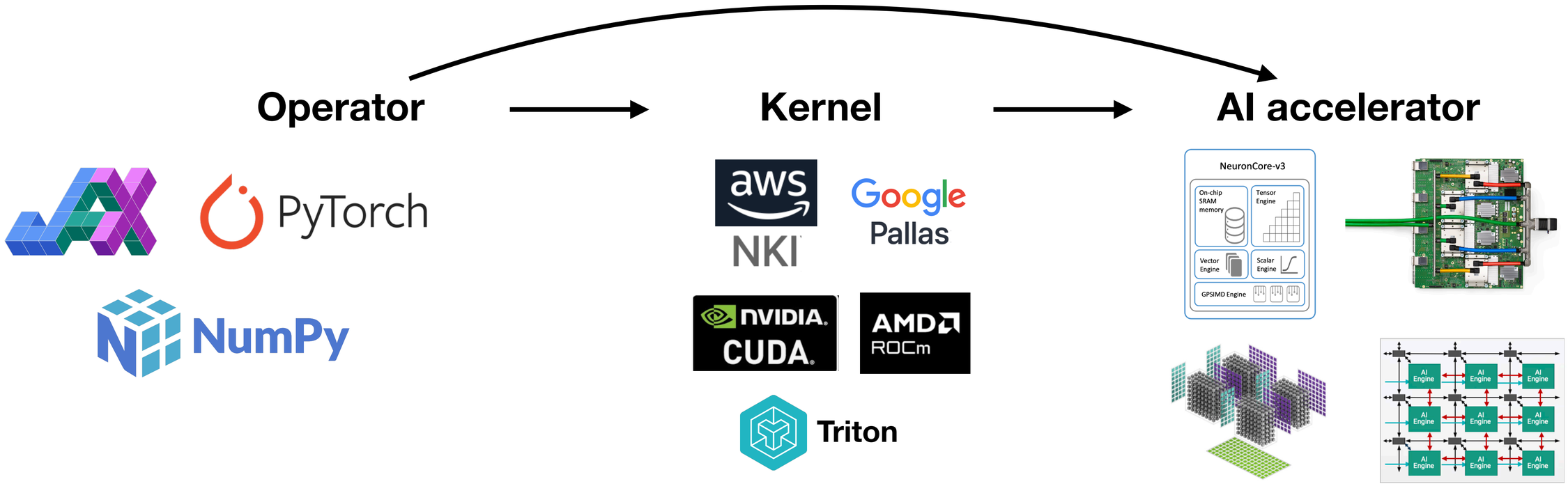
AccelOpt

A Self-improving LLM Agentic System for
AI Accelerator Kernel Optimization

Genghan Zhang
Stanford University

Kernels are Central to ML Systems



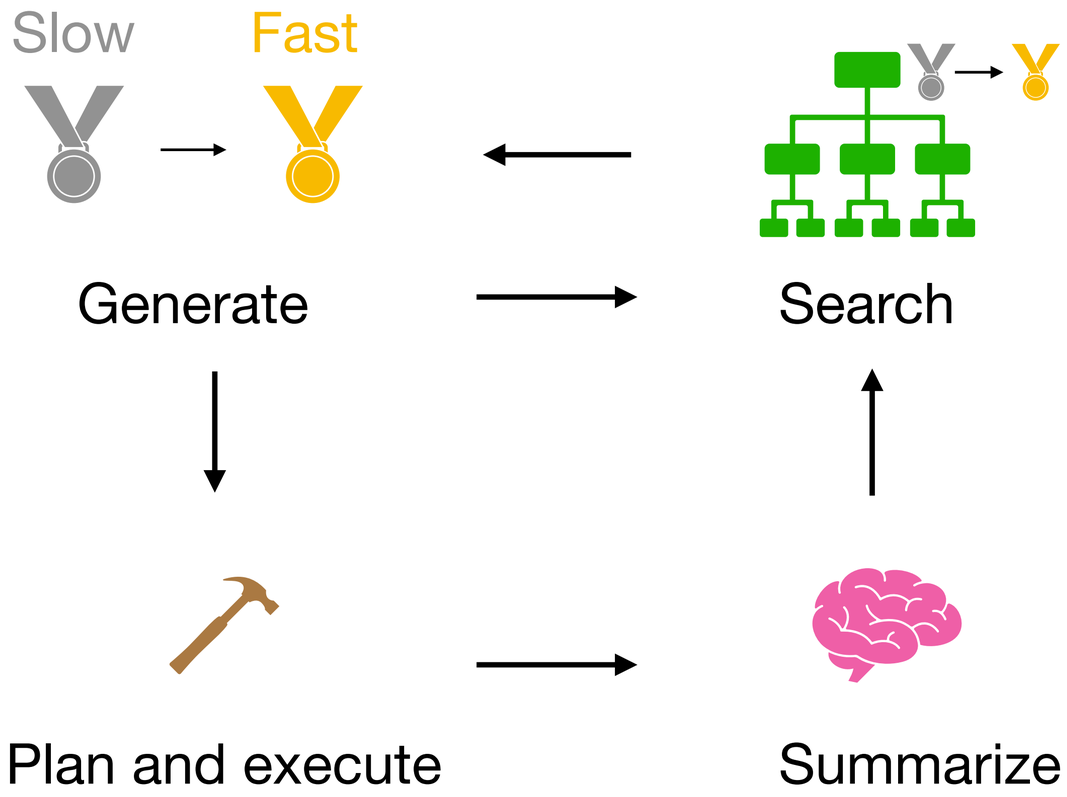
Optimizing AI Accelerator Kernels is Hard

