MLC with TVM

Digital-Nomad-Cheng
Slides credit to Tianqi Chen, OctoML

Background

Growing set of requirements: **Cost, latency, power, security & privacy**

Cambrian explosion of models, workloads, and use cases

CNN **GAN** **RNN** **MLP** **DQNN**

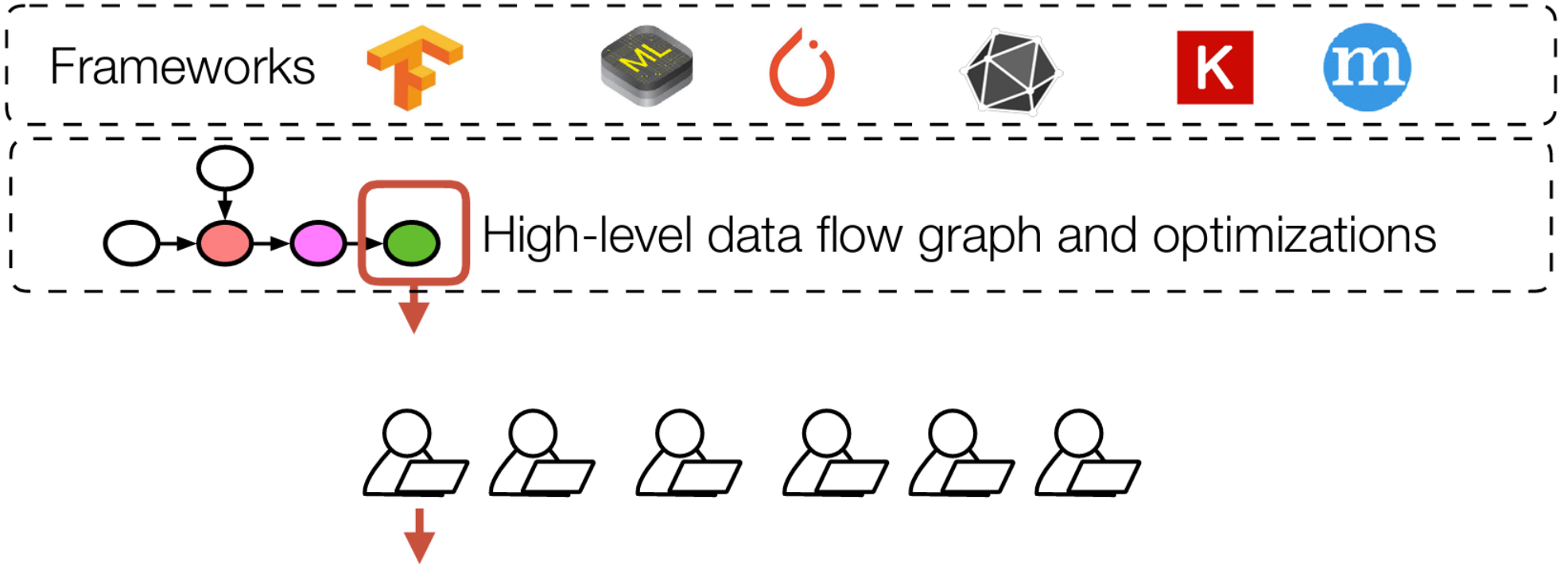
Rapidly evolving ML software ecosystem

Silicon scaling limitations (Dennard and Moore)

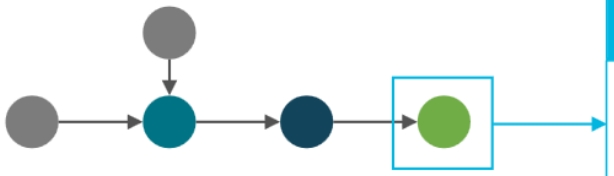
Cambrian explosion of HW backends. Heterogeneous HW

intel NVIDIA XILINX Microsoft QUALCOMM amazon Google HUAWEI

Challenges


Frameworks
ML
K
m
High-level data flow graph and optimizations
Hardware

Computation vs Implementation



Tensor Expression (Specification)

```
C = tvm.compute((m, n),
lambda y, x: tvm.sum(A[k, y] * B[k, x], axis=k))
```

