Klotski: DNN Model Orchestration Framework for Dataflow Architecture Accelerators

Chen Bai[1,3] Xuechao Wei[3] Youwei Zhuo[3] Yi Cai[3] Hongzhong Zheng[3] Bei Yu[1] Yuan Xie[2,3]

[1]The Chinese University of Hong Kong
[2]Hong Kong University of Science and Technology
[3]DAMO Academy, Alibaba Group

Introduction

AI accelerators scaling trends:

- "Brawny" scaling: Scale on-chip hardware resources.
- Scalable scaling: Scale DNN accelerators via an network-on-chip (NoC).

What is **dataflow architecture acclerators**?

- Dataflow architecture accelerators are a new kind of scalable scaling-driven AI accelerators.
- Distinct execution model compared to traditional scalable DNN accelerators.

Dataflow Execution Model

The executability and execution of instructions is solely determined based on the availability of input operands to the instructions.

What is **DNN model orchestration**?

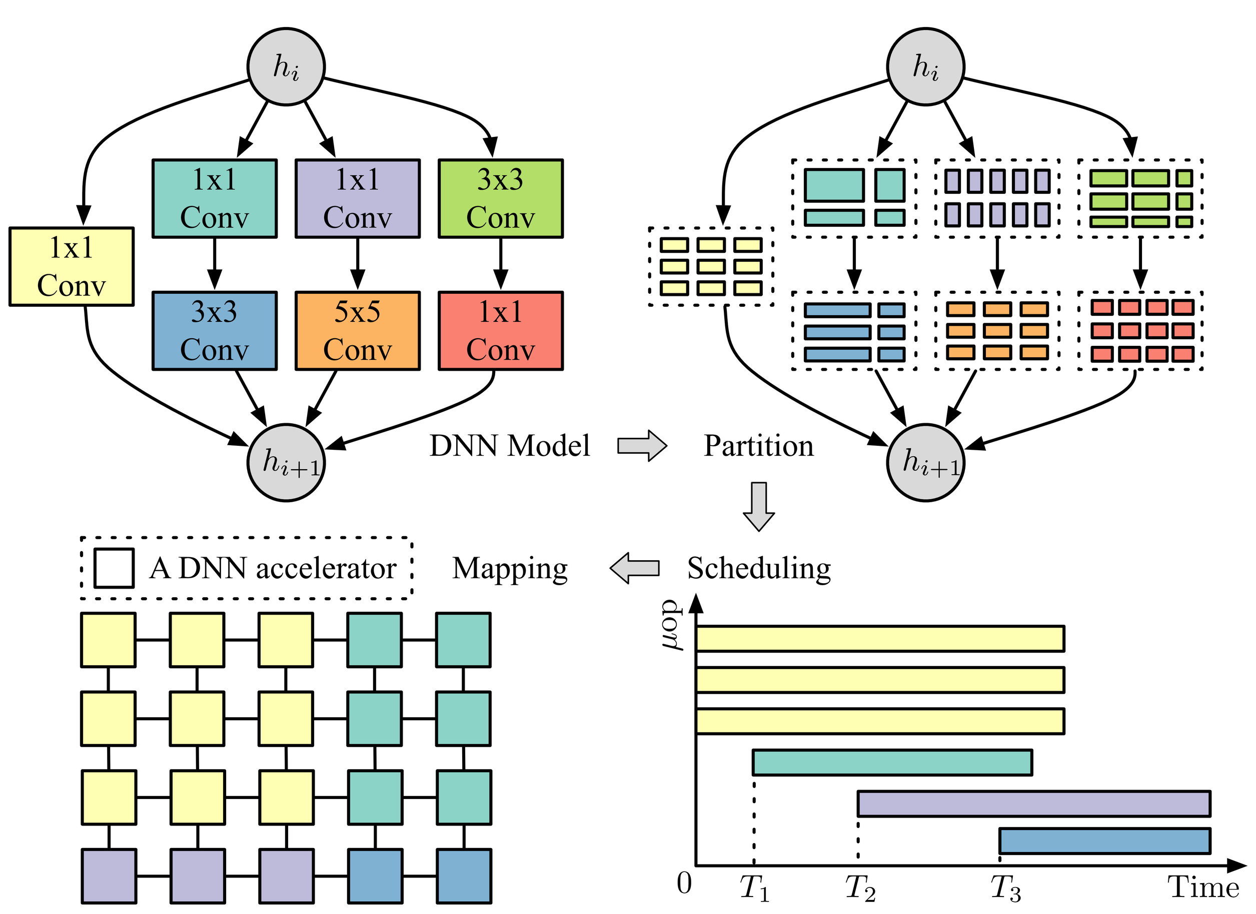

Figure 1. A pipeline overview of DNN model orchestration for scalable DNN accelerators.