Operator ⟶ **Kernel**

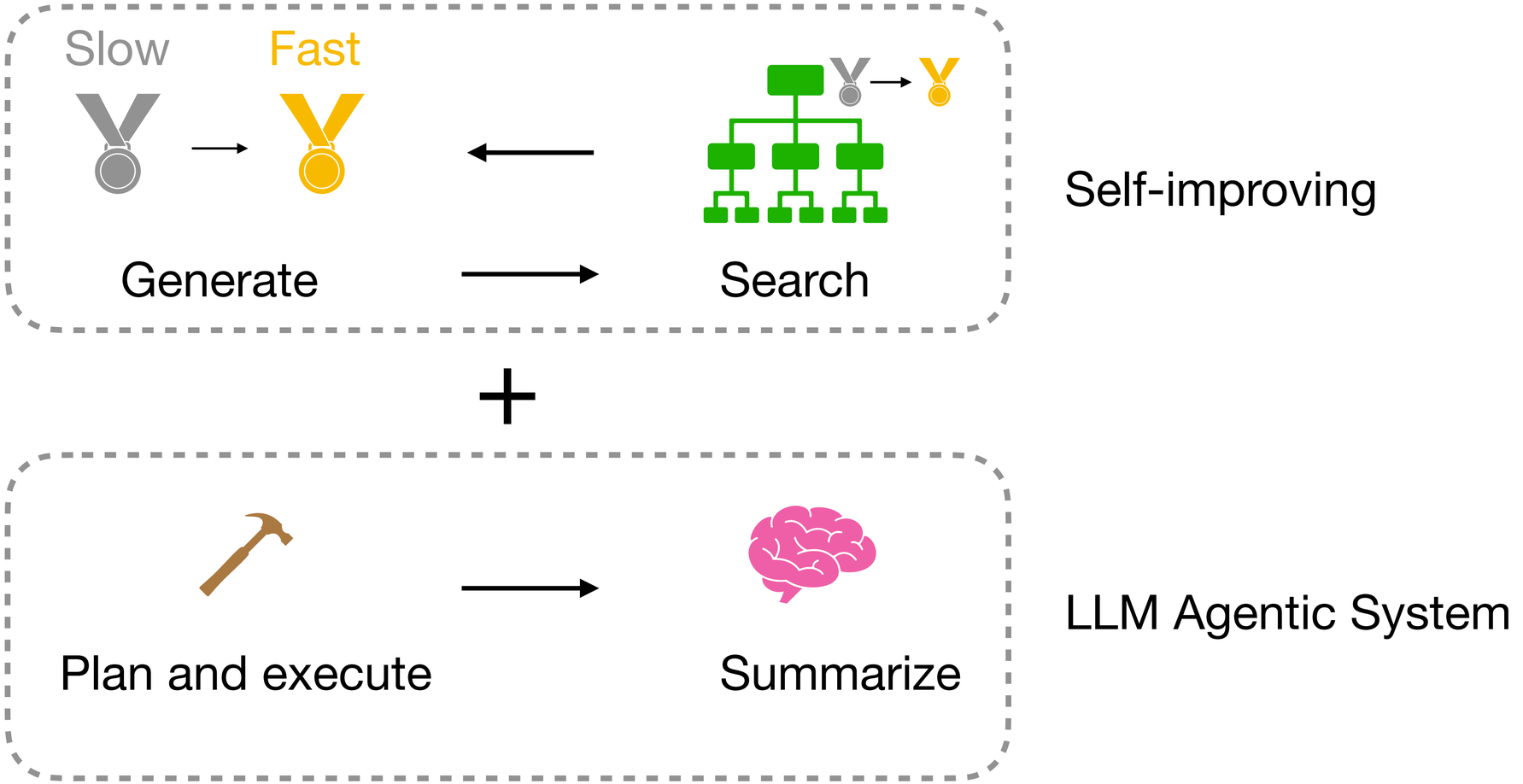
$$C = AB$$

```
TILE_M = nl.tile_size.gemm_stationary_fmax  # 128
TILE_K = nl.tile_size.pmax  # 128
TILE_N = nl.tile_size.gemm_moving_fmax  # 512

# Use affine_range to loop over tiles
for m in nl.affine_range(M // TILE_M):
  for n in nl.affine_range(N // TILE_N):
    # Allocate a tensor in PSUM
    res_psum = nl.zeros((TILE_M, TILE_N), nl.float32, buffer=nl.psum)

    for k in nl.affine_range(K // TILE_K):
      # Declare the tiles on SBUF
      lhsT_tile = nl.ndarray((TILE_K, TILE_M), dtype=lhsT.dtype, buffer=nl.sbuf)
      rhs_tile = nl.ndarray((TILE_K, TILE_N), dtype=rhs.dtype, buffer=nl.sbuf)

      # Load tiles from lhsT and rhs
      lhsT_tile[...] = nl.load(lhsT[k * TILE_K:(k + 1) * TILE_K,
                                    m * TILE_M:(m + 1) * TILE_M])
      rhs_tile[...] = nl.load(rhs[k * TILE_K:(k + 1) * TILE_K,
                                  n * TILE_N:(n + 1) * TILE_N])

      # Accumulate partial-sums into PSUM
      res_psum += nl.matmul(lhsT_tile[...], rhs_tile[...], transpose_x=True)
```

