[Processor Research Team]

# Architectural Challenges in Accelerating Sparse Matrix Multiplication with CGRAs for AI


Lin Teng[1,2], Chenlin Shi[1,2], Boma Adhi[1], Yanchen Li[1], Kentaro Sano[1]

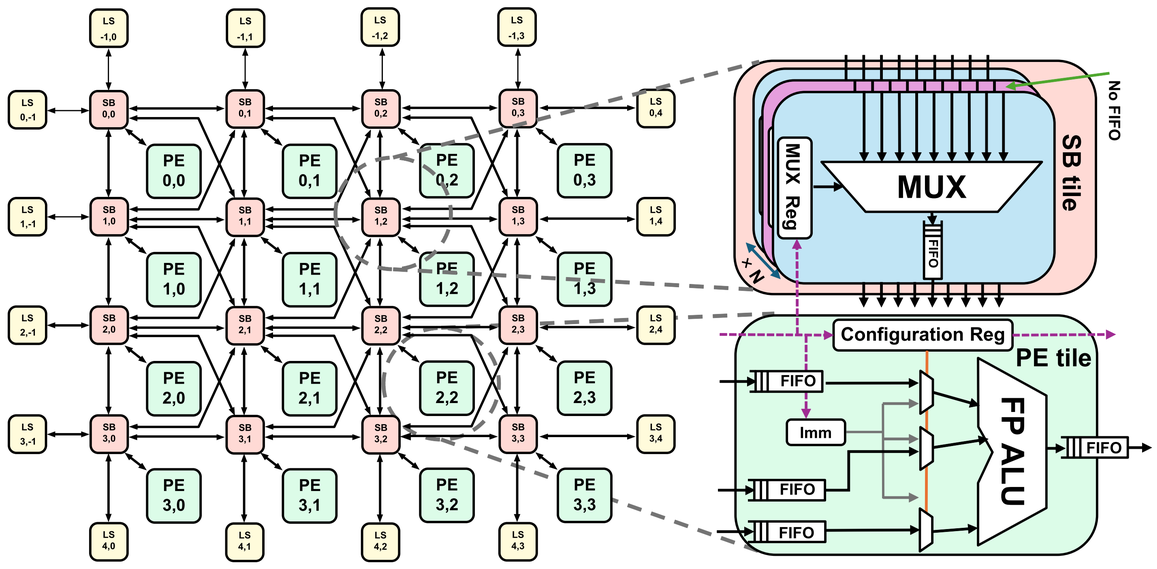
teng@hpc.is.uec.ac.jp , (chenlin.shi, boma.adhi, yanchen.li, kentaro.sano)@riken.jp

[1] RIKEN Center for Computational Science, Hyogo, Japan

[2] The University of Electro-Communications, Tokyo, Japan


## Introduction

- **Coarse-grained reconfigurable Arrays (CGRAs)** have emerged as promising accelerators with the increasing demand for AI applications
  - Can efficiently implement nonlinear functions critical to AI workloads, such as **softmax**
  - Have also shown the feasibility to be configured to **matrix** accelerators.
- **Use sparse weight matrices** instead of dense ones, it has become feasible to drastically reduce memory footprint while maintaining task-specific accuracy.
- **Supporting sparse matrix multiplication at the hardware level** has become a challenge.




## CGRAs Architecture

- **Three types of tiles:**
  - **Switch block (SB) tiles** are responsible for data forwarding.
  - **Load/store (LS) tiles** handle memory interactions.
  - **Processing element (PE) tiles** perform arithmetic computations.
- **Use bitstream** to update the configuration registers of SB and PE tiles.
- **Can emulate a systolic array-liked structure** in CGRA-based matrix multiplication.
  - **Weight- or output-stationary systolic dataflows** are widely adopted.
  - Mimic the data movement patterns of systolic arrays by constraining the SB tiles to form orthogonal connections in all four cardinal directions.

## Data format challenge

- **The conventional compressed sparse row (CSR)** is **not suitable** for SpMM in AI applications.
- **Much higher non-zero elements** in purned AI models than traditional HPC applications.
  - Each non-zero element must be stored with its column index, which can case the total data footprint to exceed that of the original dense matrix.
- **CSR cannot reduce the memory footprint**
  - This is a primary motivation fort sparsity in AI.

