MAGIS: Memory Optimization via Coordinated Graph Transformation and Scheduling for DNN

Renze Chen, Zijian Ding, Size Zheng, Chengrui Zhang,
Jingwen Leng, Xuanzhe Liu, Yun Liang

E-mail: crz@pku.edu.cn Code: https://github.com/pku-liang/MAGIS

Scan to access our code

1. Motivation

Memory Pressure

- Large **tensor-sizes**
  - Large hidden-size
  - Large batch-size
  - Long seq-length
- Long **tensor-lifetimes**
  - Forward activation reused in backward
  - Tensor reused after long skip-connection

Graph Scheduling

Manage **tensor-lifetimes**

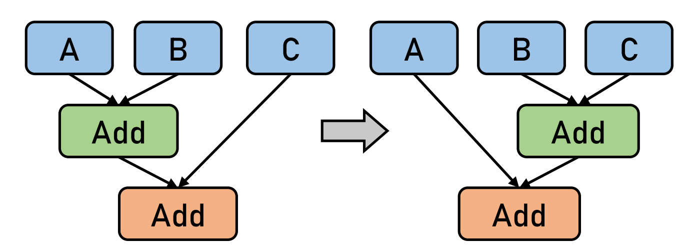
- Re-reordering: permute op
- Re-materialization: discard, re-compute op
- Swapping: swap-in/out op

Memory↓ Latency↑

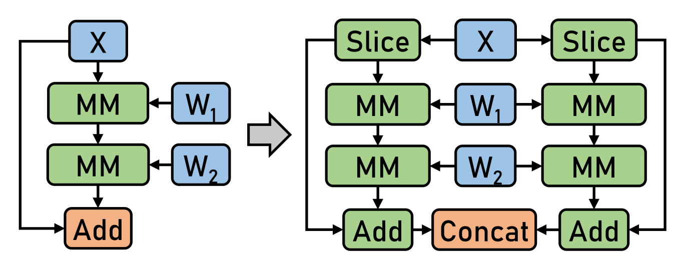
Graph Transform

Mutate graph / **tensor-sizes**

Interim Transform (I-Trans)



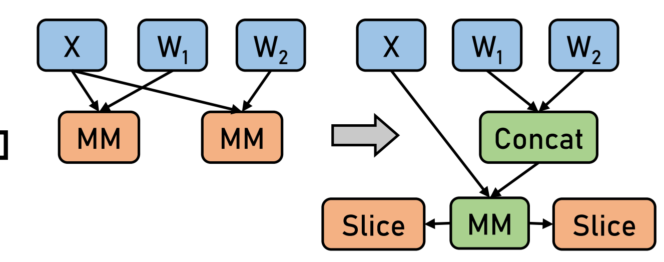
Fission Transform (F-Trans)



