TensorFlow to Cloud FPGAs

Tradeoffs for Accelerating Deep Neural Networks

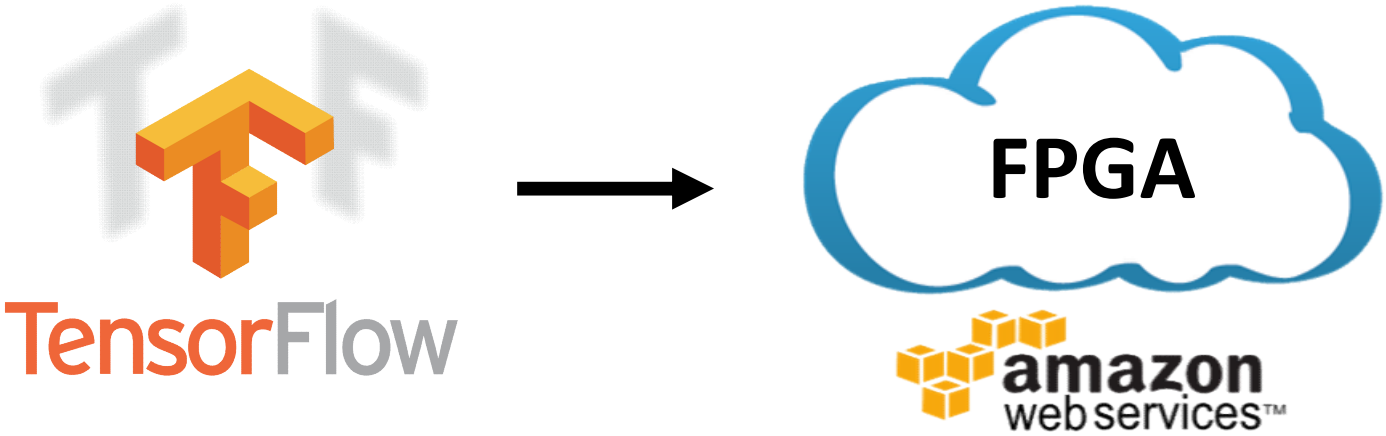
Stefan Hadjis and Kunle Olukotun

Computer Science Department

Stanford University

Summary

- Open-source TensorFlow → FPGA compiler
- Supports the Amazon EC2 FPGA instances
- Can run state-of-the-art DNNs specified in TensorFlow
- Built on top of Spatial, an open-source High-Level Design tool



Overview

- Deep neural networks provide state-of-the-art results in many applications and industries

Overview

- Deep neural networks provide state-of-the-art results in many applications and industries
- FPGAs used to provide improved latency and power compared to GPUs
  - Programmability matters because computations vary across DNNs and ML algorithms often change

Overview

- Deep neural networks provide state-of-the-art results in many applications and industries
- FPGAs used to provide improved latency and power compared to GPUs
  - Programmability matters because computations vary across DNNs and ML algorithms often change

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
  - Architecture Design (operation granularity, hardware algorithm, degree of specialization)

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
  - Architecture Design (operation granularity, hardware algorithm, degree of specialization)
  - Memory Management (on-chip vs. off-chip storage, data format in DRAM, parallelism in memory system)

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
  - Architecture Design (operation granularity, hardware algorithm, degree of specialization)
  - Memory Management (on-chip vs. off-chip storage, data format in DRAM, parallelism in memory system)

Large design space and complex implementation process

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
- DNN applications developed using high-level frameworks and libraries