Previous Methodologies & Limitations

- CNN-Partition.
- Tangram.
- Atomic dataflow.

They are proposed for traditional scalable DNN accelerators rather than dataflow architecture accelerators.

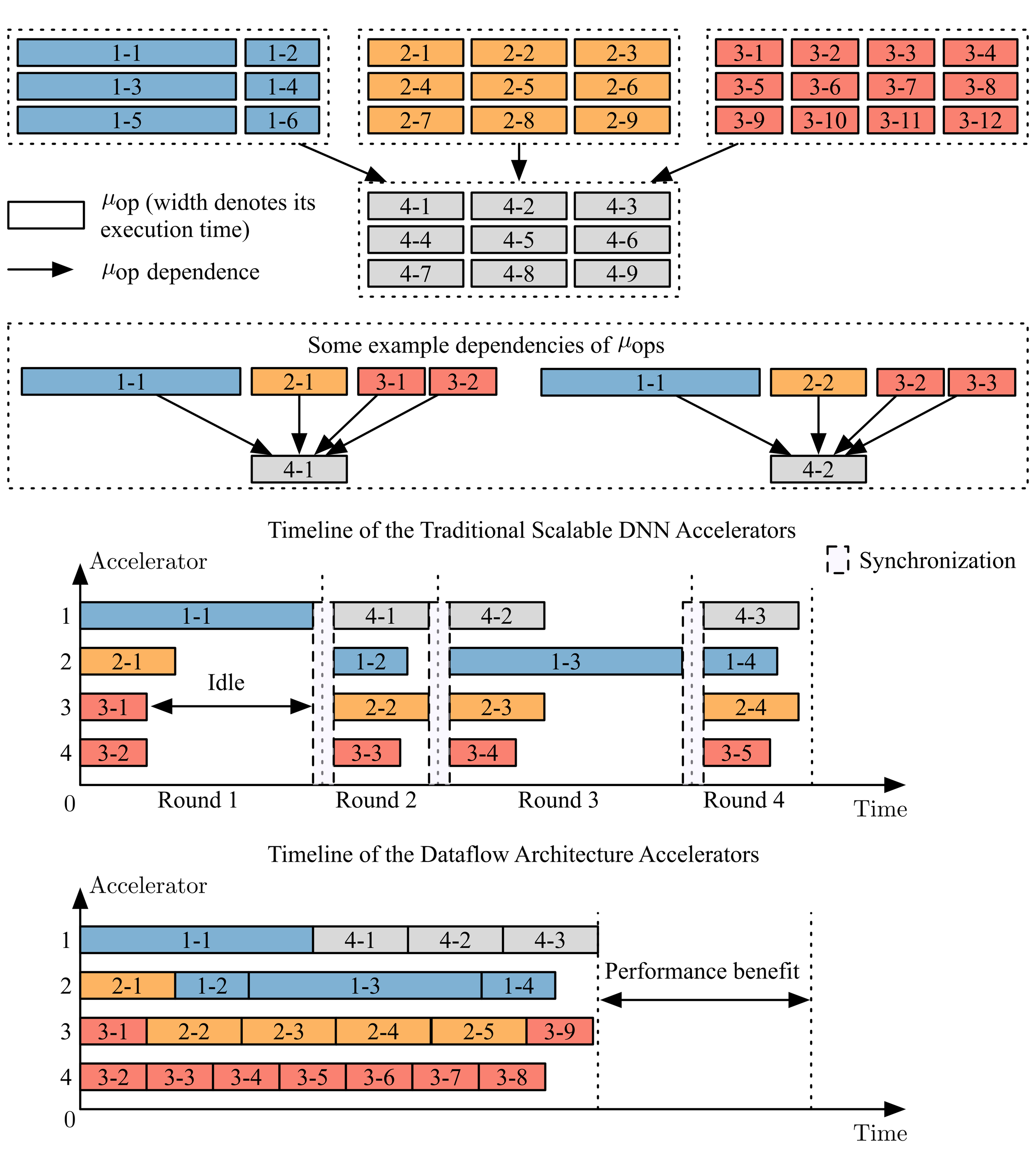

Figure 2. Comparison between traditional scalable DNN accelerators and dataflow architecture accelerators.

