Ph.D. Forum @ DAC

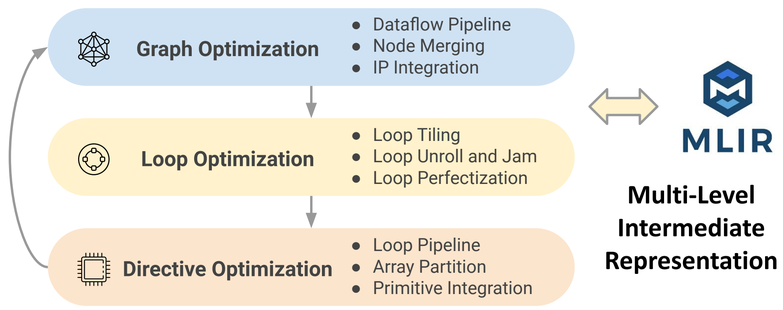
ScaleHLS: A Scalable High-Level Synthesis Framework

Hanchen Ye, Deming Chen *UIUC (University of Illinois at Urbana-Champaign)*

Challenges and Motivation

HLS (High-Level Synthesis) has a great potential to continue to drive the high-productivity designs of circuits with high-density, high-energy efficiency, and short design cycle. However, there still exhibits significant challenges on handling large-scale HLS design:

- **Representation:** Software-oriented intermediate representation (IR) is difficult to carry effective HLS optimizations
- **Optimization:** HLS optimizations highly relies on manual code rewriting, including directive, loop, and graph optimizations
- **Exploration:** Vast and complicated design space to explore



MLIR

Multi-Level Intermediate Representation

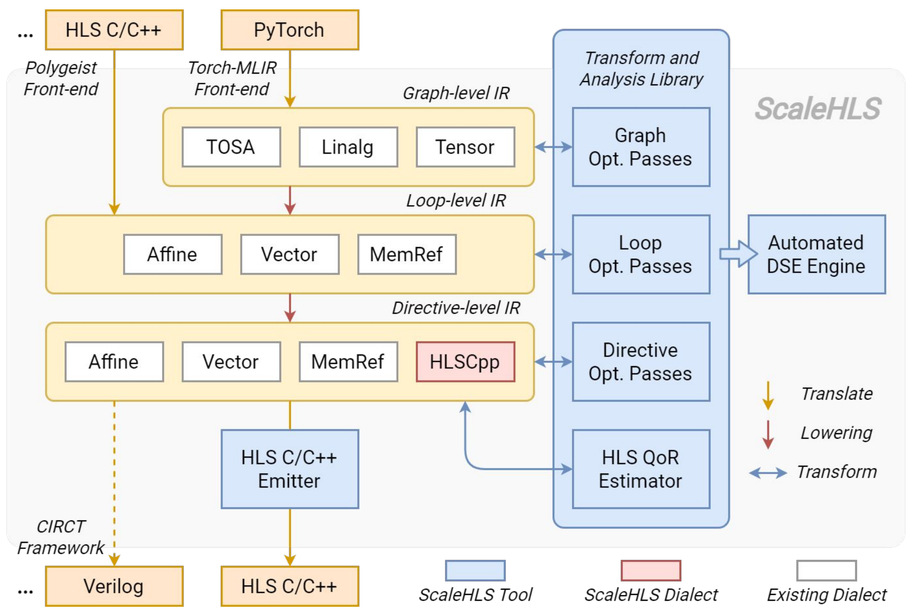
Marry HLS and MLIR

- Abstract HLS designs into multiple representation levels
- Solve the HLS optimization problems at "correct" abstraction levels
- Enable comprehensive design space exploration for optimal solutions

Large MLIR Community:

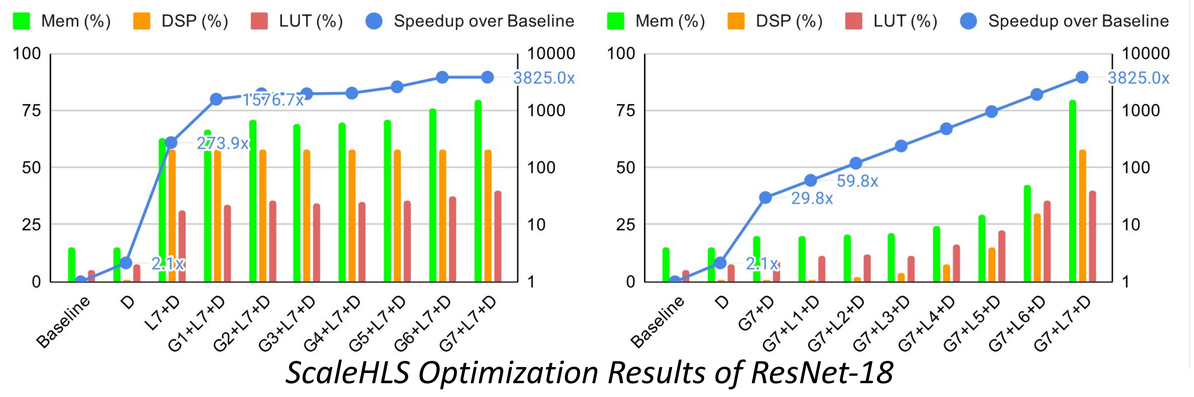
intel. NVIDIA. AMD SiFive Microsoft Google … …

