Directed Acyclic Graph Optimization

Chih-Ning Chen, Yosen Lin, Yi-Ching Chen

Carnegie Mellon University, 15642 Machine Learning Systems

Introduction

Deep neural network execution is bottlenecked by the "memory wall," where expensive data transfers between limited fast memory and slow DRAM dominate latency. To mitigate this, we propose a hierarchical DAG scheduling framework that co-optimizes operator fusion and execution granularity using a novel seed-growth strategy and frontier dynamic programming.

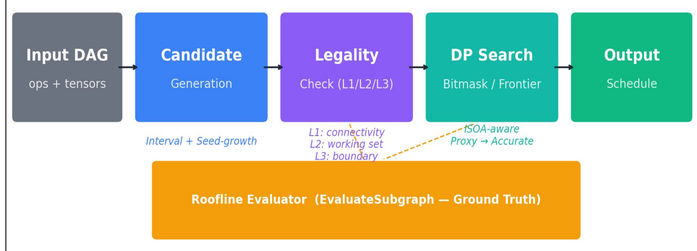
Challenges

RQ1 DAG-Aware Fusion & Validity:
How can we generate and search the combinatorial space of operator fusion candidates?

RQ2 Retain-Aware Scheduling:
What strategies enable optimal tensor retention and tiling decisions within each subgraph, balancing recomputation, spilling, and memory footprint?

RQ3 Conv-Style Guidance & Scalability:
How can we leverage conv-style accelerator models and hybrid search techniques (e.g., graph-cut decomposition, DP, and conv-guided heuristics)?

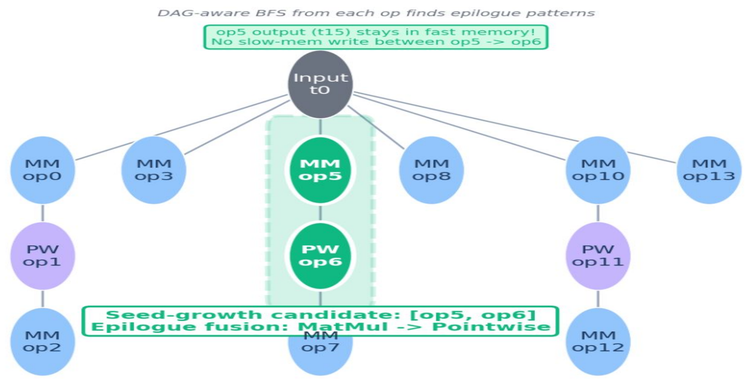
System Overview



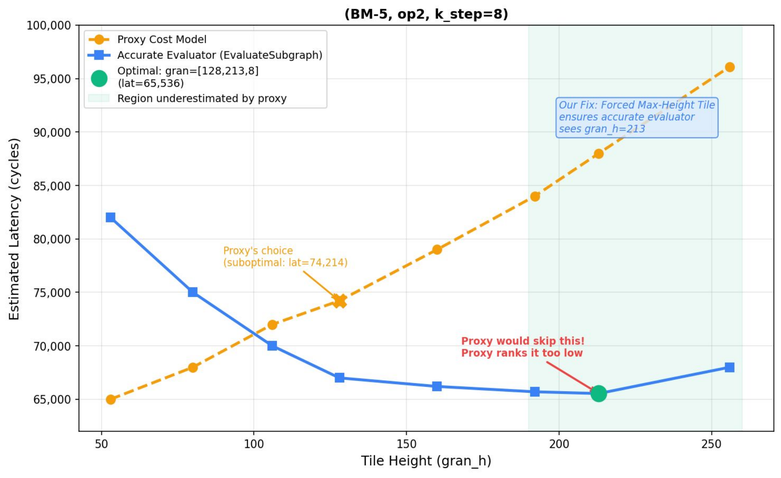
- Seed-growth BFS with beam pruning L1/L2/L3 checks: connectivity, working set ≤ C, boundary validity
- Two-stage: proxy ranking → roofline evaluator, forced max-height
- Bitmask DP (N ≤ 20) or frontier DP + graph-cut

Benchmark	Ops	Tensors	Mem (B)	BW (B/cycle)
mlsys-2026-1	5	9	60,000	20
mlsys-2026-5	19	29	30,000	15
mlsys-2026-9	32	49	250,000	25
mlsys-2026-13	63	100	600,000	50
mlsys-2026-17	103	160	500,000	100

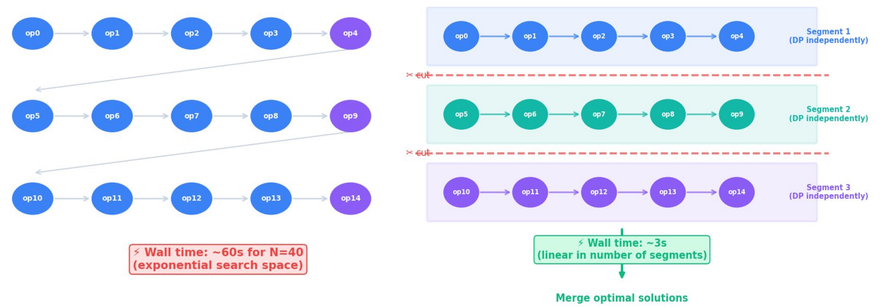
PROPOSED METHODS/DESCRIPTION



Seed-growth strategy takes advantage of DAG input and output treating each operator as a seed and expands outward along producer/consumer edges via breadth-first search. This allows the candidate generator to form groups that are non-contiguous in topological order but are nonetheless connected in the DAG and share internal tensors.



A fast proxy model first ranks candidate tile granularities, but it may overestimate the cost of larger tiles and filter out the true optimum. Therefore, selected candidates are re-evaluated using the accurate evaluator. To reduce proxy bias, We force max-height tile candidates into the accurate evaluation stage, allowing the final schedule to be chosen by the ground-truth model rather than the approximate proxy.



The DAG is decomposed at cut points where no tensor crosses the boundary, so each segment can be solved independently by exact bitmask DP without quality loss. This avoids the failure mode of naive chunking, which arbitrarily slices through live tensors and forces unnecessary slow-memory materializations. A natural extension would combine this with cut vertices to find even more valid decomposition points.

RESULTS

Benchmark	Baseline (cyc)	Scheduler (cyc)	Δ
mlsys-2026-1	471,501	400,000	-15.2%
mlsys-2026-5	1,013,623	650,528	-35.8%
mlsys-2026-9	167,531,000	164,506,000	-1.8%
mlsys-2026-13	166,415,000	166,403,000	∼0%
mlsys-2026-17	46,441,851	46,338,000	-0.2%

Table 4. Benchmark Performance Results

Contributions

(1) **DAG-Aware Operator Fusion:** Seed-growth candidate group generation algorithms, enabling automatic fusion of MatMul and Pointwise operators according to the DAG structure.

(2) **Retain-Aware Partition Search:** Tensor retention decisions into the DP-based partition search.

(3) **Conv-Style Accelerator Guidance:** Pseudo-conv mapping and conv-style accelerator models to analyze working set, memory traffic, and tile propagation.

(4) **Scalable Hybrid Search:** Graph-cut decomposition and frontier DP, splitting large graphs into segments and applying DP search for fusion strategies, balancing solution quality and runtime scalability.