[Processor Research Team]

A Needle in a Haystack: Why SoTA CGRA Compilers Struggle with the Systolic Array Computing Kernels

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

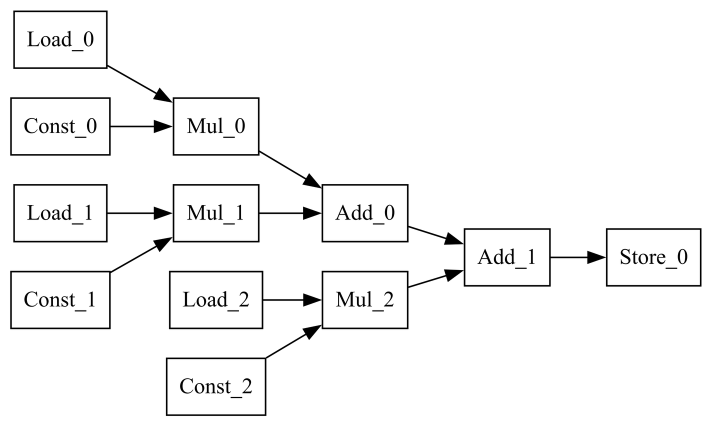
teng@hpc.is.uec.ac.jp , (chenlin.shi, boma.adhi, kentaro.sano)@riken.jp

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

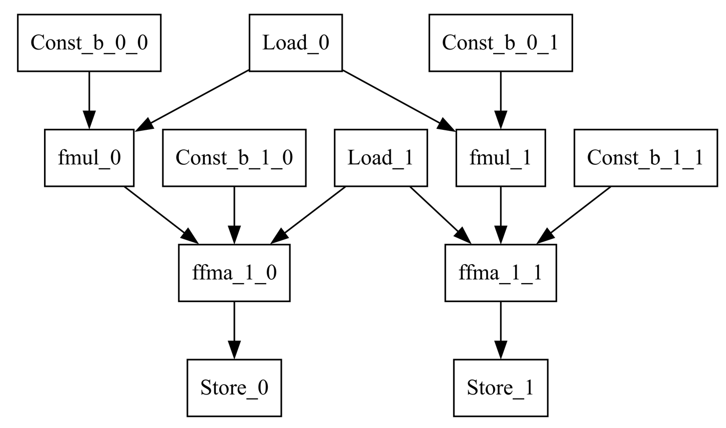
[2] The University of Electro-Communications, Tokyo, Japan

Introduction

- **Modern Coarse-grained reconfigurable Arrays (CGRAs)** provide **significant programmability advantages** over **fixed-function** systolic arrays (SAs).
  - **More flexible interconnects** and **sophisticated processing elements(PEs)** supported.
  - **PEs** often support **multiple opcodes** and **multi-context capability.**
- **"Place and Route" (P&R)** for **irregularly-shaped dataflow graphs (DFGs)** is researched to solve complex mapping problem.
  - **State-of-the-art mappers** like GenMap, CluMap, E2EMap **effectively explore the complex solution space** for these **irregular kernels.**
- The kernel of **systolic array-style (SA-style)** workloads like **GeMM** are **highly regular** but **possess dense, structured connectivity.**
  - **Enormous solution space** cause conventional design of mappers focus creates a **fundamental mismatch,** yet the subset of valid, functionally **correct mapping is exceptional sparse.**
- Some mapping algorithm relies on mathematical optimization or heuristic approaches.
  - Respectively, we have CGRA-ME's ILP mapper and RAAP-CGRA.
  - Also often **fail** due to the dense connectivity patterns of these kernels unfortunately.



irregularly-shaped DFG



GeMM DFG

RIKEN CGRA

- **High-performance, elastic CGRA** optimized for **HPC/AI workloads.**
  - Available as an example architecture in the **CGRA-ME** framework.
- **Composed** of three **specialized** tile types:
  - **Load/Store (LS) tiles** for memory operations.
  - **Processing Element(PE)** tiles featuring FMA FPU.
  - **Switch Block (SB)** tiles for intra-CGRA routing.
- **Elastic architecture isolates** the P&R challenge from the complexities of static scheduling
  - Thus, **easier for the mappers to find a valid spatial solution,** as they are not constrained by timing requirements associated with static CGRA.

Methodology & Experimental Setup

- **We evaluated the efficacy of four place-and-route compilers:**
  - CluMap, GenMap, RAAP-CGRA, ILP mapper of CGRA-ME.
  - **RIKEN CGRA** as the target architecture.
- **CluMap decouples** the steps of **P&R** that most elastic CGRA mappers perform simultaneously.
  - Use a Simulated Annealing algorithm for placement.
  - Use a Pathfinder algorithm for routing.
- The separation allowed us to conduct a **targeted experiment**.
  - We provided CluMap with the ideal, hand-placed operator mapping to indepently iisolate and evaluate the efficacy of its routing algorithm.

