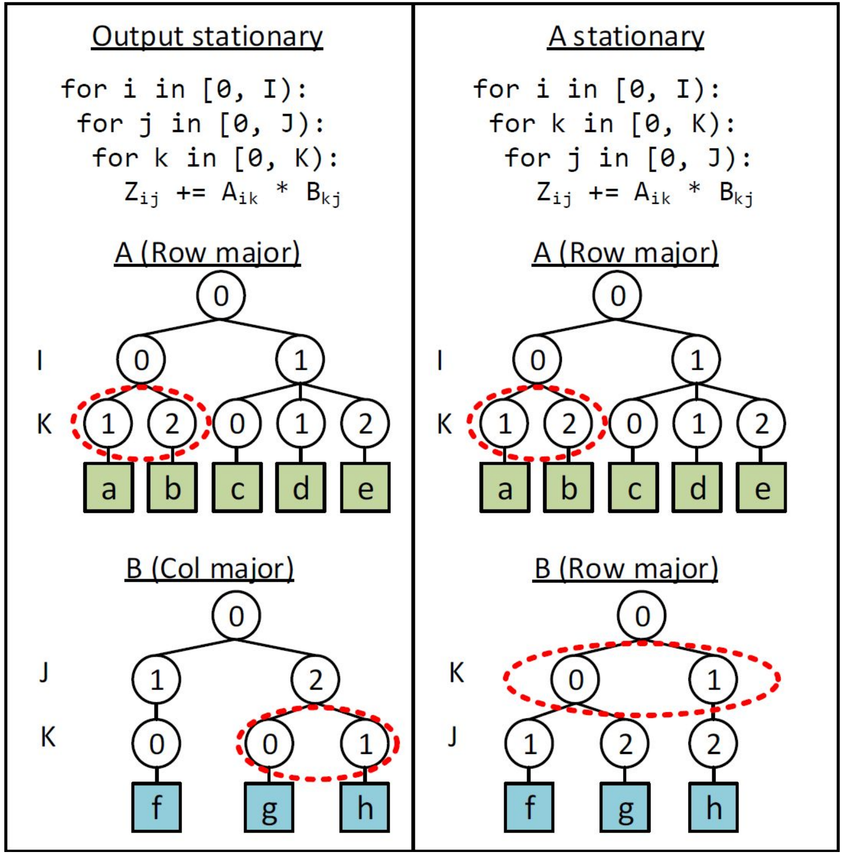
Staging Intersections

- A sequencer determines the order in which streams are transferred.
- Two scanners fetch and prepare the data for intersection.
- Dynamic scheduling reduces idle times.



Using Intersected streams

- Intersections are used as lookup.
- Intersected data is moved to faster-access memory.



Macro Architecture

- **DRAM**: Primary memory storage
- **LLB**: Closer to processing elements
- **PEs**: Perform computations
- **NoC**: Facilitates data transfer

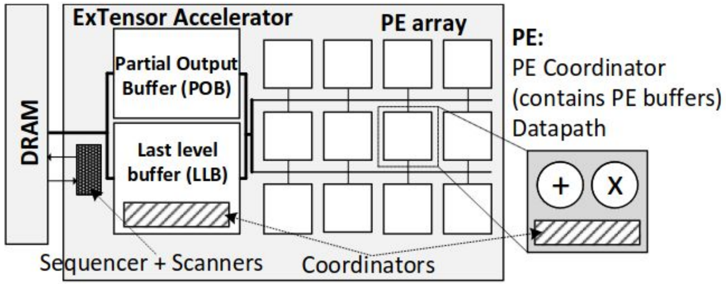
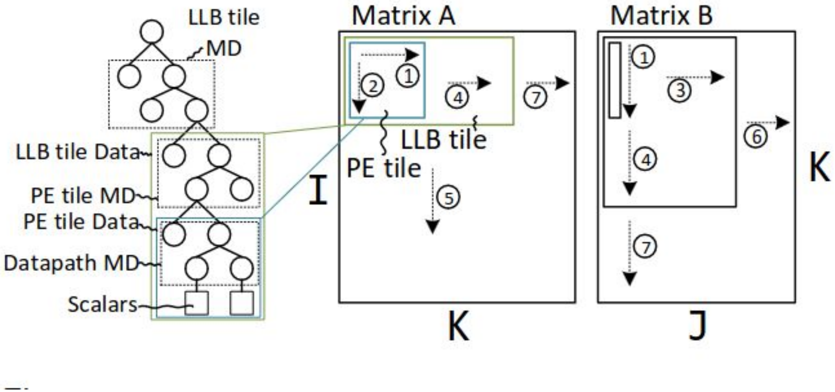


