

DyNAMoE: Dynamic Reconfigurable NoC-based Accelerator for Mixture-of-Expert Models

Mohit Upadhyay and Li-Shiuan Peh

School of Computing, National University of Singapore

mohitu@u.nus.edu, peh@nus.edu.sg

Abstract—Our characterization of MoE execution on GPUs revealed that while GPUs can parallelize expert execution well, expert index computation and routing of MoE inputs lead to bottlenecks. This work introduces DyNAMoE, an accelerator designed to map and execute MoE layers efficiently for increased performance and energy efficiency. Specifically, DyNAMoE proposes specialized dynamically reconfigurable NoCs for routing tokens, distributing inputs and weights and reducing results at runtime for accelerating MoE layers. Our results show DyNAMoE realizing more than $40\times$ faster latency than edge GPUs and more than $13.7\times$ faster than statically scheduled systolic array architectures.

Index Terms—mixture-of-experts, network-on-chip, reconfigurable architecture

I. INTRODUCTION

With neural networks widely deployed [10], [13], Mixture-of-Expert (MoE) models have grown in complexity and computational demand. An MoE layer uses a subset of small networks (experts) selected by a gating function [17], creating a performance bottleneck in GPUs and motivating specialized hardware accelerators for MoEs.

GPUs are inefficient for MoEs, as a big fraction of inference time is spent on expert selection. Our experiments showed that despite expert parallelism within FastMoE [7] and Tutel [8] frameworks, the gating function takes 28.5%, 63.6% and 14.1% of compute time using native Pytorch, FastMoE and Tutel respectively. Existing accelerators [2], [9] with statically mapped dataflow are not suited for MoEs. FPGA-based MoE acceleration face challenges in energy efficiency versus ASICs [4], [11], [16], such as Space-Mate [12] which is specialized for SLAM-based workloads.

In this paper, we propose DyNAMoE, a dynamically reconfigurable MoE accelerator enabling high performance and energy efficient conditional selection and execution of experts at runtime. Hence, DyNAMoE runs MoEs more than $10\times$ and $39.5\times$ faster than MI210 GPU and Ampere Edge GPU respectively, and $13.7\times$ faster than systolic array architectures. It also delivers $3.05\times$ better power efficiency at a similar performance than an FPGA-based design [16].

II. DYNAMO E ARCHITECTURE AND MAPPER

A. DyNAMoE accelerator architecture

We propose DyNAMoE, an accelerator with dynamic NoCs for steering of MoE tokens and parameters to a MAC (multiply-accumulate) array for MoE computations (see Figure 1). We briefly explain each component next.

1) *GEMM cores*: These are the compute hardware for matrix computations similar to many AI accelerators [3], [5], [6], [18], for the matrix-vector computations using 256 MAC units and associated control logic fed by two 256×256 16-bit weights SRAM banks each. While the current token in one bank is being computed, the other preloads parameters for the next token.

2) *DyNAMoE's On-chip Global Buffer*: DyNAMoE has a 16MB on-chip global buffer (4 banks of 4MB dual ported SRAM) to store expert parameters and inputs during the MoE inference to hide compute pipeline stalls due to PCIe latency.

3) *DyNAMoE's NoCs (Redux, Distro and Weight Distro NoCs)*: DyNAMoE routers support two modes: header mode (for router reconfiguration) and data mode (for data transfer). Table I shows the packet encoding for both modes, supporting unicast (UC) and multicast (MC). The wide link width (256×16 bits) allows multiple header packets' transfer in parallel.

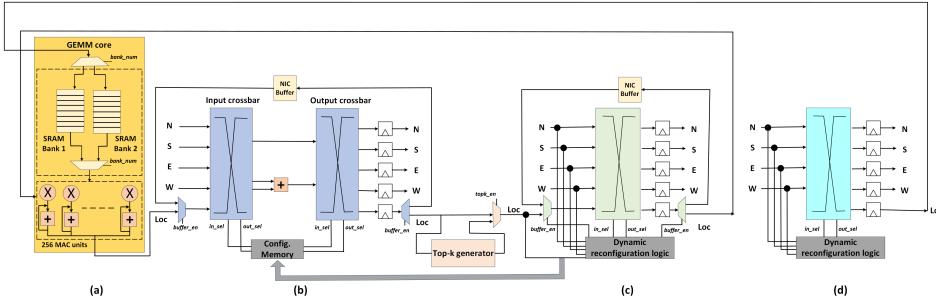
Each Distro router dynamically *scatters* MoE inputs using 5×5 crossbars connecting neighboring routers and the local port. In header mode, each packet is used for reconfiguration. In data mode, the MoE data traverses the path configured in the header mode. Since experts are only known at runtime, Distro routers route MoE data across different cores at runtime. The Weight Distro router has the same design but uses a 4×5 crossbar, as there is no partial sum in weight distribution.