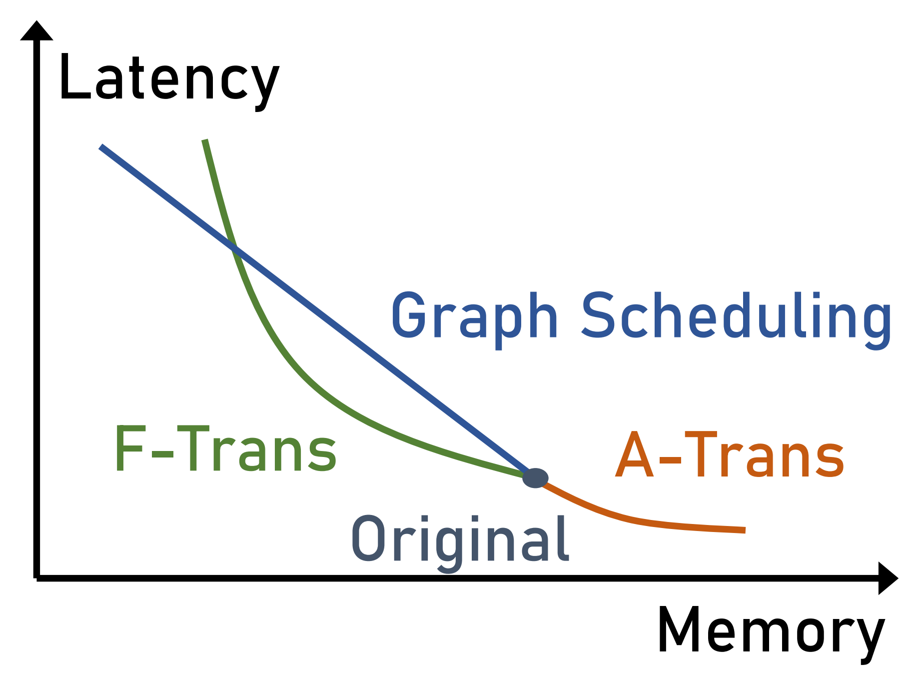
Memory↓ Latency↑

Dual

Aggregation Transform (A-Trans)