Overview

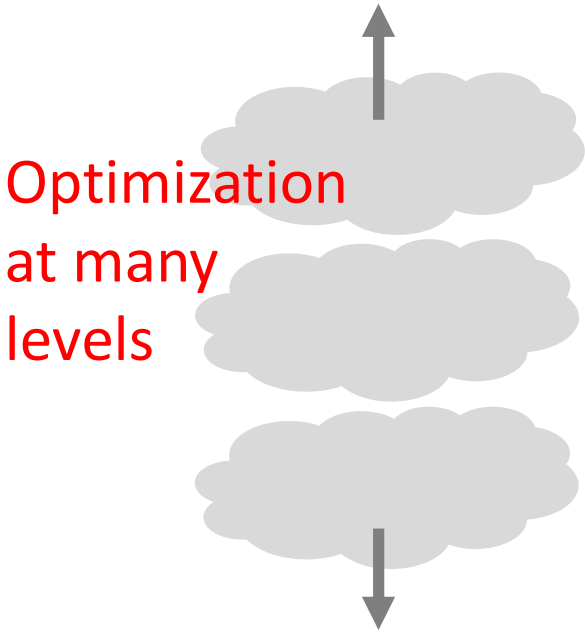
- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
- DNN applications developed using high-level frameworks and libraries

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
- DNN applications developed using high-level frameworks and libraries
- **Problem 2**: Efficiently running a high-level DNN model on a low-level target requires optimization at many levels of abstraction

Overview

- **Problem 1**: Complex process and many choices to design an accelerator for a given DNN
- DNN applications developed using high-level frameworks and libraries
- **Problem 2**: Efficiently running a high-level DNN model on a low-level target requires optimization at many levels of abstraction