The Redux router dynamically accumulates expert outputs within MoE layers. It employs a 5×3 input and a 2×5 output crossbar with an adder within their datapath. The output ports of the input crossbar feed into the adder or the bypass path when addition is not required, with the reconfiguration similar to the Distro router.

Dynamic router reconfiguration: Distro and Weight Distro NoCs share routing tables using the header packets. The mapper ensures the same dataflow within the Distro and Redux router, but in the *reverse* direction, to reuse the identical configuration bits. Upon receiving a header at each router per turn, 4 most significant bits (MSB) set the crossbar switches. For every source-destination-pair, The configuration bits are set for every router along with support for in-network multicast for concurrent multiple expert selection.

B. Mapping MoEs on DyNAMoE

DyNAMoE's software mapper maps multiple gating function instances on DyNAMoE by generating multiple expert IDs per unique input token. Experts for the subsequent tokens are prefetched during current token inference, hiding I/O latency. The source and destination router pairs are decided at runtime



Fields	Per-neuron bit width		Total bit width
	MC/UC	Output port numbers	
Width	1 bit	15 bits (3 bits per output port)	
Header mode (Unicast)	0	010	16×256 bits
Header mode (Multicast)	1	{010, 011, ...}	16×256 bits
Data mode	16 bits		16×256 bits

Fig. 1: Microarchitecture of the (a) GEMM core (b) Redux, (c) Distro router and (d) Weight Distro router

TABLE I: Packet encoding format

by the hardware controller based on the mapping algorithm. The gating function and the expert parameters are transferred through the PCIe interconnect after the expert IDs generation, followed by each batch of tokens.

III. EVALUATION

DyNAMoE is synthesized at 22nm with wire-modeled SRAMs and Cacti [1]-modeled global buffer at 0.8V and 300 MHz while the Top-K generator runs at 30 MHz due to its critical path. The performance and memory accesses are computed on custom Python-based functional simulator with performance, power and energy evaluations compared against the Ampere GPU and AMD MI210.

A. DyNAMoE Performance Evaluation Results

1) *DyNAMoE architecture performance*: The GPU latency in Table II is measured for MoE layers' runtime with FastMoE, Tutel frameworks and native Pytorch on the MI210 and Ampere GPUs, with results from the best performing frameworks (Tutel on MI210 and FastMoE on Ampere) shown.

Workload	MI210 Latency (μ s) [1700 MHz]	Ampere GPU Latency (μ s) [300 MHz]	DyNAMoE Latency (μ s) [300 MHz]
M ³ ViT	2864.499	15089.06	568.8
CIFAR-10 MoE	1614.548	5651.096	65.92
Switch-base-8	2655.497	19908.576	3258.28

TABLE II: Latency results (Batch size is 16)

The overall compute utilization for DyNAMoE is 55%, much higher than MI210 GPU (with Tutel framework). The Top-K expert generation and input routing consumes 1.8% of total compute time in DyNAMoE, much lower than the 35.4% of total compute time on GPUs.

Workload	M ³ ViT (GPU:Tutel)	M ³ ViT (DyNAMoE)	CIFAR-10 (GPU:Tutel)	CIFAR-10 (DyNAMoE)	Switch-base-8 (GPU:Tutel)	Switch-base-8 (DyNAMoE)
Gating function (Expert generation + Top-K)	428.48	8.5	267.04	1.733	286.88	23.49
Expert execution	2548.019	560.3	1347.508	64.187	2368.617	3234.79
Total latency	2864.499	568.8	1614.548	65.92	2655.497	3258.28
DyNAMoE Speedup	5.03×	1×	24.49×	1×	0.815×	1×

TABLE III: Performance breakdown of MoE models on GPUs and DyNAMoE

2) *Performance of DyNAMoE vs. systolic array accelerators*: DyNAMoE was compared against three baseline systolic arrays (all with 4K MAC units and 16 MB memory) using the SCALE-Sim toolchain [15]. Our results show that DyNAMoE

runs MoE $13.7\times$ faster than statically scheduled systolic architectures on average (Table IV).

Workloads	TPU-like	ShiDianna-like	Systolic Array-IS	DyNAMoE
M ³ ViT	8824.7	3925.7	4437.5	568.8
CIFAR-10 MoE	496.54	272.34	357.74	65.92
Switch-base-8	140918.04	52058.76	54575.64	3258.28

TABLE IV: Performance comparison of Systolic Array architectures versus DyNAMoE (Latency in μ s)

B. DyNAMoE Architecture Synthesis Results

1) *Area*: Table V shows the area breakdown within a DyNAMoE core. The 4×4 DyNAMoE configuration has a total area of around 78 mm^2 with the on-chip global buffer taking 57.27 mm^2 .

2) *Power*: Table V shows the GEMM core power distribution. The 4×4 DyNAMoE consumes around 4.79W, dominated by the GEMM core. For comparison, MI210 GPU and Ampere GPU consumes 45W and around 3W, respectively measured at the board level.