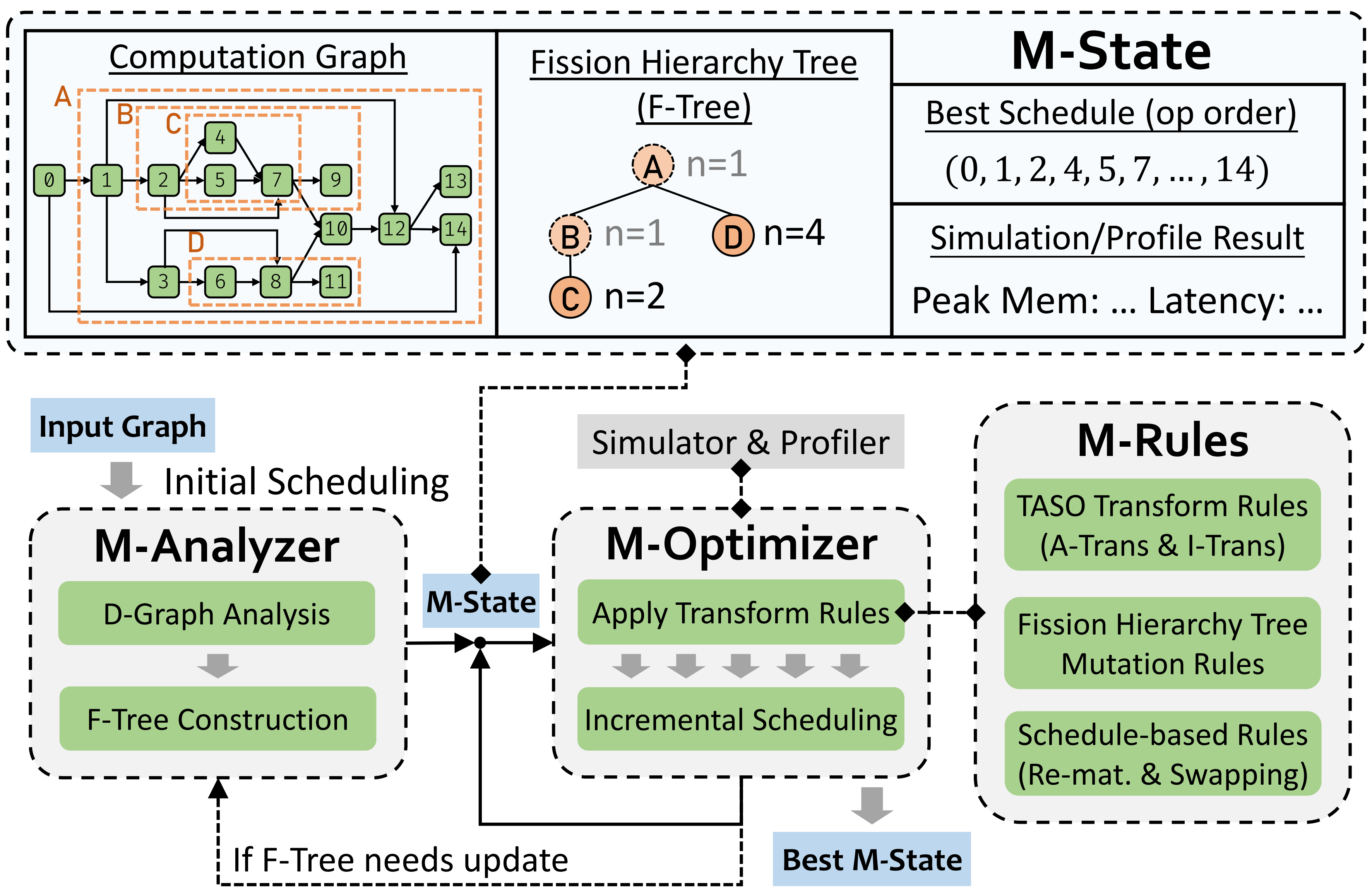
Memory↑ Latency↓



Graph Transform provides additional trade-off space for memory & latency, enhancing the capability of Graph Scheduling for memory optimization.

2. Overview

MAGIS optimizes DNN memory/latency under a latency/memory constraint.



3. M-Analyzer

Dimension Graph (D-Graph)

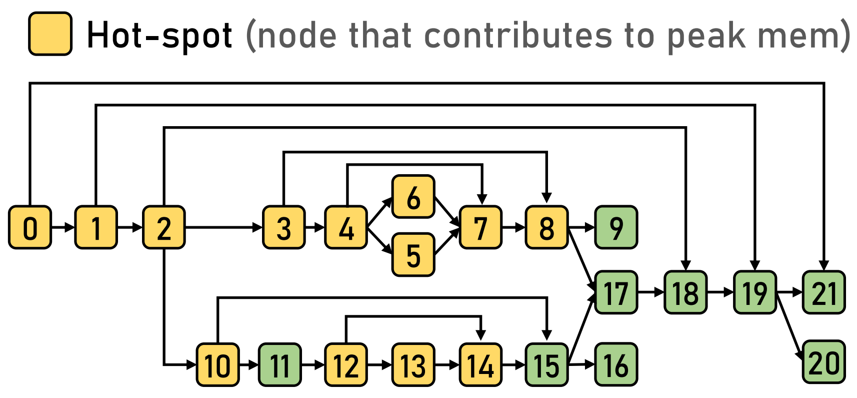
- Each node represents a spatial/reduction-dim of an operator.
- Each edge represents a dim mapping between adjacent ops.
- Connected sub-graph of D-Graph represents graph-level dim.

Fission Transformation (F-Trans)

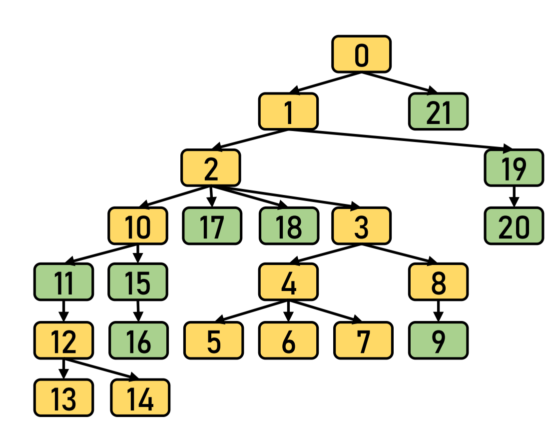
Fission Transformation $f = (S, D, n)$ is defined as:
splitting sub-graph S (of graph G) along D-Graph D into n partitions

Fission Hierarchy Tree (F-Tree)