Search Space of Possible Program Optimizations

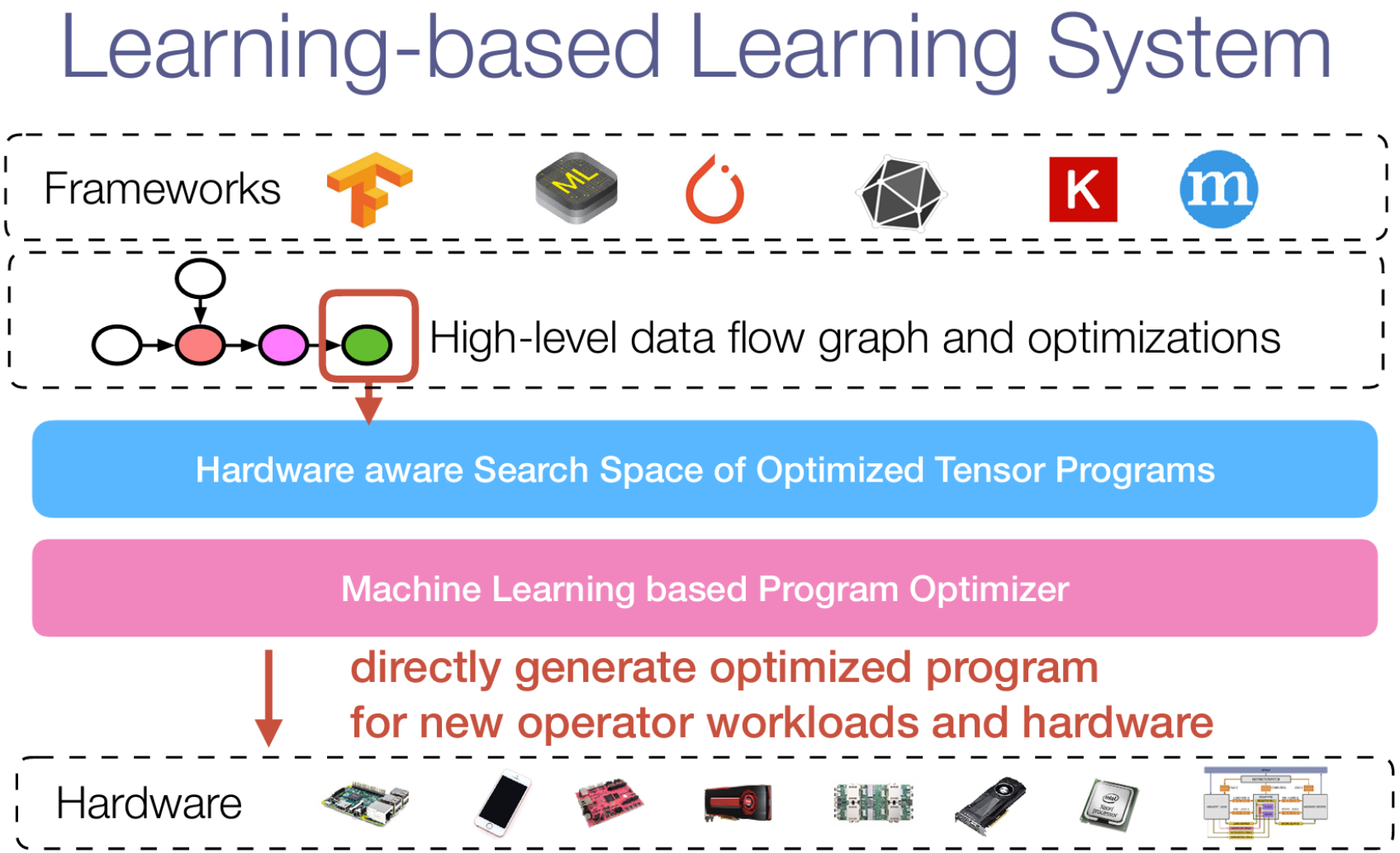
Low-level Program Variants

```
inp_buffer AL[8][8], BL[8][8]
acc_buffer CL[8][8]
for yo in range(128):
  for xo in range(128):
    vdla.fill_zero(CL)
    for ko in range(128):
      vdla.dma_copy2d(AL, A[ko*8:ko*8+8][yo*8:yo*8+8])
      vdla.dma_copy2d(BL, B[ko*8:ko*8+8][xo*8:xo*8+8])
      vdla.fused_gemm8x8_add(CL, AL, BL)
    vdla.dma_copy2d(C[yo*8:yo*8+8,xo*8:xo*8+8], CL)
```