Sub-module	Active Power (mW)	Active Power (% per core)	Area (mm^2)	Area (% per core)
GEMM core	236.04	79.2	1.155	87.78
Distro router	21.162	7.12	0.051	3.87
Weight Distro router	15.98	5.36	0.0417	3.18
Redux router	23.77	7.98	0.0474	3.6
Top-K generator	0.877	0.32	0.0206	1.57

TABLE V: Synthesis results for each DyNAMoE core

C. *Energy consumption results*

DyNAMoE achieves over $48.7\times$ average energy savings over TPU-like architecture (RTL obtained from [14]) since the systolic arrays process experts sequentially per sample, reducing overall hardware utilization and increasing compute latency and energy.

IV. CONCLUSION

This work proposes DyNAMoE, a NoC based accelerator which supports dynamic routing that allows for the efficient mapping and execution of MoE layers. Our results show the DyNAMoE architecture being more than $40\times$ faster than edge GPUs, with comparable power consumption.

V. ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Singapore (NRF-CRP23-2019-0003) and the Ministry of Education, Singapore (MOE-MOET32024-0003).

REFERENCES

- [1] R. Balasubramonian *et al.*, “Cacti 7: New tools for interconnect exploration in innovative off-chip memories,” *ACM Trans. Archit. Code Optim.*, vol. 14, no. 2, jun 2017. [Online]. Available: <https://doi.org/10.1145/3085572>
- [2] Y. Chen *et al.*, “Eyeriss: An energy-efficient reconfigurable accelerator for deep cnn,” *IEEE JSSC*, 2017.
- [3] S. Choi *et al.*, “Trainware: A memory optimized weight update architecture for on-device convolutional neural network training,” in *Proceedings of the International Symposium on Low Power Electronics and Design*, ser. ISLPED '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: <https://doi.org/10.1145/3218603.3218625>
- [4] J. Dong *et al.*, “Ubimoe: A ubiquitous mixture-of-experts vision transformer accelerator with hybrid computation pattern on fpga,” 2025. [Online]. Available: <https://arxiv.org/abs/2502.05602>
- [5] Z. Du *et al.*, “Shidiannao: Shifting vision processing closer to the sensor,” in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 92–104.
- [6] S. Han *et al.*, “EIE: efficient inference engine on compressed deep neural network,” *CoRR*, vol. abs/1602.01528, 2016. [Online]. Available: <http://arxiv.org/abs/1602.01528>
- [7] J. He *et al.*, “Fastmoe: A fast mixture-of-expert training system,” 2021.
- [8] C. Hwang *et al.*, “Tutel: Adaptive mixture-of-experts at scale,” *CoRR*, vol. abs/2206.03382, Jun. 2022. [Online]. Available: <https://arxiv.org/pdf/2206.03382.pdf>
- [9] N. P. Jouppi *et al.*, “In-datacenter performance analysis of a tensor processing unit,” *SIGARCH Comput. Archit. News*, vol. 45, no. 2, p. 1–12, Jun. 2017.
- [10] A. Kirillov *et al.*, “Segment anything,” 2023.
- [11] H. Liang, Z. Fan, R. Sarkar, Z. Jiang, T. Chen, K. Zou, Y. Cheng, C. Hao, and Z. Wang, “M³vit: Mixture-of-experts vision transformer for efficient multi-task learning with model-accelerator co-design,” 2022.
- [12] G. Park, S. Song, H. Sang, D. Im, D. Han, S. Kim, H. Lee, and H.-J. Yoo, “20.8 space-mate: A 303.5mw real-time sparse mixture-of-experts-based nerf-slam processor for mobile spatial computing,” in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67, 2024, pp. 374–376.
- [13] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, “Hierarchical text-conditional image generation with clip latents,” 2022.
- [14] A. Samajdar, “Tpu: Tensor-processing-unit,” <https://github.com/scalesim-project/scale-sim-v2/tree/main/code-examples/systolic-array-rtl>, 2021.
- [15] A. Samajdar *et al.*, “A systematic methodology for characterizing scalability of dnn accelerators using scale-sim,” in *ISPASS 2020*, 2020, pp. 58–68.
- [16] R. Sarkar *et al.*, “Edge-moe: Memory-efficient multi-task vision transformer architecture with task-level sparsity via mixture-of-experts,” 2023.
- [17] N. Shazeer *et al.*, “Outrageously large neural networks: The sparsely-gated mixture-of-experts layer,” *CoRR*, vol. abs/1701.06538, 2017. [Online]. Available: <http://arxiv.org/abs/1701.06538>
- [18] M. Upadhyay *et al.*, “REACT: A heterogeneous reconfigurable neural network accelerator with software-configurable nocs for training and inference on wearables,” in *DAC*, 2022.