Algorithms

Overview of Klotski

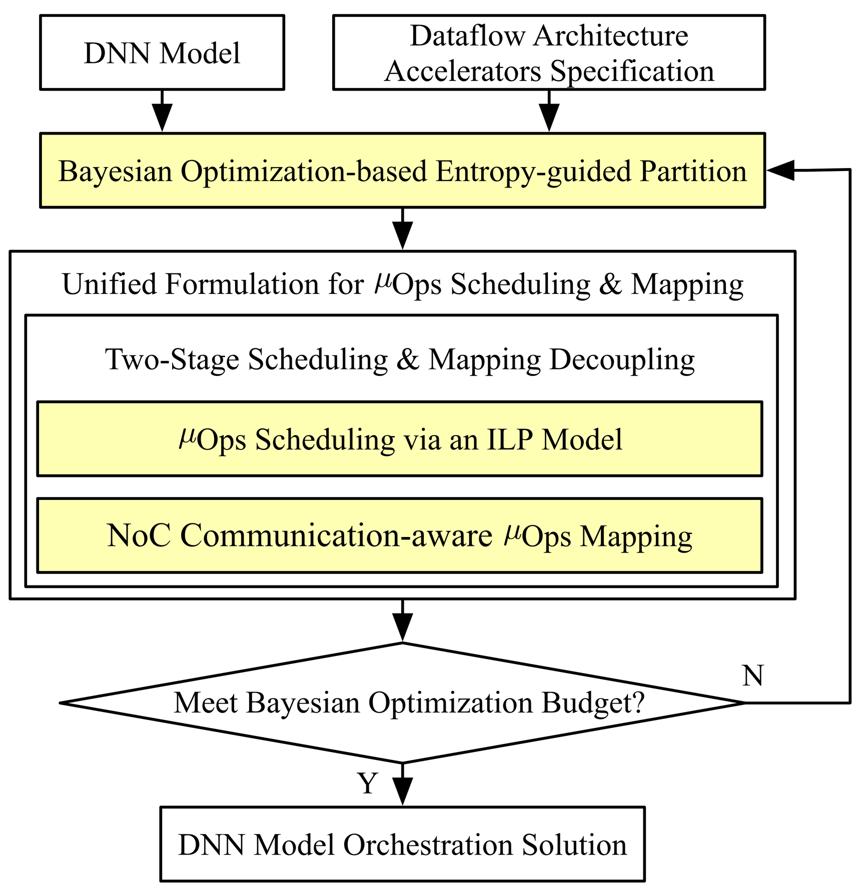

Figure 3. An overview of Klotski framework.

Bayesian Optimization-based Entropy-guided Partition

Algorithm 1 BO-based Entropy-guided Partition

Require: G: a DNN model. $\mathbb{D}$: the design space for $\boldsymbol{s}$. T: optimization budget.
$S = \emptyset$; Sample $\boldsymbol{s} \in \mathbb{D}$;
for $i = 1 \rightarrow T$ **do**
  Partition G with $\boldsymbol{s}$;
  Schedule, map, and execute μops;
  Evaluate $E(\boldsymbol{s})$; ▷ Equation ??
  $S = S \cup \{(\boldsymbol{s}, E(\boldsymbol{s}))\}$;
  Construct a Gaussian process model with S;
  $\boldsymbol{s}^* = \mathrm{argmax}_{\boldsymbol{s}\in\mathbb{D}}$ UCB$(\boldsymbol{s})$; $\boldsymbol{s} = \boldsymbol{s}^*$
end for
return Optimal $\boldsymbol{s}^*$ from S.