ScaleHLS - HPCA'22, LATTE'21



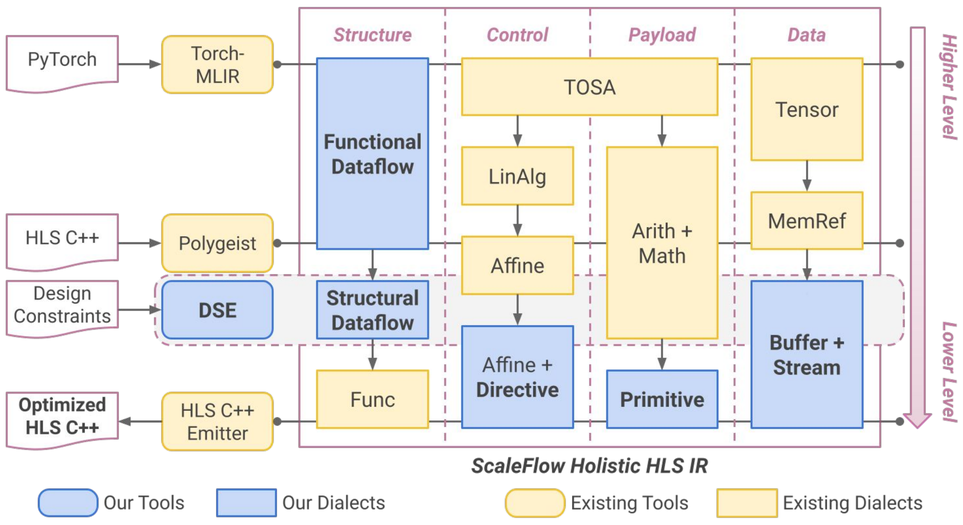
First Multi-Level HLS Representation Optimization Exploration