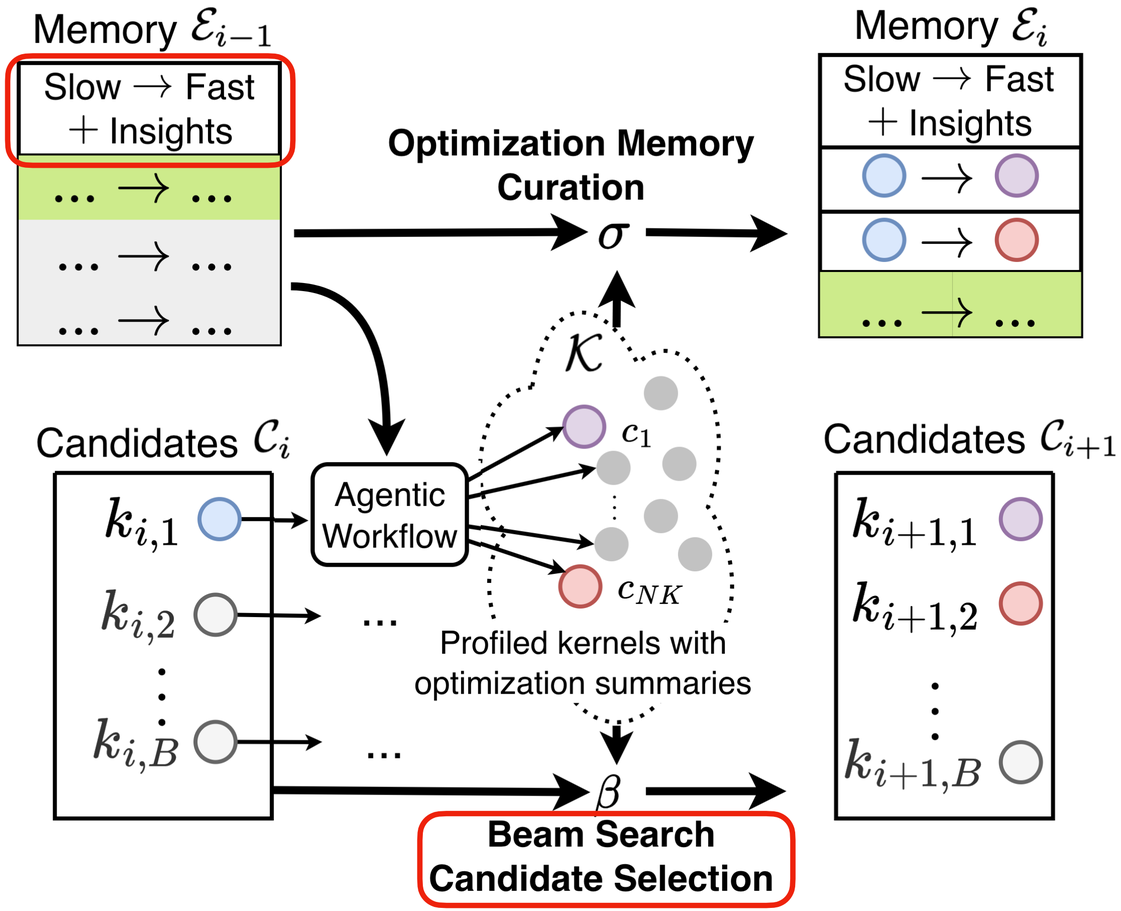
Automate Kernel Optimization with LLM



Automate Kernel Optimization with LLM

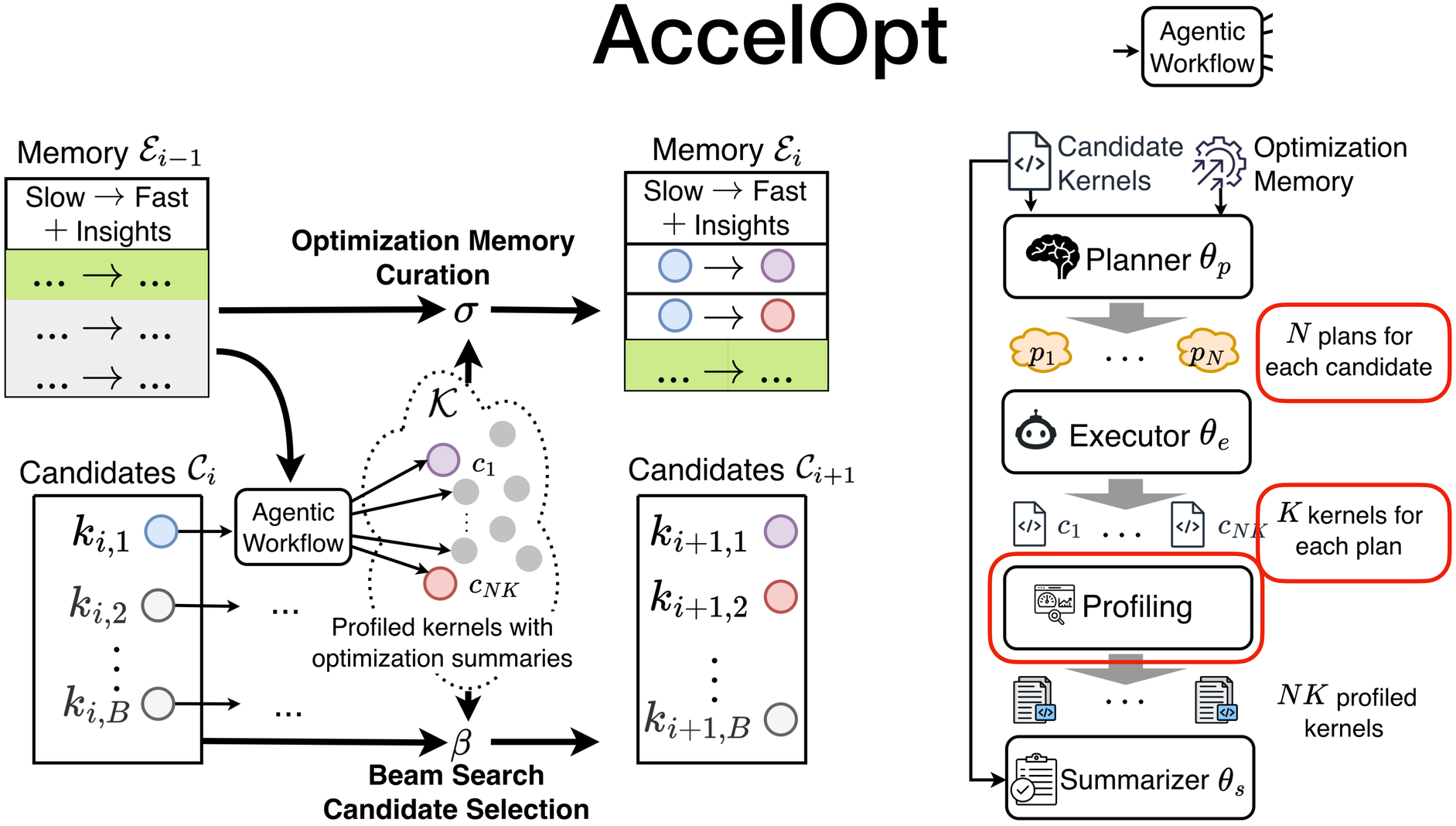


AccelOpt



Self-improving + LLM Agentic System

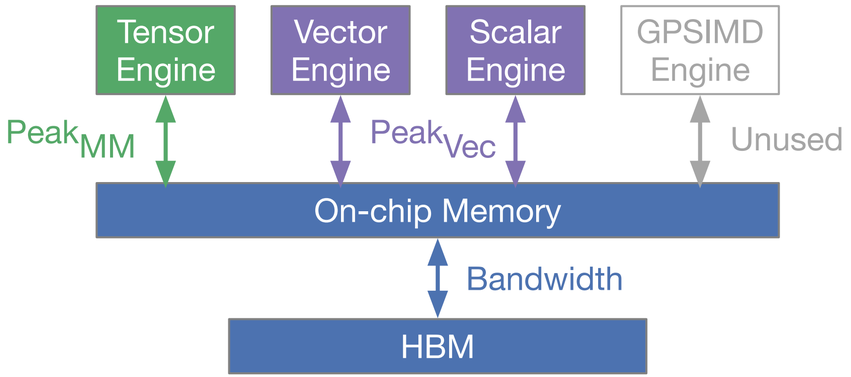
AccelOpt



Self-improving + LLM Agentic System