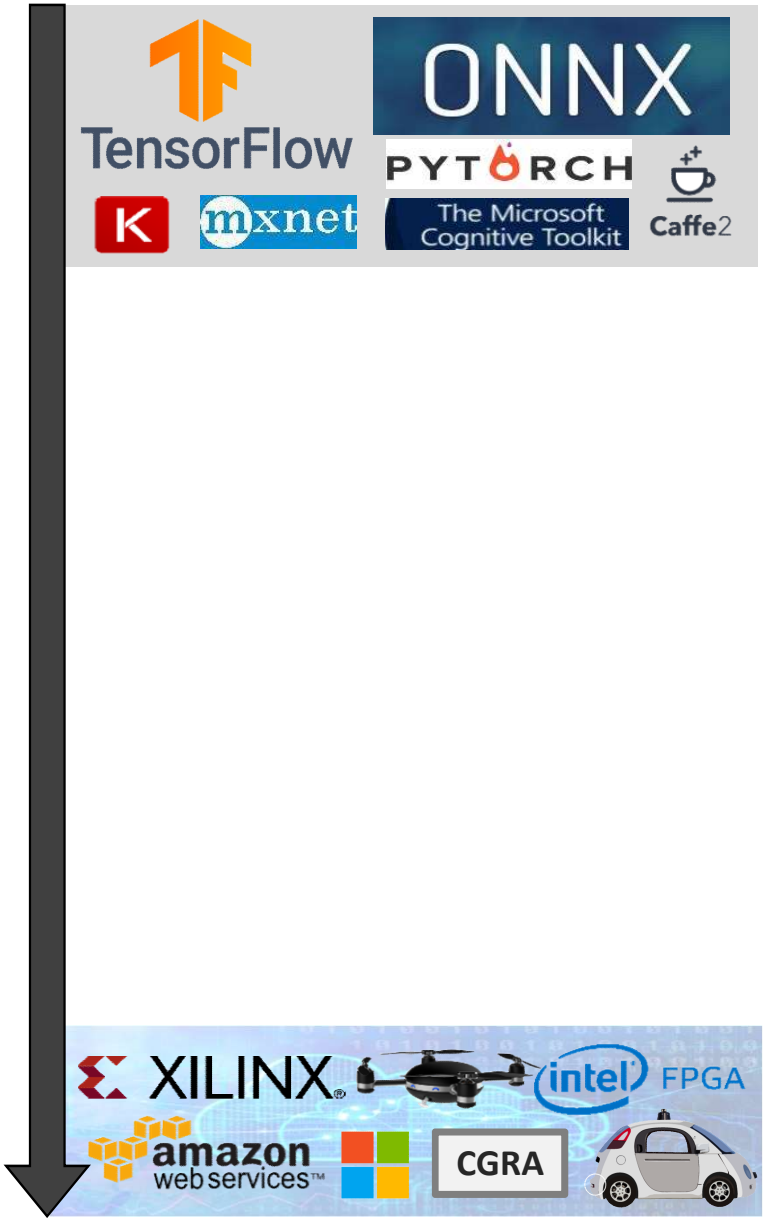


Overview

- **Solution:**

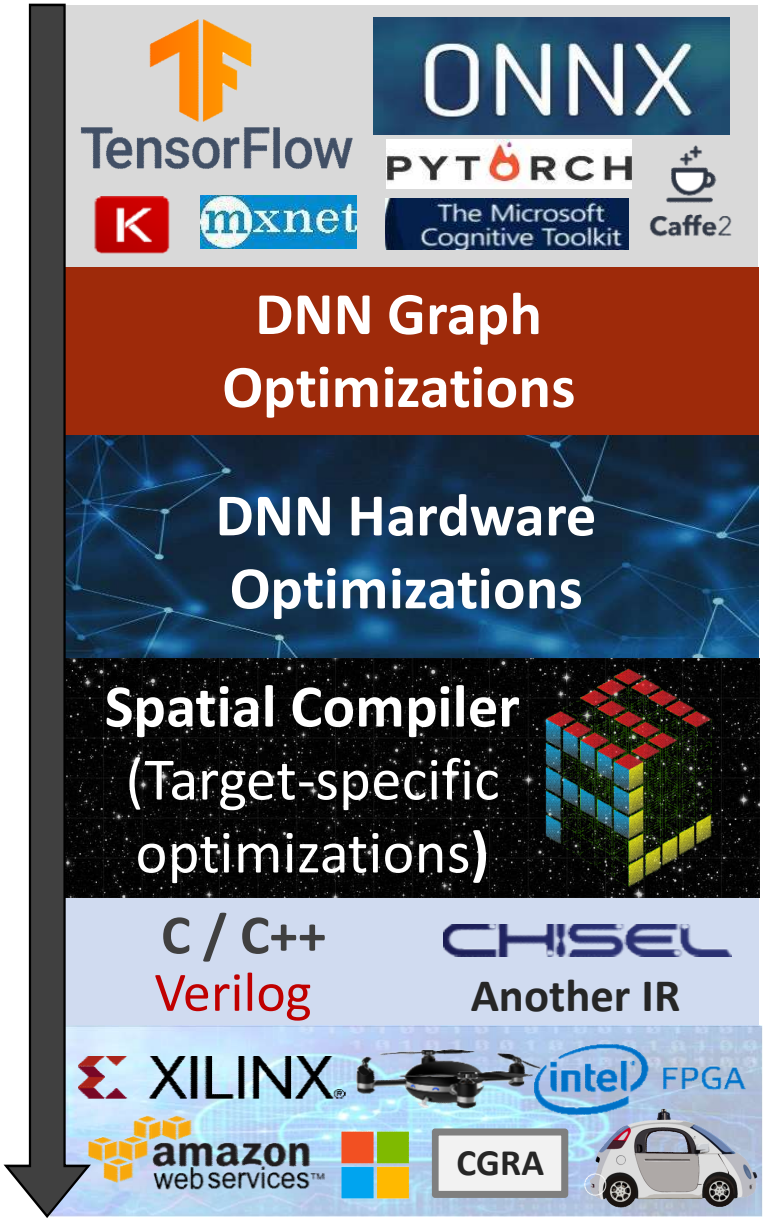
Overview

- **Solution:**