- **Fast** Compilation
- **Thorough** DSE
- **Efficient** U-Arch

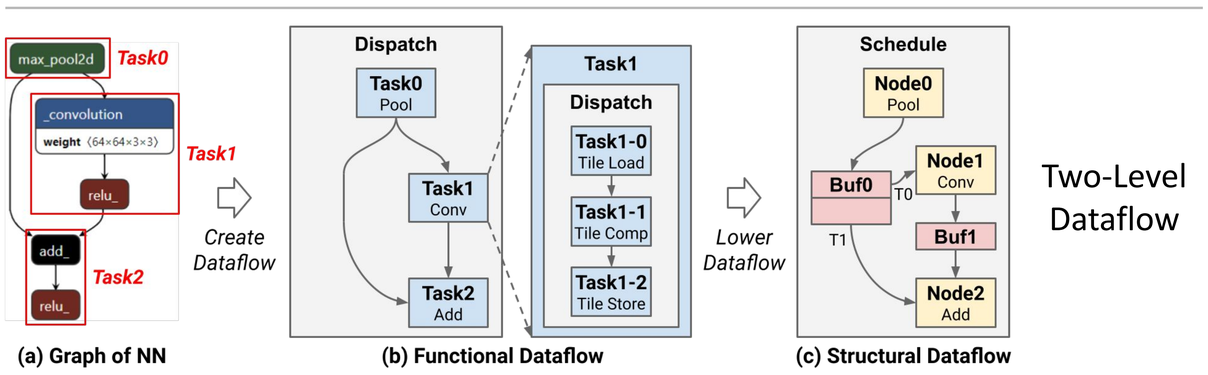


ScaleHLS Optimization Results of ResNet-18

ScaleFlow - TECHCON'23



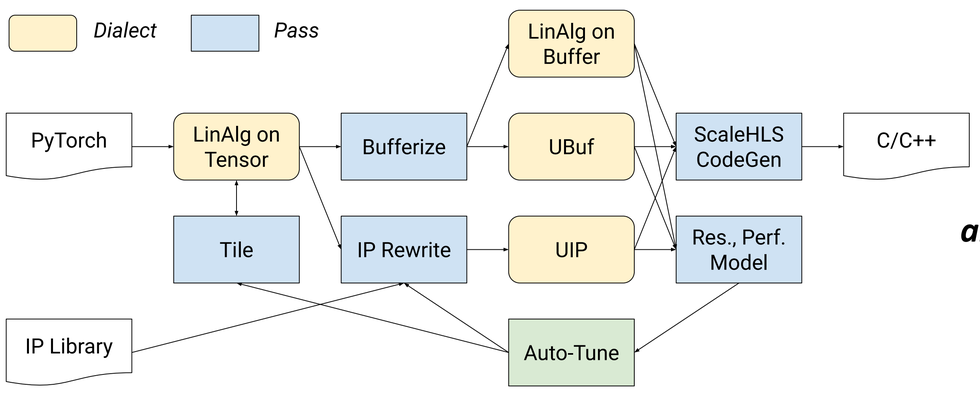
Automated Coarse-Grained Dataflow Optimization



Two-Level Dataflow

- **Functional Dataflow**: High-level IR without hardware details for fast dataflow construction and task partition
- **Structural Dataflow:** Low-level IR with hardware details, such as the communication between dataflow nodes, for comprehensive dataflow scheduling, optimization, and design space exploration

UniLibrary (WIP)

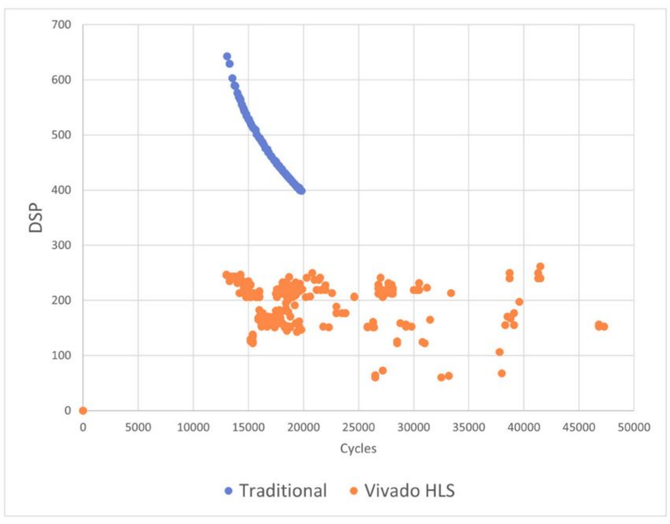


Automated IP Matching and Integration

```python
@uflow.register_ip("gemv")                                  # Register a IP "gemv"
class Gemv(uflow.Ip):
    def __init__(self):
        super().__init__()
        self.data_type = uflow.Float()                      # IP data types
        self.index_type = uflow.UInt(32)

        self.par_m = uflow.CompileParam(self.index_type, [2, 4, 8])   # IP compile-time params
        self.dim_m = uflow.DynamicParam(self.index_type, (16, 1024))
        self.dim_n = uflow.DynamicParam(self.index_type, (16, 1024))  # IP runtime params

        self.mat_a = uflow.InputPort(uflow.DynamicTensor(   # IP input port "mat_a"
            self.data_type, [self.dim_m, self.dim_n],
            lambda m, n: [m / self.par_m, n, m % self.par_m]))   # Shape and layout of "mat_a"
        self.vec_b = uflow.InputPort(
            uflow.DynamicTensor(self.data_type, [self.dim_n], lambda n: [n]), size=[1])
        self.vec_c = uflow.OutputPort(                      # IP output port "vec_c"
            uflow.DynamicTensor(self.data_type, [self.dim_m],
            lambda m: [m / self.par_m, m % self.par_m]), size=[1])   # Size 1 indicates "vec_c" is FIFO

    def semantics(self):
        for m in self.dim_m:                                # IP semantics for pattern match
            for n in self.dim_n:
                self.vec_c[m] += self.mat_a[m, n] * self.vec_b[n]
```

An Example of GEMV IP Registration

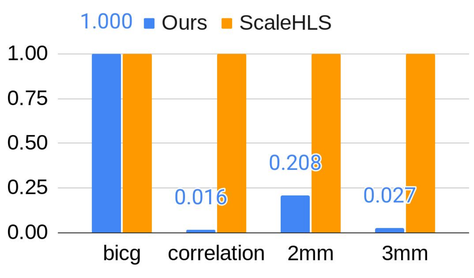
AutoScaleDSE - DAC'22, TRETS



Traditional quality-of-result estimation exhibits large inaccuracies

ML-augmented HLS Design Space Exploration

1. Random forest-based quality-of-result estimation
2. Evolutionary algorithm-based buffer partition



Open-Source Contribution

ScaleHLS GitHub Repository

https://github.com/hanchenye/scalehls

35k Views and **2.9k** Downloads since 2022