Metric: Percentage of Peak

$$D = Norm(A @ B) + C$$

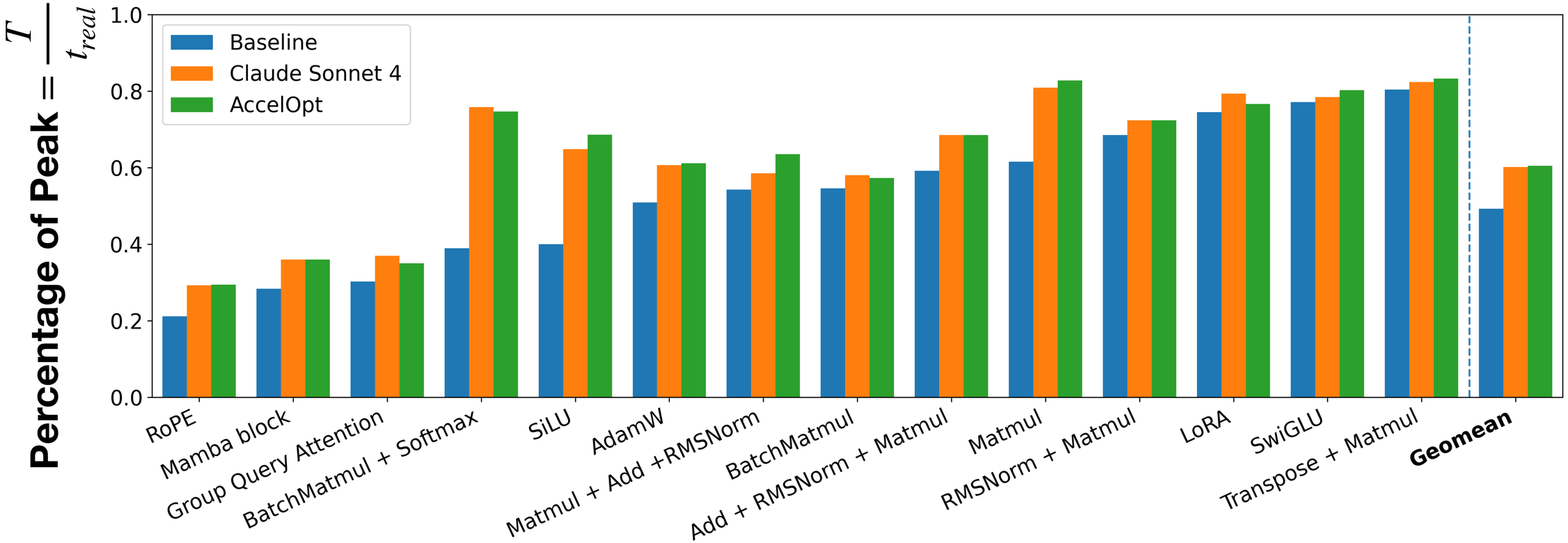


$$T = \max\left(\frac{\text{Traffic}_{\text{Min}}}{\text{Bandwidth}}, \frac{\text{FLOPs}_{\text{MM}}}{\text{Peak}_{\text{MM}}}, \frac{\text{FLOPs}_{\text{Vec}}}{\text{Peak}_{\text{Vec}}}\right)$$