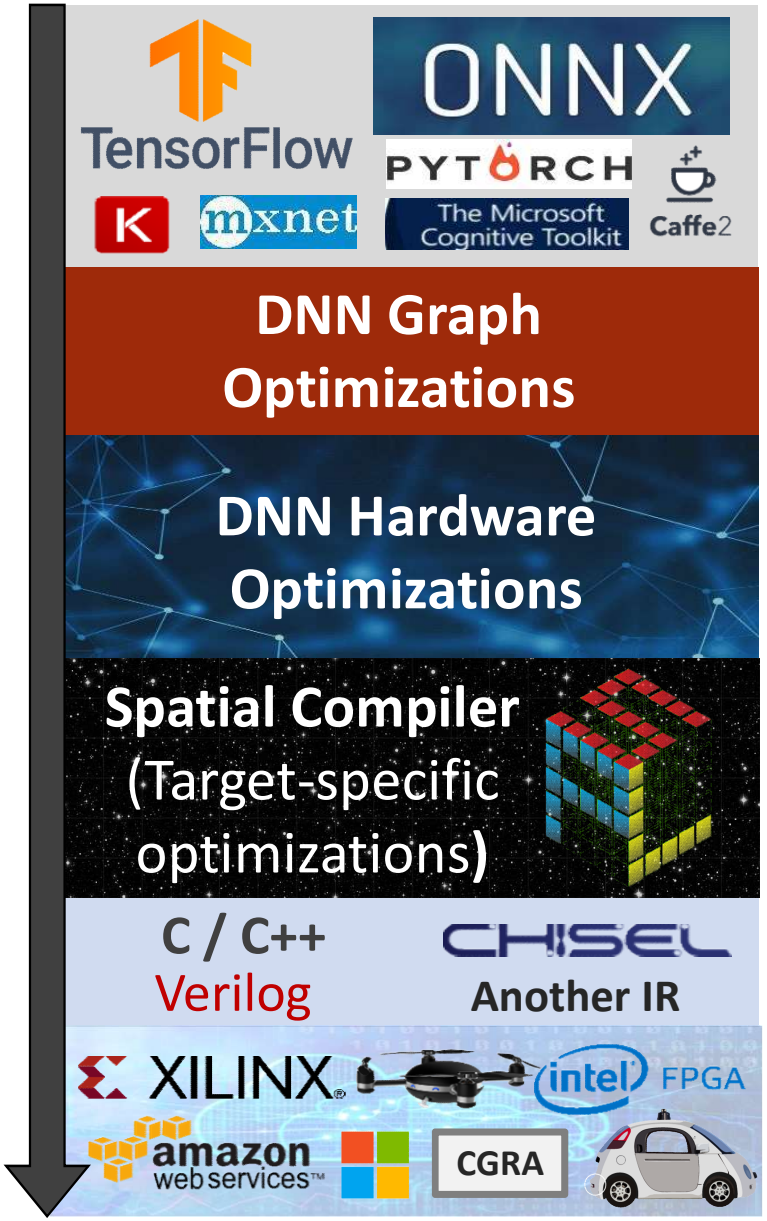
Overview

- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design