$$E(\boldsymbol{s}) = -(\sum_{u_i \in V} \frac{l(u_i)}{l(V)} \ln \frac{l(u_i)}{l(V)})/(\alpha \cdot \text{makespan}), \tag{1}$$

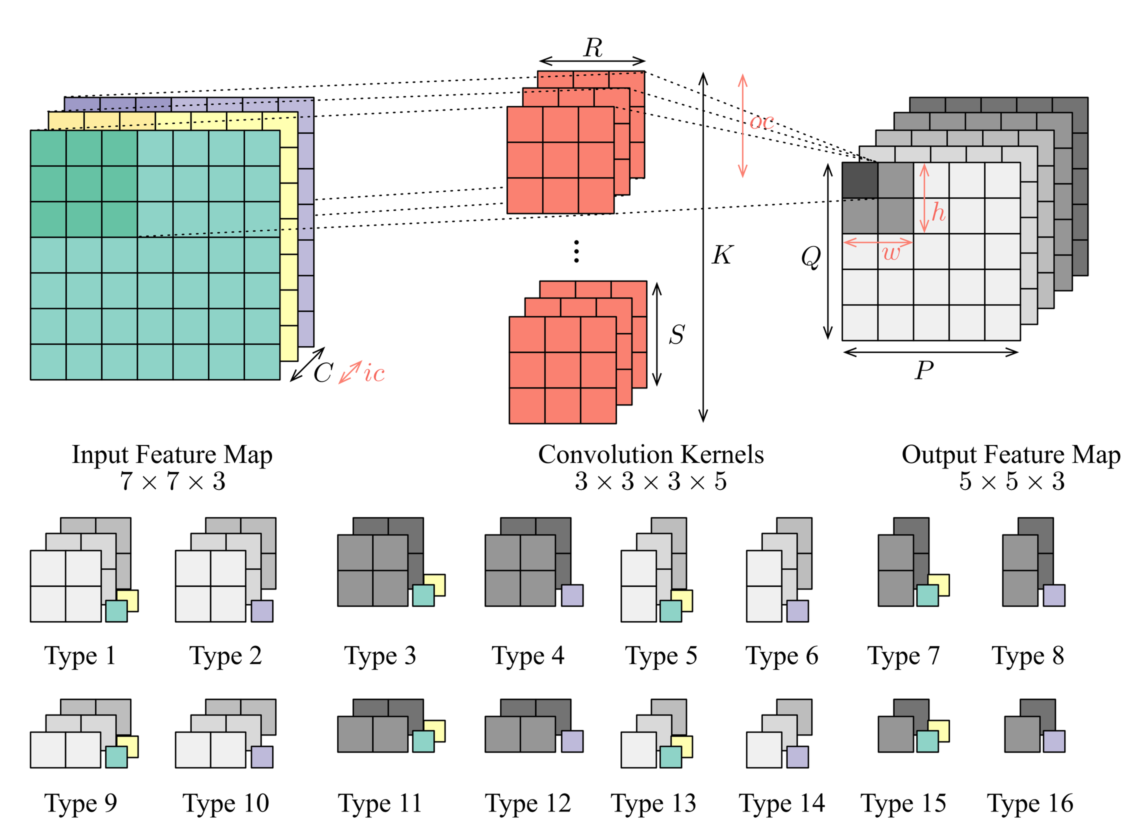

Figure 4. An example shows a partition with $\boldsymbol{s}(2, 2, 2, 3)$.

Unified Formulation for μOps Scheduling & Mapping

1. Acquire the upper bound of the makespan by list scheduling.
2. Acquire the scheduling flexibility by ASAP & ALAP.
3. Define the solution with a binary tensor $\boldsymbol{\mathcal{X}}$.
4. Construct constraints for the scheduling & mapping.
5. Construct optimization objectives.
6. Solve the model with off-the-shelf solvers.

$$\boldsymbol{\mathcal{X}}_{ijk} = \begin{cases} 1, \mu\text{op } u_i \text{ is scheduled to the } k\text{-th accelerator} \\ \quad \text{at the } j\text{-th time slot.} \\ 0, \text{otherwise.} \end{cases} \tag{2}$$

Limitations of the unified formulation:

- It costs high runtime to construct constraints.
- Non-linearity in NoC communication formulations.