Traditional HPC sparse matrix

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 2 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 3 | 0 |
| 0 | 0 | 0 | 0 |

data=[2,3]
indices=[3,2]
indptr=[0,1,1,2,2]

Pruned AI sparse matrix

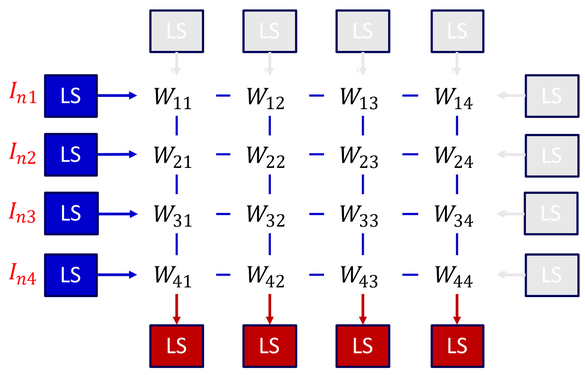
| | | | |
|---|---|---|---|
| 2 | 0 | 0 | 3 |
| 0 | 4 | 0 | 0 |
| 0 | 0 | 5 | 3 |
| 1 | 0 | 0 | 2 |

data=[2,3,4,5,3,1,2]
indices=[0,3,1,2,3,0,3]
indptr=[0,2,3,5,7]

## Systolic Dataflow Challenge

- **Systolic dataflow** offers efficient **dense** matrix computation.
  - **Fails to exploit** the performance benefits of sparsity in modern AI workloads
  - **Precise spatiotemporal alignment** of activations and weights at corresponding PEs are relied to produce correct results.
- **Irregular** positions of **non-zero elements** in pruned AI models.
  - **Paths across SB tiles** are no longer **fixed.**
  - **The internal data forwarding paths across SB tiles** can not be determined **statically.**
- **Frequent bitstream reconfiguration** to accommodate the dynamic sparsity patters required for CGRAs.
  - **Overhead and complexity** of executing SpMM **significantly increased.**

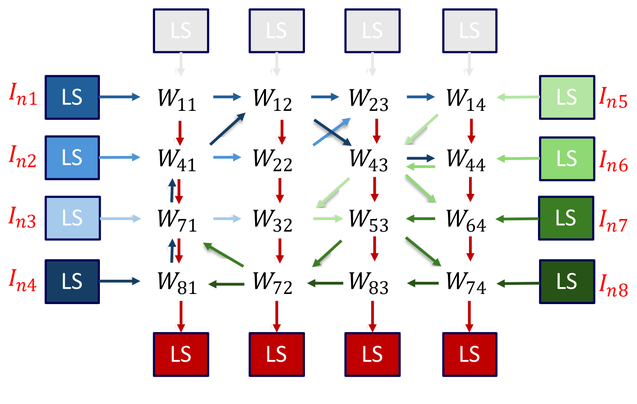
| | | | |
|---|---|---|---|
| $W_{11}$ | $W_{12}$ | $W_{13}$ | $W_{14}$ |
| $W_{21}$ | $W_{22}$ | $W_{23}$ | $W_{24}$ |
| $W_{31}$ | $W_{32}$ | $W_{33}$ | $W_{34}$ |
| $W_{41}$ | $W_{42}$ | $W_{43}$ | $W_{44}$ |




## Early-stage SpMM approach

- **LS on the right-side** can be used for more input matrix
  - Assume that CGRAs size is **n × n**, we can calculate when input matrix is **a × 2n,** weight matrix is **2n x n.**
  - Make at least **half of the elements of weight matrix** for each column **zero** to ensure every result output from the bottom LS tiles can be specified.
  - **NB-complete problem** or even **impossible** if weight matrix have no more limitation to find the path.
- Make all rows of **weight matrix** can be **paired complementarily**
  - It will **significantly reduce the difficulty** to find the path, only need to decide where to input the data and data from which side need to be calculated
  - **Reconfiguration** for each **weight matrix** is **inevitable.** How ever, only few path need to be changed, so it can potentially be **simplified.**

| | | | |
|---|---|---|---|
| $W_{11}$ | $W_{12}$ | 0 | $W_{14}$ |
| 0 | $W_{22}$ | $W_{23}$ | 0 |
| 0 | $W_{32}$ | 0 | 0 |
| $W_{41}$ | 0 | $W_{43}$ | $W_{44}$ |
| 0 | 0 | $W_{53}$ | 0 |
| 0 | 0 | 0 | $W_{64}$ |
| $W_{71}$ | $W_{72}$ | 0 | $W_{74}$ |
| $W_{81}$ | 0 | $W_{83}$ | 0 |