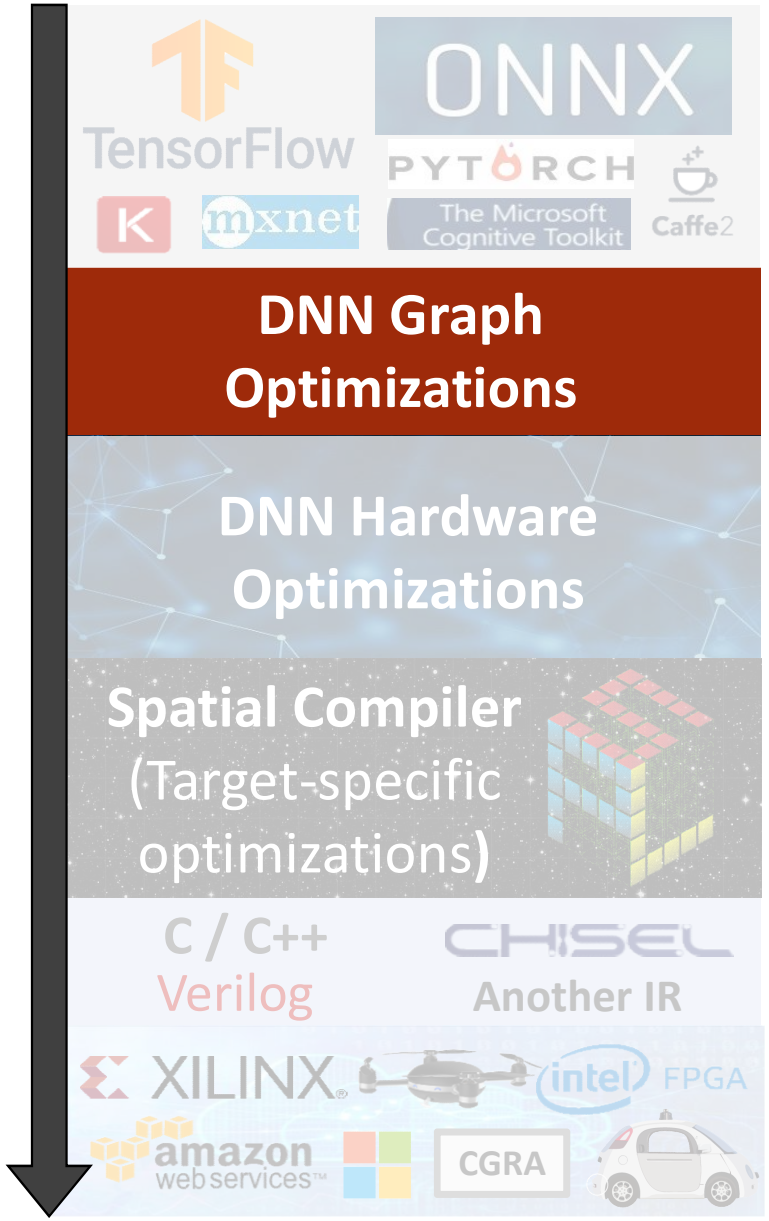
Overview

- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design
- Three Optimization Levels:



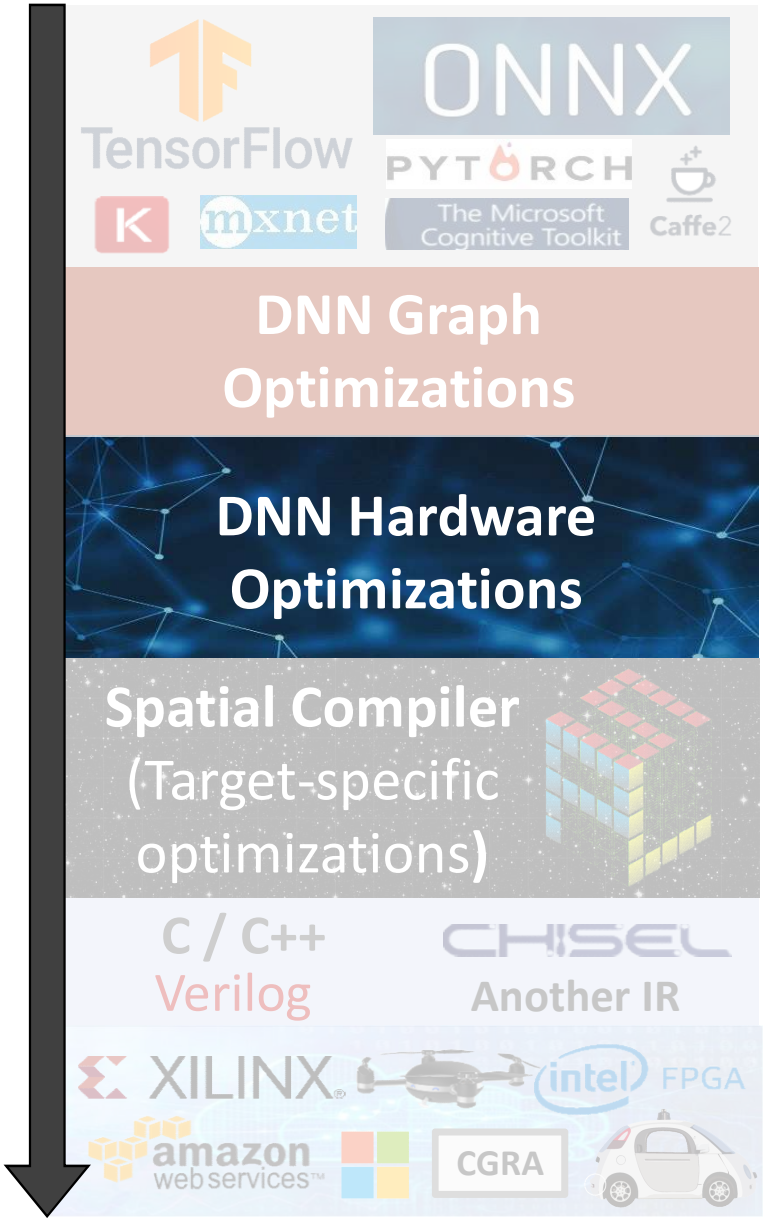
Overview

- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design
- Three Optimization Levels:
  - DNN Graph optimizations



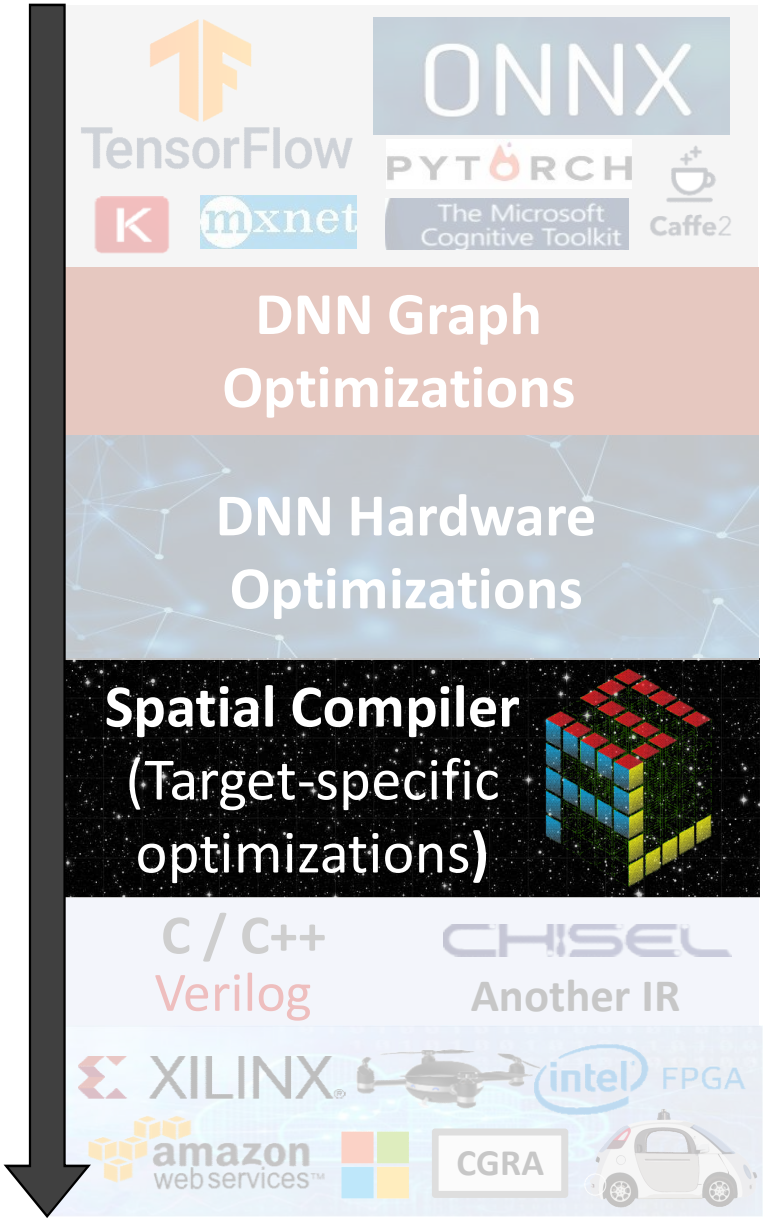
Overview

- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design
- Three Optimization Levels:
  - DNN Graph optimizations
  - Optimizations for DNN hardware accelerators