$$\textbf{Percentage of Peak} = \frac{T}{t_{real}}$$

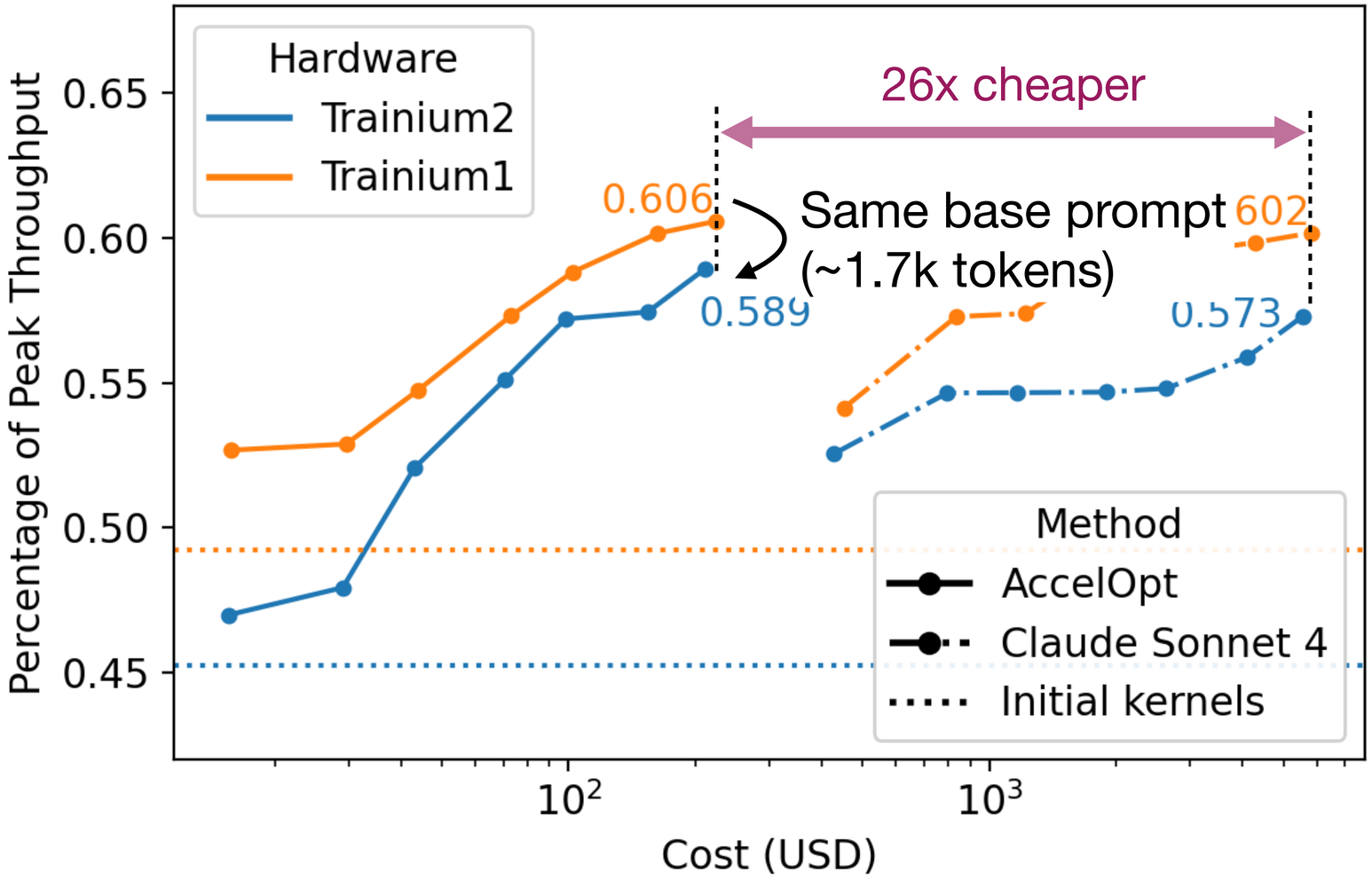
Results on NKIBench

NKIBench: agent environment with initial NKI kernels from real LLM workload

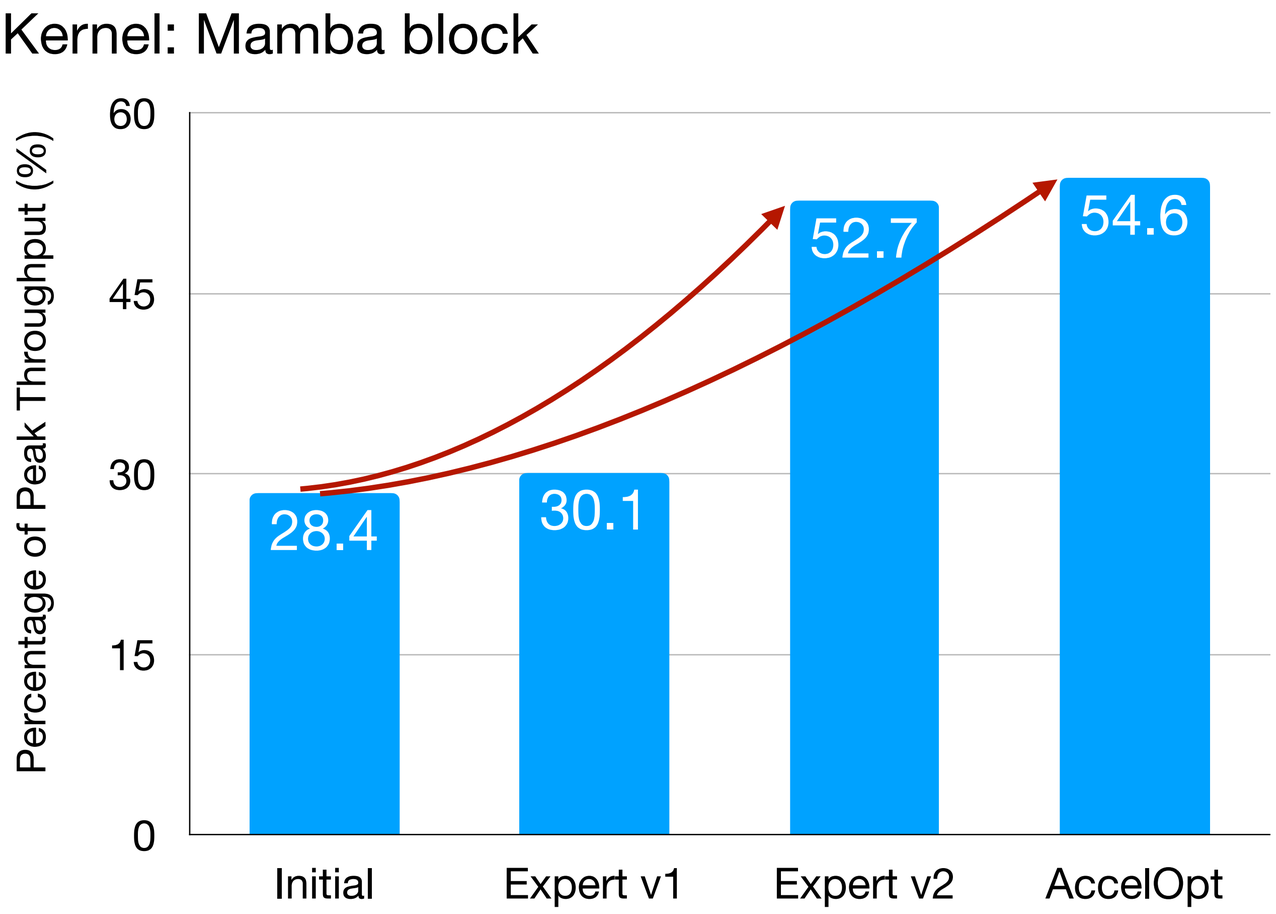


AccelOpt (gpt-oss-120b + Qwen3-Coder-480B) is on par with Claude Sonnet 4

AccelOpt is Cost Efficient and Adaptive