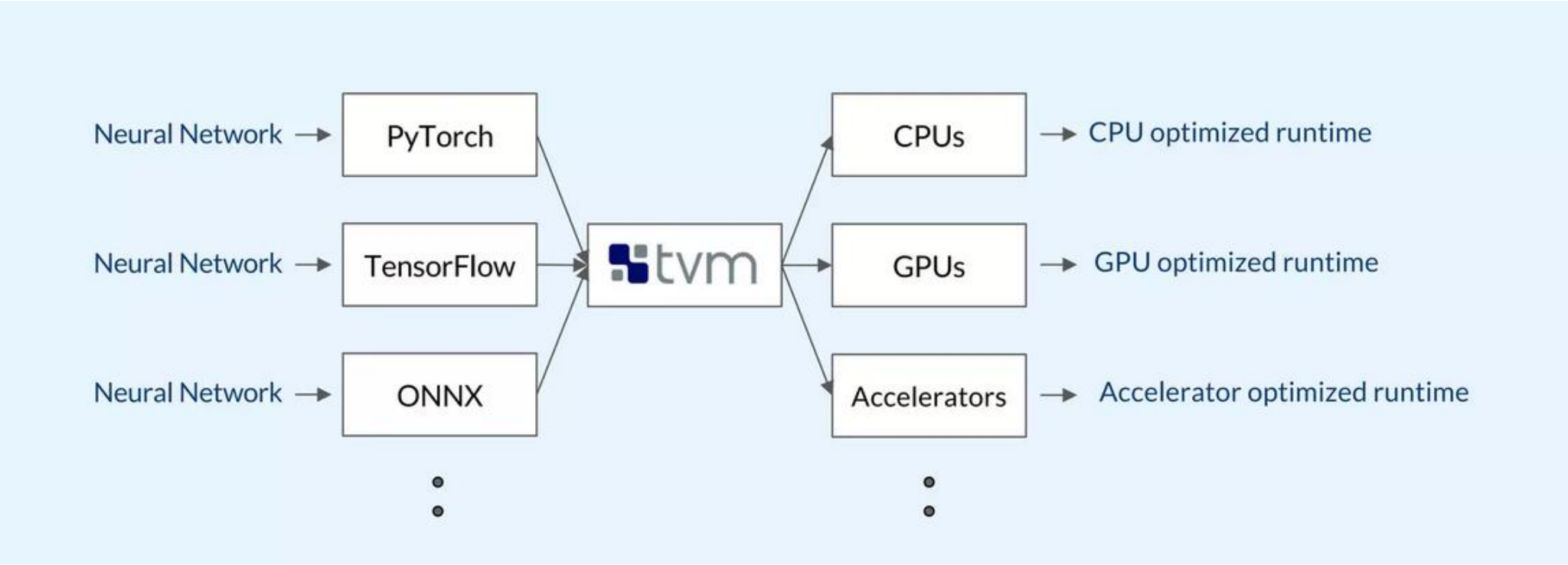
```
for yo in range(128):
  for xo in range(128):
    C[yo*8:yo*8+8][xo*8:xo*8+8] = 0
    for ko in range(128):
      for yi in range(8):
        for xi in range(8):
          for ki in range(8):
            C[yo*8+yi][xo*8+xi] +=
              A[ko*8+ki][yo*8+yi] * B[ko*8+ki][xo*8+xi]
```

```
for y in range(1024):
  for x in range(1024):
    C[y][x] = 0
    for k in range(1024):
      C[y][x] += A[k][y] * B[k][x]
```