Figure 12: The ExTensor accelerator.

Dataflow and Tiling



- **Tiling:** Breaks data into smaller pieces.
- Organize data in **multiple levels** of tiles.
- Intersections are handled at multiple levels.
- The data needed for computations is pre-staged.

Partial Output management

- Two key observations:
    1. Computations at PE level have a smaller chance to generate a partial output.
    2. The partial output reductions can be ordered.
- Partial Output Buffer (POB): Stores immediate results.
- Dynamic memory management.

05

Experiments & Results

Methodology

- Detailed simulation model to evaluate the performance.
- Use of FROSTT tensor dataset and SuiteSparse matrix collection.
- Real - Life tensors and tensors operations (SpMSpM, SpMM,...)
- Performance compared to optimized CPU codes

  - Intel MKL Library

  - TACO tensor compiler

Main Results

- **Generalized Matrix Multiplication (GEMM):** ExTensor is 3.4x (SpMSpM) and 1.3x (SpMM) faster than the CPU.
- **Generalized Tensor Algebra:** Extensor has 2.8x (TTV), 24.9x (TTM) and 2.7x (SDDMM) average speedups.
- **Synthetic Data:** showed scalable performance when tested with synthetic data that varied in size and sparsity, affirming its robustness across different data conditions.
- **Hardware Implementation:** The practical implementation of the accelerator in hardware demonstrated its feasibility.

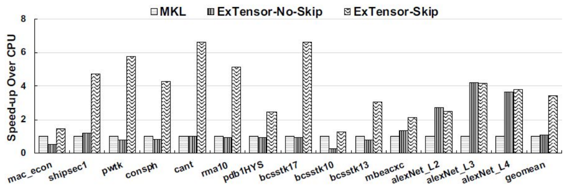


Figure 14: ExTensor speed-up relative to MKL (SpMSpM).

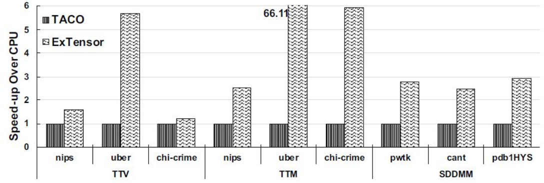


Figure 17: Performance comparison between ExTensor variants and TACO for generalized tensor algebra.

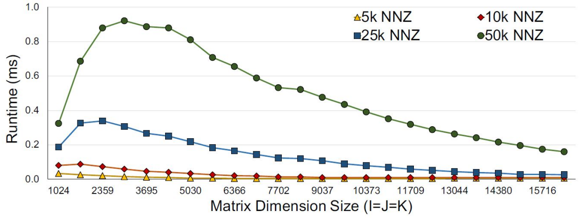


Figure 19: ExTensor's SpMSpM performance across varying dimension sizes with constant number of non-zeros (NNZ) per matrix.

06

Conclusions

Conclusions

- Extensor: new approach for performing general tensor algebra using hierarchical and compositional intersection
- First accelerator for general, sparse tensor algebra.
- ExTensor demonstrated significant performance over traditional CPU-Systems.

Future Work Opportunities:

- Efficient real-time conversion between compressed data formats.
- Implementing online tiling strategies rather than offline.
- Addressing issues that limit bandwidth scaling

Thanks!

Do you have any questions?

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**