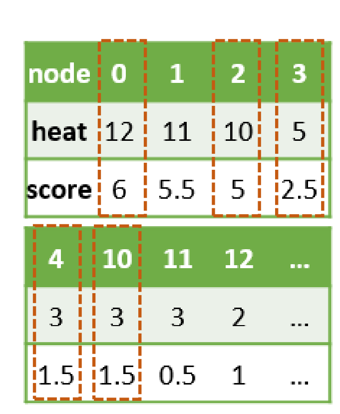
- *To avoid increasing graph complexity*, instead of directly rewriting graph, we record F-Trans definitions and construct tree structure based on subgraph containment relations.
- *To prevent vast search space*, we analyze the "heat" (peak memory contribution) and "score" (minimum peak memory reduction after fission) of the subgraph dominated by every single node to select some subgraphs to build F-Tree as the lightweight search space for Fission Transformation.



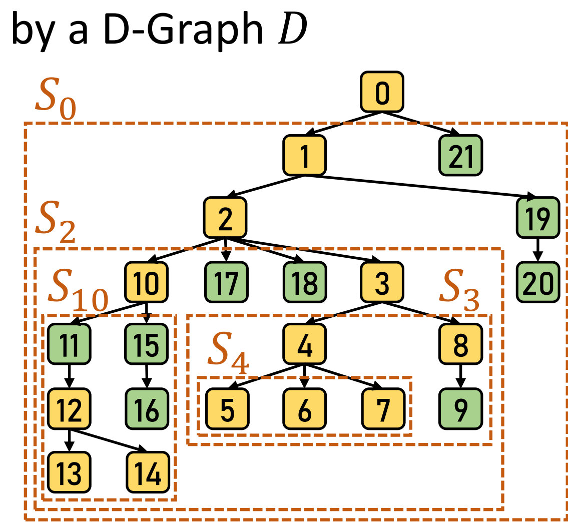
A subgraph S covered by a D-Graph D



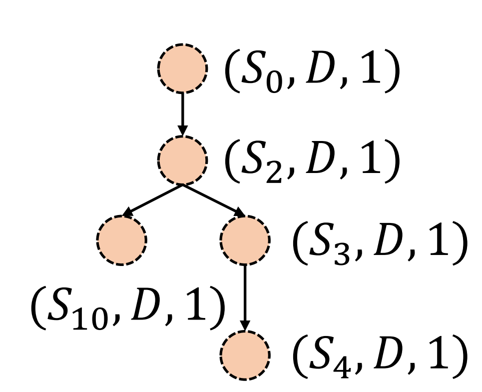
Dominator tree of S



Score of the subgraph dominated by each node



Selected subgraphs of S for Fission Transformation

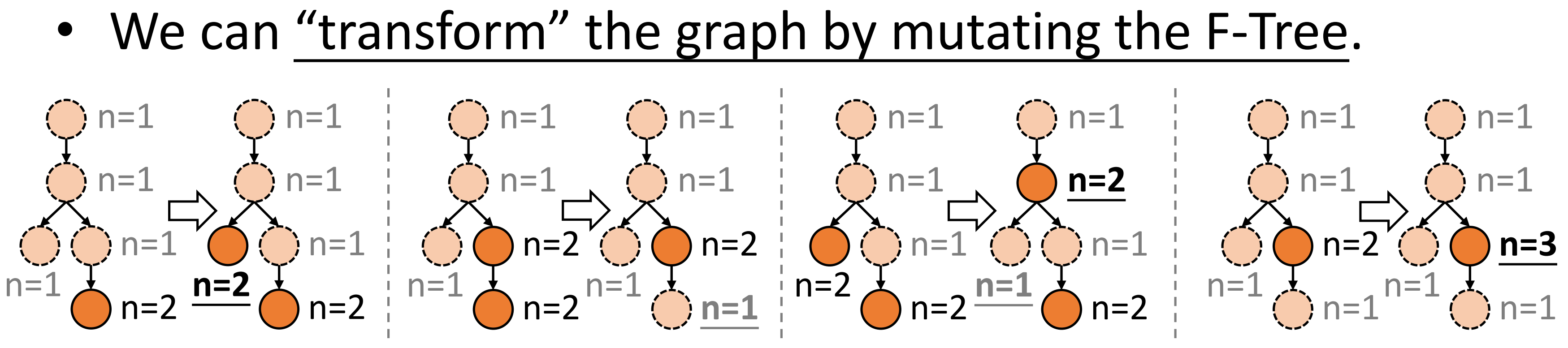


Constructed F-Tree based on selected subgraphs

4. M-Rules & M-Optimizer

Fission Hierarchy Tree Mutation Rules

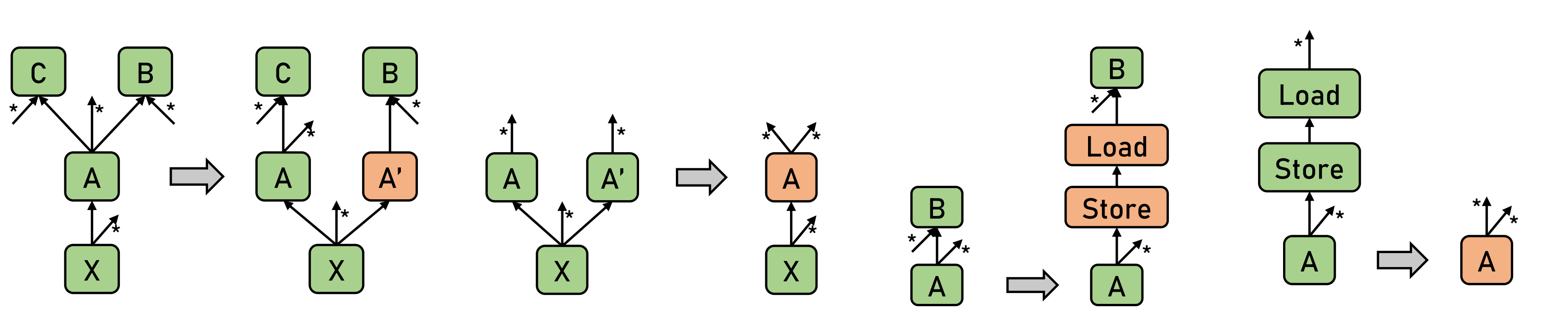
- The Fission Hierarchy Tree serves as a record of the state of a Graph after applying several Fission Transformations.
- We can "transform" the graph by mutating the F-Tree.



Enable leaf node / Disable leaf node / Lift a node / Change fission number

Scheduling-based Rules

- Graph scheduling is a *complex multi-objective optimization for memory & latency* and is *frequently invoked for every newly transformed graphs*.
- We decouple re-materialization and swapping into graph transformation + graph scheduling with only re-ordering, where re-ordering optimizes memory without hurting latency.



Re-mat. Rule / De-Re-mat. Rule / Swapping Rule / De-Swapping Rule

Incremental Scheduling

- We schedule new graph incrementally based on previous schedule result and newly transformed sub-graph region.

5. Evaluation

Intel(R) Xeon(R) Silver 4210R CPUs
NVIDIA GeForce RTX 3090 GPU

Experiment Setup

- Baselines: Torch, POFO, DTR, XLA, TVM, Torch-Inductor
- Networks: ResNet, BERT, ViT, UNet, UNet++, GPT-Neo, BTLM

Main Results

- MAGIS uses 15%~85% peak memory of baselines under the latency overhead constraint 10% and 5%.
- MAGIS brings less than 5% (15%) latency overhead under the memory ratio limit 80% (40%) while baselines cannot.
- MAGIS can achieve better latency & memory pareto frontier than the naïve combination of graph transform & scheduling.