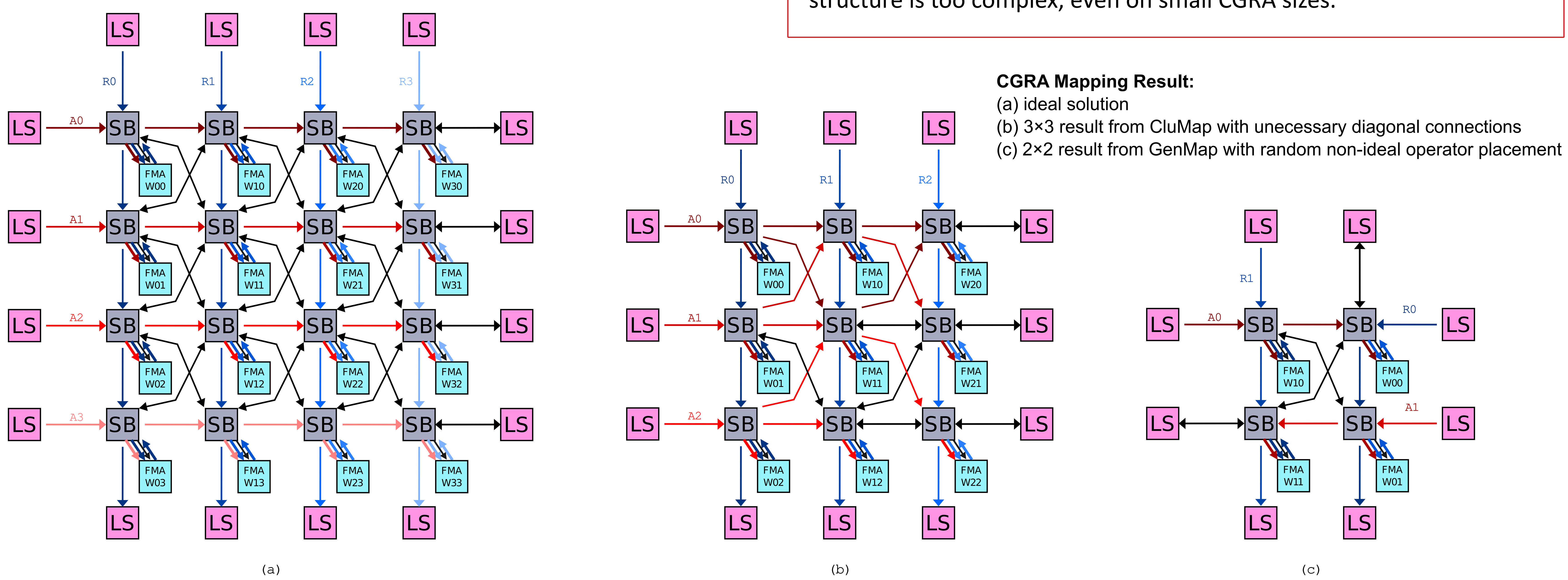
Analysis & Discussion

- **All** evaluated mappers are unable to scale:
  - **Failing** on relatively small CGRA sizes **far below** the 16 × 16 or larger configurations.
- Figure (a) serves as a reference, an **idealized**, handcrafted GeMM mapping to identify specific limitations.
  - **Diagonal connections** between **SBs** are not used.
- Figure (b) shows a valid mapping generated by GenMap on a minimal 2 × 2 CGRA.
  - The stochastic nature of its genetic algorithm is evident in the non-optimal placement.
  - As the array size increases, **the probability** of this approach converging on the highly-structured placement required by GeMM **diminishes rapidly**, leading to compilation failure.
- Figure (c) illustrates our pre-placement experiment where CluMap's Pathfinder routers was tasked with routing an ideal 3 × 3 placement.
  - The router created **unnecessarily circuitous diagonals routes** while it found a valid solution.
  - In an elastic fabric, these detours do not impact latency as the datapath still has the same number of hops. However, eventually the random routes can cause resource contention on specific SB tiles, resulting in failed routing.

This experiment demonstrates that the randomness central to these mappers, while effective for irregular DFGs, are fundamentally ill-suited for converging on the sparse, highly-regular solutions required by systolic-array-style kernels.
Meanwhile, ILP mapper and RAAP-CGRA fail simply because the DFG structure is too complex, even on small CGRA sizes.

Makeshift plan & conclusion

- There is also a way to proceed with research in advance by analyzing the bitstream and manually constructing it, in addition to mapping.
- We analyzed the bitstream files generated by CGRA-ME and directly utilized the Verilog files of the generated CGRA. In addition, we generated GeMM-specific bitstreams suitable for those Verilog files using a dedicated algorithm, since the ideal GeMM mapping had already been determined.
- However, this approach requires separate analysis for each different mapper or framework.
- In this poster, we explores the failure mode of existing SoTA mappers laying the groundwork for future mapping algorithm research.



CGRA Mapping Result:
(a) ideal solution
(b) 3×3 result from CluMap with unecessary diagonal connections
(c) 2×2 result from GenMap with random non-ideal operator placement