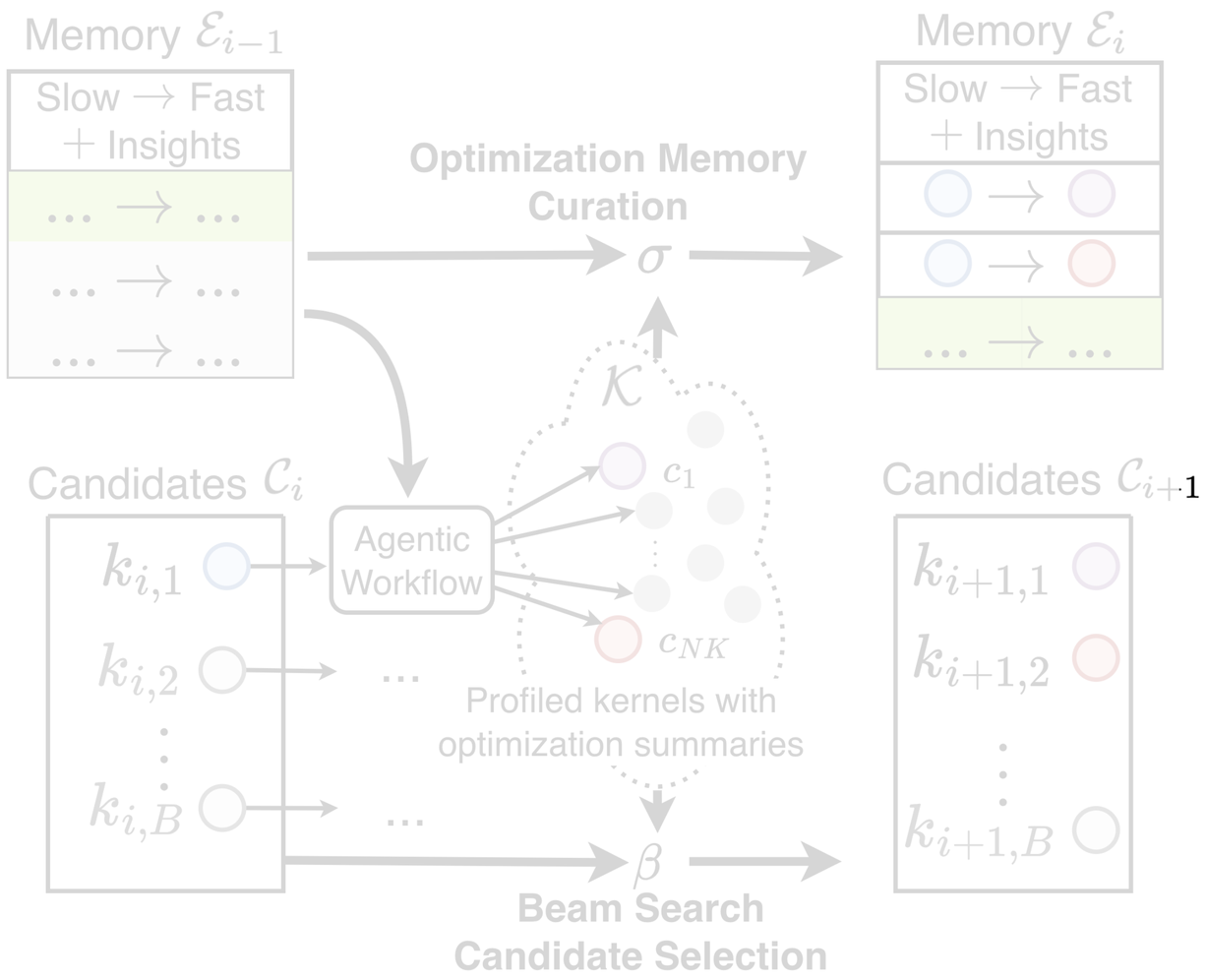


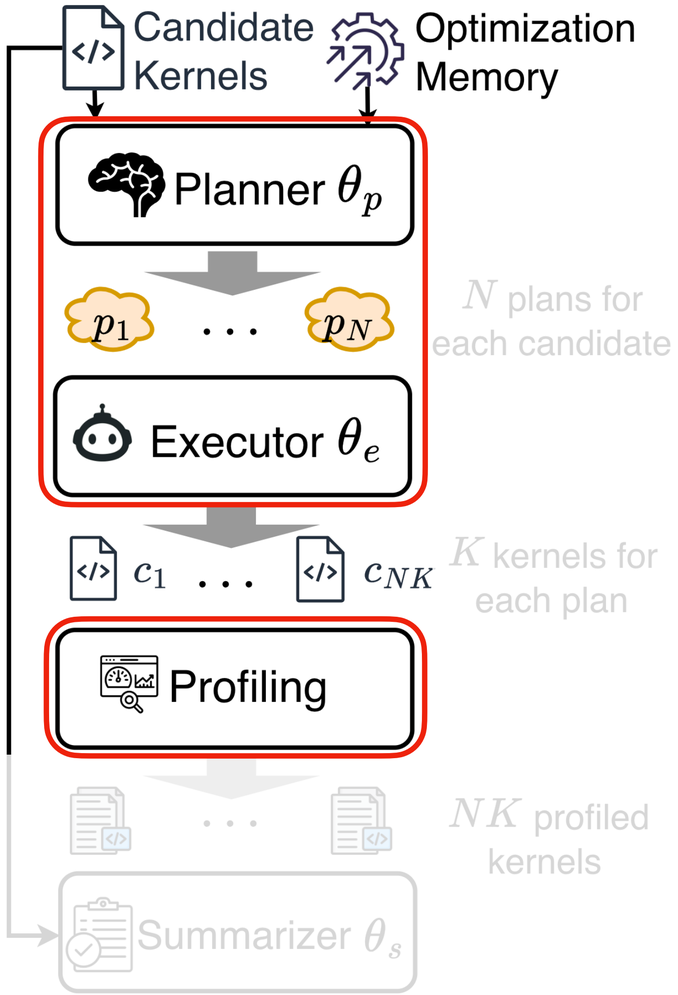
Compare AccelOpt with Experts



AccelOpt is a step towards human-expert level of kernel optimization on emerging AI accelerators

Apply AccelOpt to Other AI Accelerators





Apply AccelOpt to NVIDIA H100

Kernel: Paged Grouped Query Attention

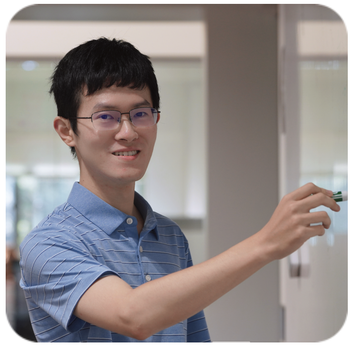
Workload	FlashInfer v0.5.3	Best Triton baseline	AccelOpt + Gemini 3 Flash
Qwen3-30B-A3B Decode	1.0x	0.26x (GPT-5)	**1.50x** / **5.77x**
Llama-3.1-8B Decode	1.0x	0.43x (Claude Opus 4.1)	**1.08x** / **2.51x**
Qwen3-30B-A3B Prefill	1.0x	0.12x (Claude Opus 4.1)	**1.38x** / **11.5x**
Llama-3.1-8B Prefill	1.0x	0.04x (Gemini 2.5 Pro)	0.10x / **2.5x**