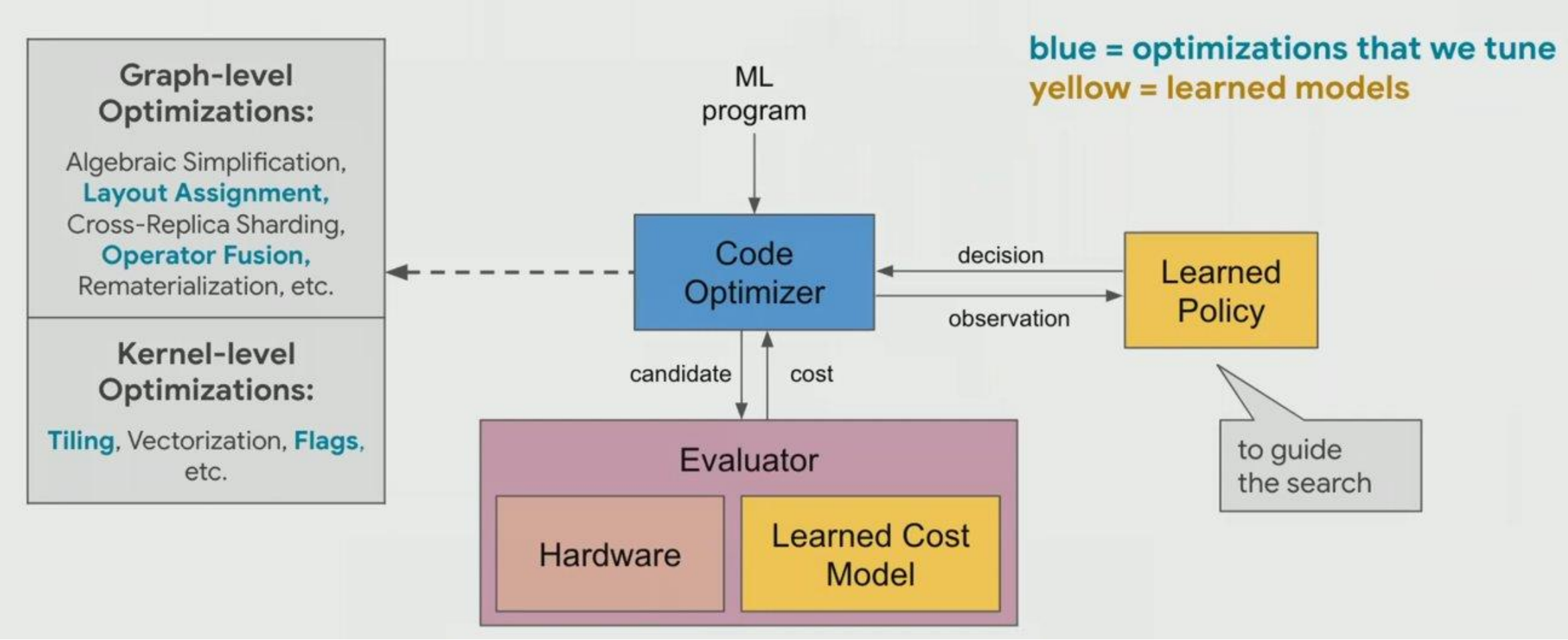
Here comes TVM



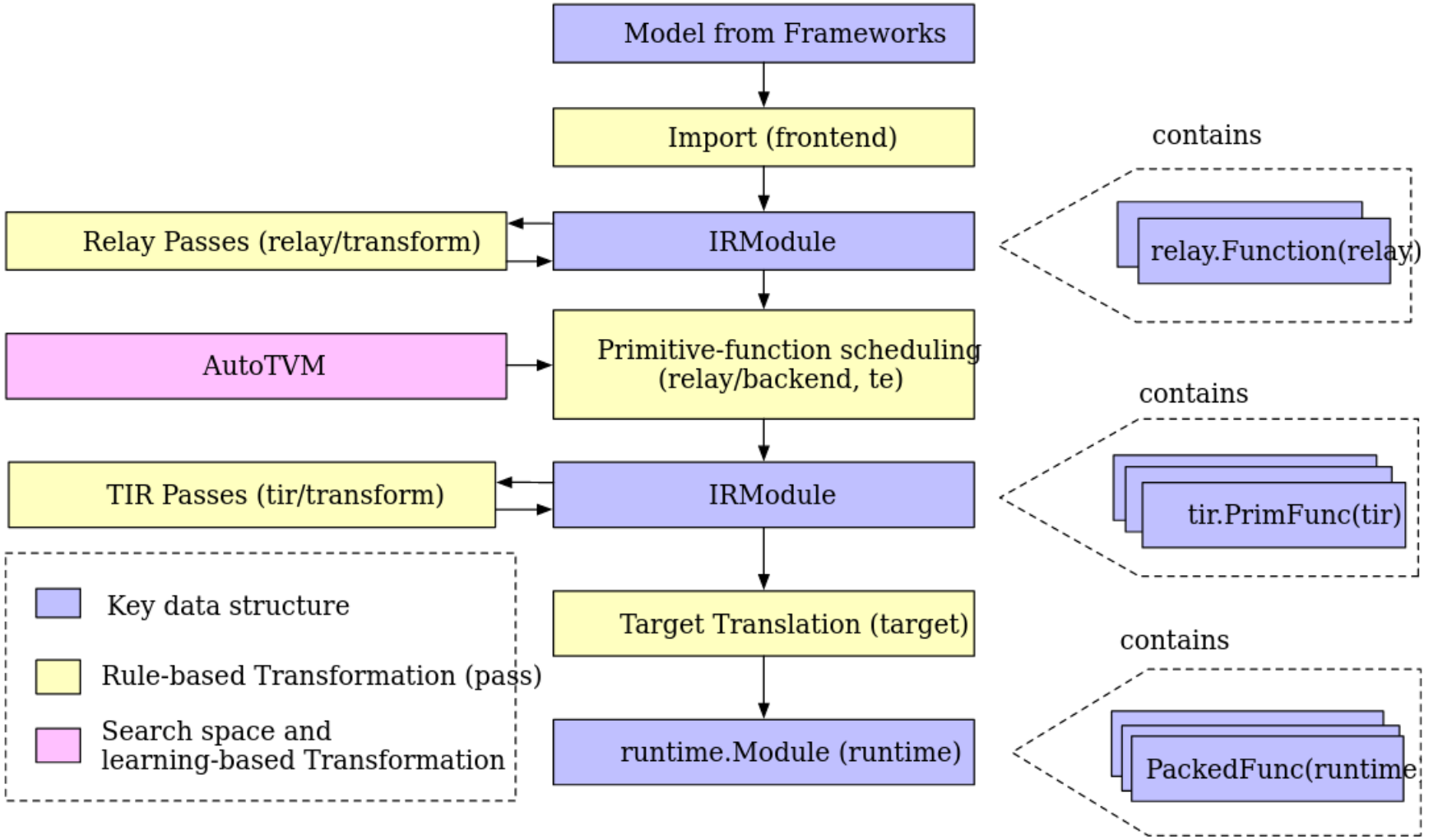
Introduction to TVM



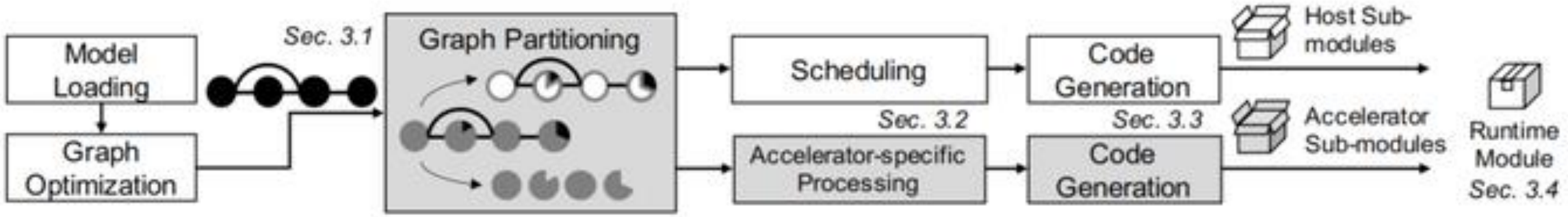
Auto Tuning


blue = optimizations that we tune
yellow = learned models
Graph-level Optimizations:
Algebraic Simplification,
Layout Assignment,
Cross-Replica Sharding,
Operator Fusion,
Rematerialization, etc.
Kernel-level Optimizations:
Tiling, Vectorization, Flags,
etc.
ML program
Code Optimizer
decision
observation
Learned Policy
candidate
cost
Evaluator
Hardware
Learned Cost Model
to guide the search

How it works in TVM



BYOC: support vendor library



- TensorRT is supported with 4403 lines of code
- Xilinx Vitis AI is supported with 1924 lines of code
- Other supported hardware/codegen: TensorFlow Lite, Arm compute library, etc.

Some projects I have done/doing with TVM

- Generate efficient matmul cuda source kernel