- **For example, weight martix W, can be paird like**

| | | | |
|---|---|---|---|
| $W_{11}$ | $W_{12}$ | 0 | $W_{14}$ |
| 0 | 0 | $W_{53}$ | 0 |

(1,5)

| | | | |
|---|---|---|---|
| 0 | $W_{22}$ | $W_{23}$ | 0 |
| $W_{61}$ | 0 | 0 | $W_{64}$ |

(2,6)

| | | | |
|---|---|---|---|
| 0 | $W_{32}$ | 0 | 0 |
| $W_{41}$ | 0 | $W_{43}$ | $W_{44}$ |

(3,4)

| | | | |
|---|---|---|---|
| 0 | $W_{72}$ | 0 | $W_{74}$ |
| $W_{81}$ | 0 | $W_{83}$ | 0 |

(7,8)

- **Then combine each pair into one row**

| | | | |
|---|---|---|---|
| $W_{11}$ | $W_{12}$ | $W_{53}$ | $W_{14}$ |

(1,5)

| | | | |
|---|---|---|---|
| $W_{61}$ | $W_{22}$ | $W_{23}$ | $W_{64}$ |

(2,6)

| | | | |
|---|---|---|---|
| $W_{41}$ | $W_{32}$ | $W_{43}$ | $W_{44}$ |

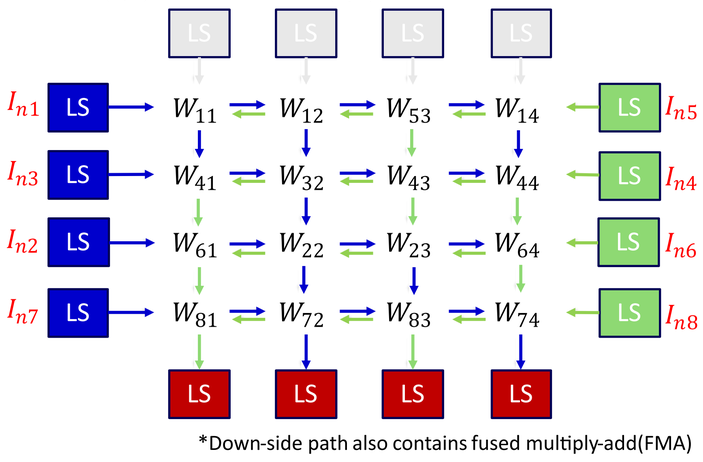
(3,4)

| | | | |
|---|---|---|---|
| $W_{81}$ | $W_{72}$ | $W_{83}$ | $W_{74}$ |

(7,8)

| | | | |
|---|---|---|---|
| $W_{11}$ | $W_{12}$ | 0 | $W_{14}$ |
| 0 | $W_{22}$ | $W_{23}$ | 0 |
| 0 | $W_{32}$ | 0 | 0 |
| $W_{41}$ | 0 | $W_{43}$ | $W_{44}$ |
| 0 | 0 | $W_{53}$ | 0 |
| $W_{61}$ | 0 | 0 | $W_{64}$ |
| 0 | $W_{72}$ | 0 | $W_{74}$ |
| $W_{81}$ | 0 | $W_{83}$ | 0 |

$W$




*Down-side path also contains fused multiply-add(FMA)

## Conclusion

- In this poster, we present a detailed analysis of the challenges of executing SpMM on CGRAs, focusing on mainstream encoding and systolic computing techniques commonly used in AI accelerators.
- Our future work aims to develop novel software-hardware co-design methodologies to enable efficient SpMM on CGRAs.