AccelOpt can optimize complex Triton kernels (sometimes better than kernel libraries)

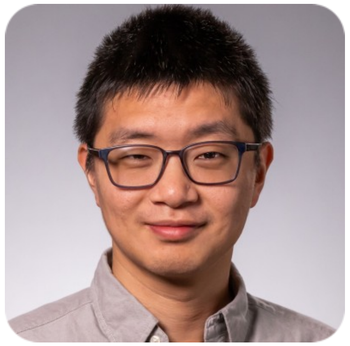
Large workloads: best Triton baseline > 0.1ms on FlashInfer-Bench

Collaborators

Shaowei Zhu



Anjiang Wei



Zhenyu Song

Allen Nie

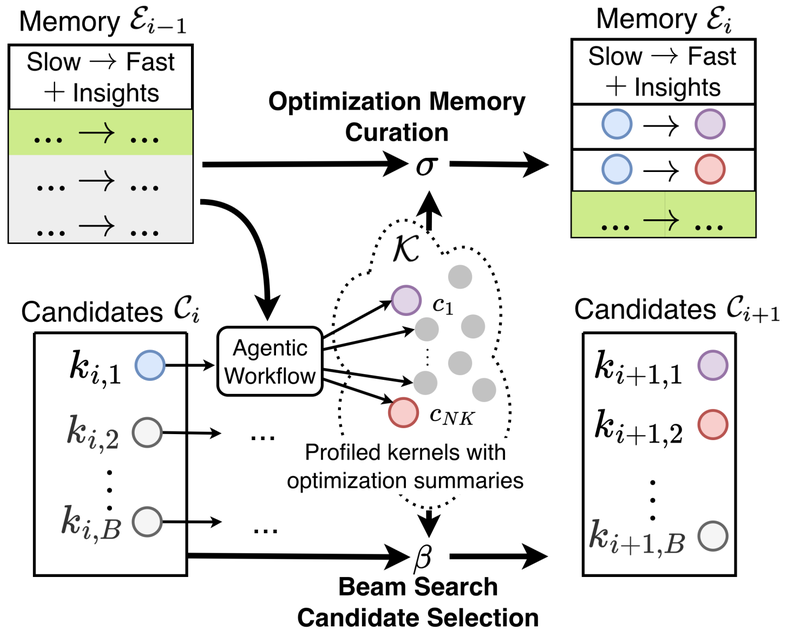
Zhen Jia

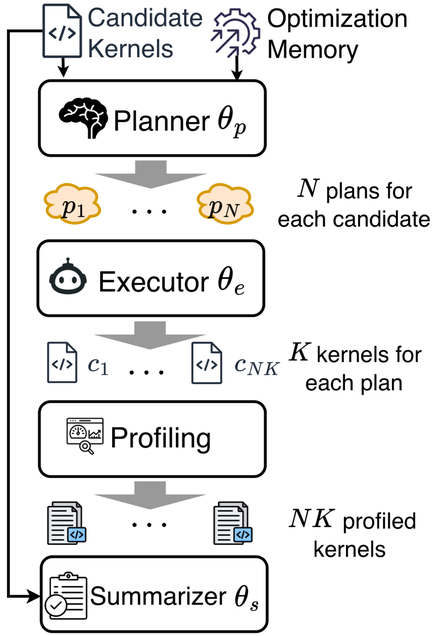
Nandita Vijaykumar

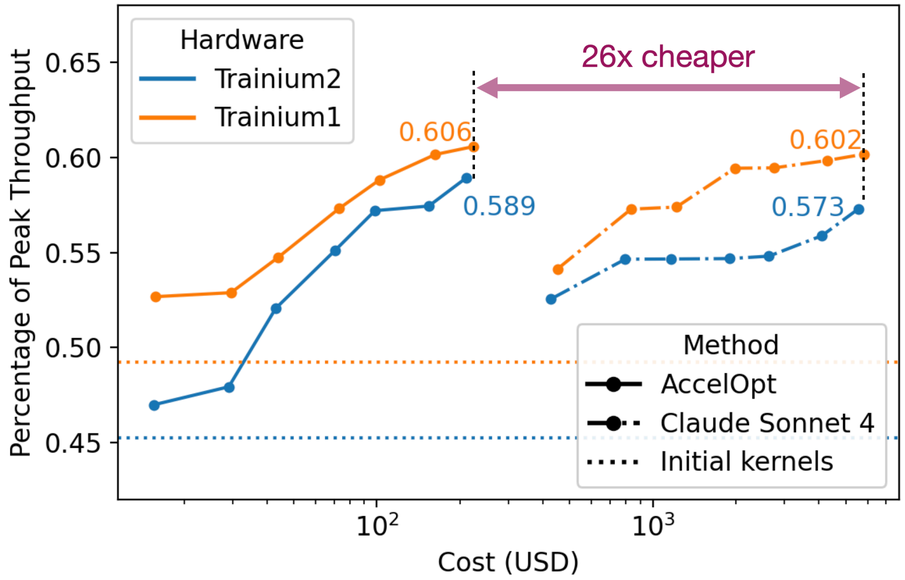
Yida Wang

Kunle Olukotun

AccelOpt: A Self-improving LLM Agentic System for AI Accelerator Kernel Optimization







AccelOpt - Stanford Pervasive Parallelism Lab
ppl.stanford.edu

Genghan Zhang
zhang677.github.io

Thank you!