- Export a face detection model from pytorch to tvm, perform auto-scheduler, call tvm runtime from C++ and parse result, and benchmarks
- Benchmark mnv2/yolov8's power/memory/time performance in tvm using different codegen mechanisms: default, tensorrt, auto-scheduler
- Use BYOC to support a new codegen

matmul cuda kernel

- Define computation using TensorIR
- Schedule
  - Schedule computation manually: shared memory tiling, thread tiling, etc
  - Using TVM's auto-scheduler's to search the best schedule strategy
- Results
  - Auto scheduler can generate cuda kernel on pair with human expert in a short time

Face Detection with TVM runtime

- Parse computation from pytorch/onnx to relay IR
- Codegen
  - Use auto schedule to search best strategy for implementation on target device
- Export runtime library
- Import runtime library from C++, call runtime, parse input/output

Different codegen benchmark

- Default: runnable, not efficient
- TensorRT: fast, more memory footprint
- AutoScheduler: fast, less memory footprint, long tuning time

Backends	Power Consumption	Memory Footprint	Inference Time
Static	30W	6MB	
Peak	80W	8GB	
Default	67W	170MB	1.6139ms
TensorRT	79W	246MB	0.5078ms
Auto Scheduler	79W	170MB	0.4949ms

Source code

- matmul: https://github.com/digital-nomad-cheng/matmul_cuda_kernel_tvm/
- Face detection demo: https://github.com/digital-nomad-cheng/RetinaFace_TVM
- Benchmark: https://github.com/digital-nomad-cheng/tvm_project_course

Have questions?

- The Deep Learning Compiler A Comprehensive Survey
- Bring Your Own Codegen to Deep Learning Compiler
- TVM discuss: https://discuss.tvm.apache.org/
- TVM issues: https://github.com/apache/tvm/issues?q=is%3Aissue+is%3Aopen+BYOC
- Docs: https://tvm.apache.org/docs/
- Source code: tutorials, demos, tests