Two-Stage Scheduling & Mapping Decoupling

$$\boldsymbol{X}_{ij} = \begin{cases} 1, \mu\text{op } u_i \text{ is scheduled to the } j\text{-th time slot.} \\ 0, \text{otherwise.} \end{cases} \tag{3}$$

Results

- We build an in-house simulator for the dataflow architecture accelerators.
- We use MAESTRO as performance model for individual accelerators.
- We use *nn_dataflow* as the front end of DNN models, and we implement the partition based on the framework.
- We use Gurobi v10.0 as the off-the-shelf solver.

Comparison to Baselines:

Table 1. The experimental results for the 3×3 topology

Workload	Method	Cycles	Ratio	Overall Runtime	Ratio	HUR [1]
VGG16	Baseline 1	1.2283E + 08	1.0000	−− [2]	−−	1.0000
	Baseline 2	5.5633E + 07	0.4529	477.6634	1.0000	2.5617
	Klotski	4.0659E + 07	0.3310	878.8832	1.8399	3.0602
VGG19	Baseline 1	1.5523E + 08	1.0000	−−	−−	1.0000
	Baseline 2	7.4207E + 07	0.4781	576.3081	1.0000	2.5229
	Klotski	5.5381E + 07	0.3568	887.5790	1.5401	2.9857
ResNet50	Baseline 1	7.7422E + 07	1.0000	−−	−−	1.0000
	Baseline 2	5.7060E + 07	0.7370	583.6488	1.0000	0.9762
	Klotski	4.8174E + 07	0.8443	1779.0426	3.0481	1.3050
ResNet152	Baseline 1	1.8984E + 08	1.0000	−−	−−	1.0000
	Baseline 2	1.7102E + 08	0.9009	867.0853	1.0000	1.2523
	Klotski	1.5947E + 08	0.8400	2800.9154	3.2302	1.3605
Inception	Baseline 1	2.5122E + 07	1.0000	−−	−−	1.0000
	Baseline 2	1.6345E + 07	0.6506	470.3763	1.0000	2.5103
	Klotski	1.3348E + 07	0.5313	1397.9008	2.9719	3.2996

[1] Hardware utilization ratio
[2] Not applicable

Table 2. The experimental results for the 4×4 topology

Workload	Method	Cycles	Ratio	Overall Runtime	Ratio	HUR
VGG16	Baseline 1	1.2283E + 08	1.0000	−−	−−	1.0000
	Baseline 2	4.5869E + 07	0.3734	317.5903	1.0000	2.1196
	Klotski	3.0670E + 07	0.2497	881.6310	2.7760	2.4547
VGG19	Baseline 1	1.5523E + 08	1.0000	−−	−−	1.0000
	Baseline 2	5.8049E + 07	0.3740	388.8627	1.0000	1.9895
	Klotski	3.9934E + 07	0.2573	1130.6444	2.9076	2.2964
ResNet50	Baseline 1	7.7422E + 07	1.0000	−−	−−	1.0000
	Baseline 2	5.3365E + 07	0.6893	541.8091	1.0000	2.8954
	Klotski	4.6260E + 07	0.5975	1019.2198	1.8811	3.1953
ResNet152	Baseline 1	1.8984E + 08	1.0000	−−	−−	1.0000
	Baseline 2	1.6578E + 08	0.8733	793.7304	1.0000	1.2264
	Klotski	1.5754E + 08	0.8299	2327.4657	2.9323	1.3438
Inception	Baseline 1	2.5188E + 07	1.0000	−−	−−	1.0000
	Baseline 2	1.5183E + 07	0.6028	419.3479	1.0000	2.2822
	Klotski	1.0781E + 07	0.4280	1432.0112	3.4148	2.8579

Ablation study:

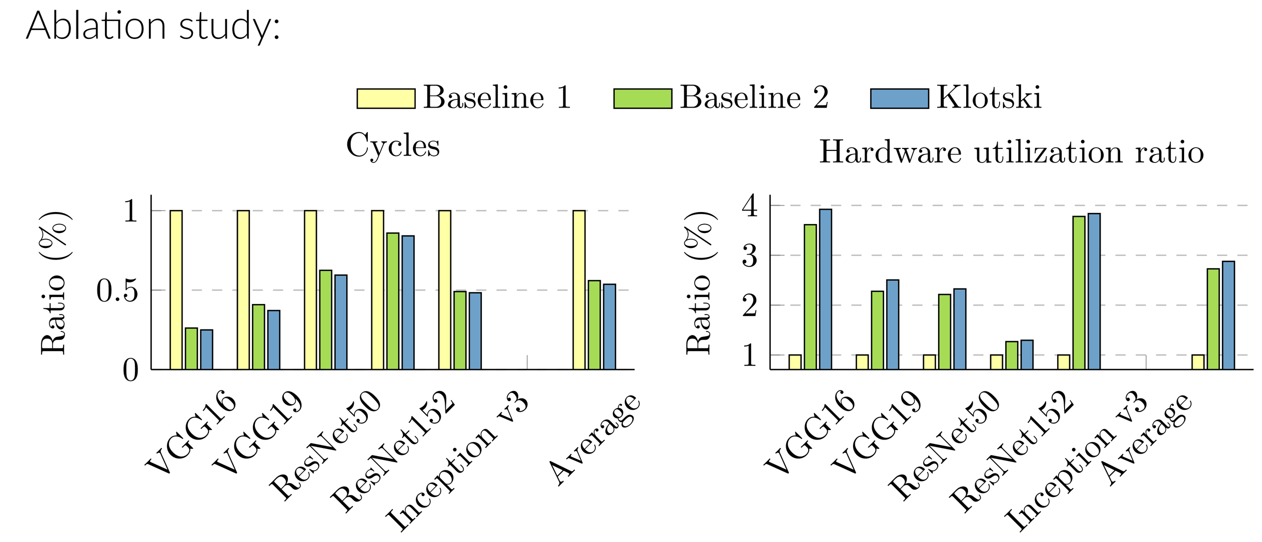

Figure 5. Results of the 3×3 topology.

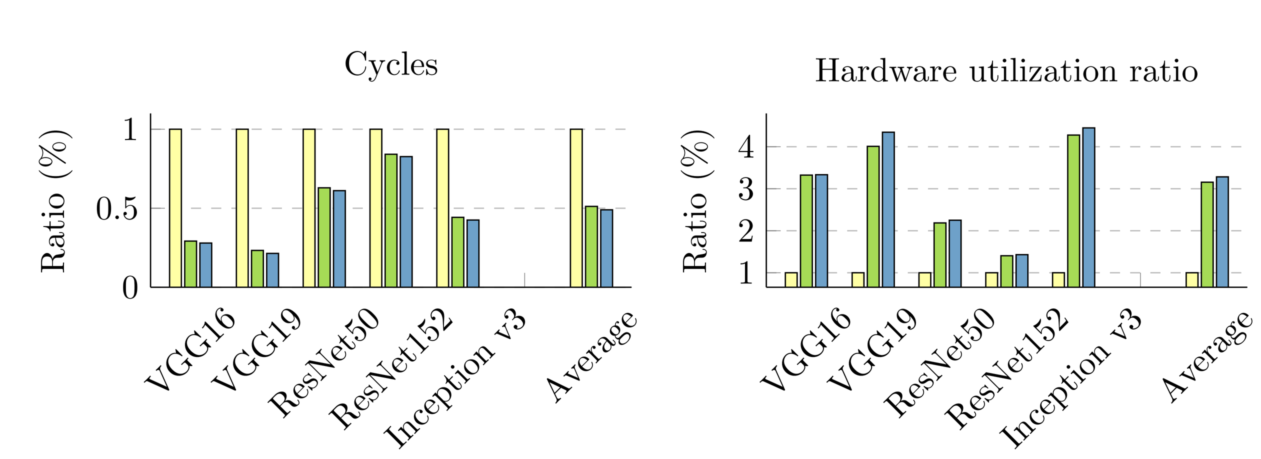

Figure 6. Results of the 4×4 topology.

Summary of results:

- Across all different scales of topologies, compared to baseline 1 and baseline 2, the solution given by Klotski outperforms by an average of 48.65% and 9.55% in cycles.
- Klotski costs higher runtime than baselines due to that Klotski leverages much time to solve the scheduling and mapping in the two-stage methodology.

IEEE/ACM International Conference on Computer-Aided Design (ICCAD) 2023, San Francisco