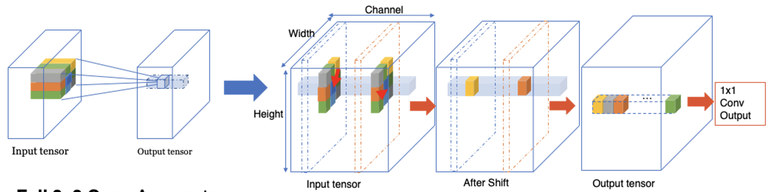
Co-optimization of Algorithms, Hardware, and Scheduling for Deep Learning Applications

Qijing Jenny Huang
Email: qijing.huang@berkeley.edu
Advisor: John Wawrzynek
URL: hqjenny.com

1. Hardware-Friendly Algorithm Design

Synetgy: Image Classification without 3x3 Convolution [FPGA'19]



Full 3x3 Conv Aggregates neighboring pixels and mixes channel info

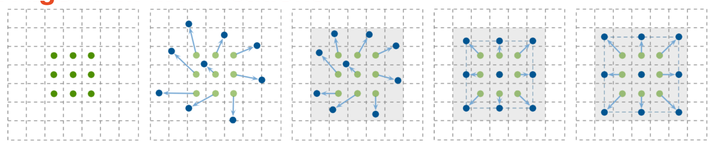
Shift moves a neighboring pixel to the center position

1x1 Conv aggregates spatial info along the channel dimension

- 3x3 Conv → Shift and 1x1 Conv
- Dataflow accelerator on embedded FPGA
- Achieves **equal top-1 accuracy**, **11.6x higher framerate**, **6.3x better power efficiency**

CoDeNet: Object Detection with Deformable Convolution Codesign [EMC2'19]

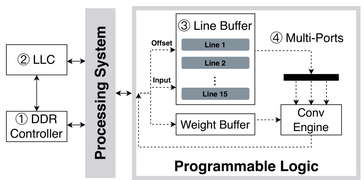
I. Algorithm Modifications



(a) Regular (b) Deformable (c) Bound d) Square. (e) Round

- **Deformable Conv** samples inputs from variable offsets generated based on the input
- **9.76×** speedup for the deformable conv on FPGA
- **20.9×** smaller model but **10%** higher accuracy than Tiny-YOLO

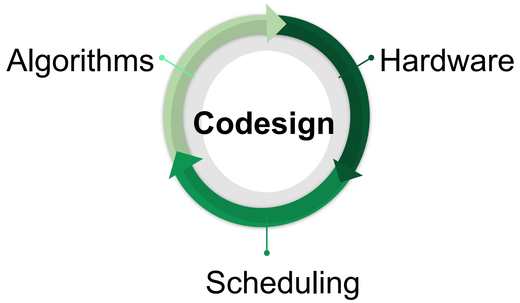
II. Hardware Optimizations



(2) Caching (3) Buffering (4) Parallel Ports

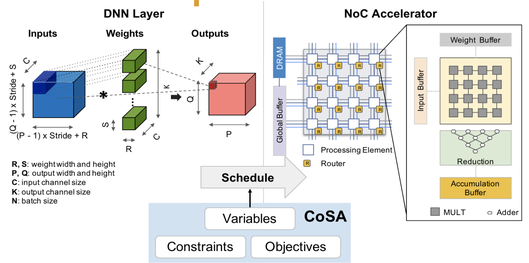
Motivation

- **Increasing DNN model complexity**
- **Growing hardware capacity**
- **More challenges in scheduling and full-system co-optimization**



3. Hardware-Aware Scheduling and Scheduling-Informed Hardware Design

CoSA: Scheduling by Constrained Optimization for Spatial Accelerators



- **Mixed Integer Programming (MIP)** for scheduling DNN without costly brute-force search or autotuning
- Achieves **2x** speedup compared to the state-of-the-art work with **116x** shorter time-to-solution

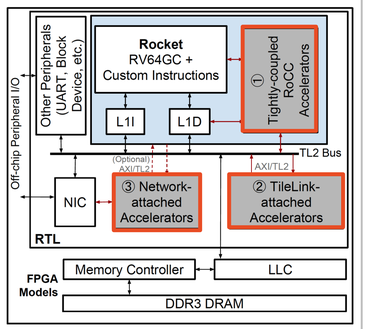
2. Automatic Hardware Generation

Centrifuge: Accelerator-SoC Generation from High-Level Description [ICCAD'19]

Enable rapid accelerator SoCs generation with High-level Synthesis (HLS)

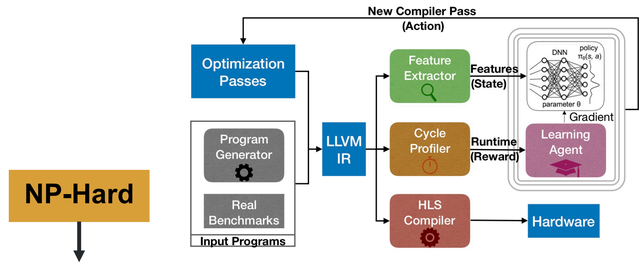
Centrifuge offers the user:

1. Full-system evaluation
2. Fast development and verification cycle for both HW/SW
3. Large design space for rapid algorithm-hardware exploration:
   - Hardware Integration
   - Architectural Design Variation
   - Software Integration



System Diagram

AutoPhase: Reinforcement Learning for HLS Phase-Ordering [MLSys'20]



NP-Hard

- **Phase-Ordering:** Choose and apply compiler optimizations in a good order. It can be formulated as an MDP.
- **Reinforcement Learning (RL)** improves circuit performance by **29%** when compared to using the -O3 compiler flag, and it achieves competitive results compared to the state-of-the-art solutions, while requiring **fewer samples**