Overview

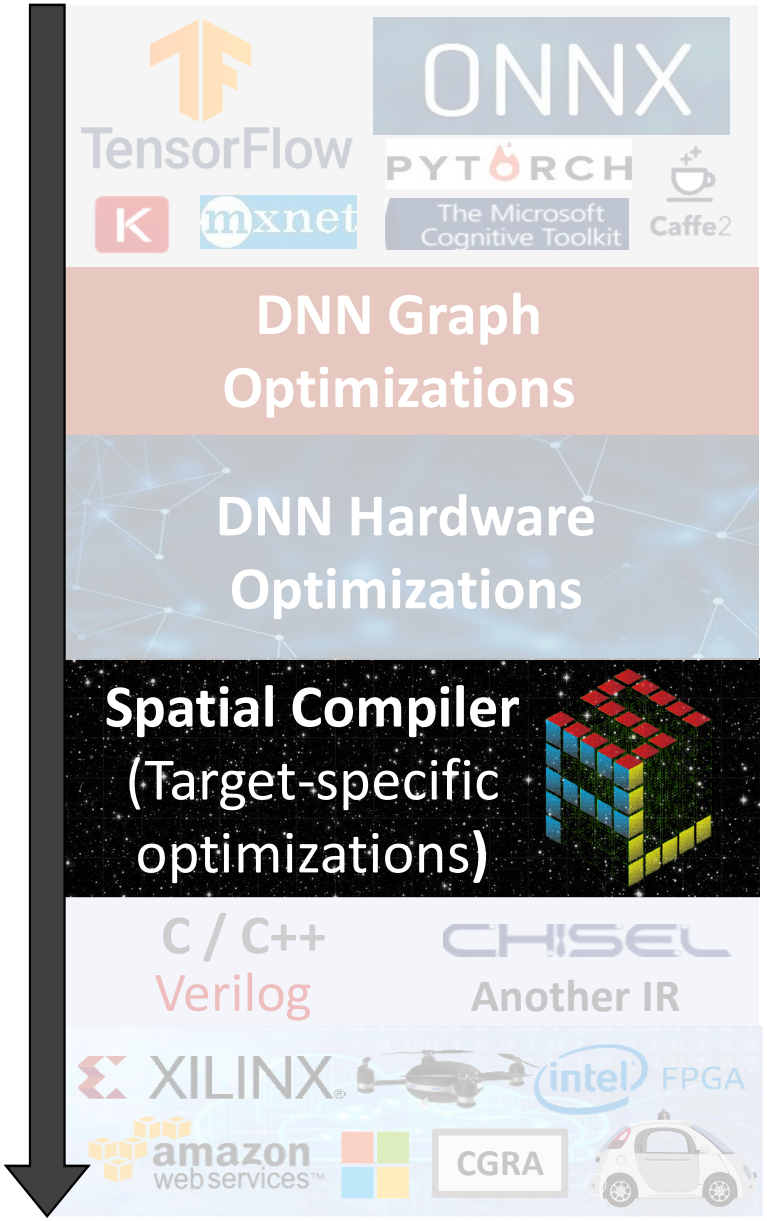
- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design
- Three Optimization Levels:
  - DNN Graph optimizations
  - Optimizations for DNN hardware accelerators
  - DNN-agnostic, target-specific optimizations



Overview

- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design
- Three Optimization Levels:
  - DNN Graph optimizations
  - Optimizations for DNN hardware accelerators
  - DNN-agnostic, target-specific optimizations

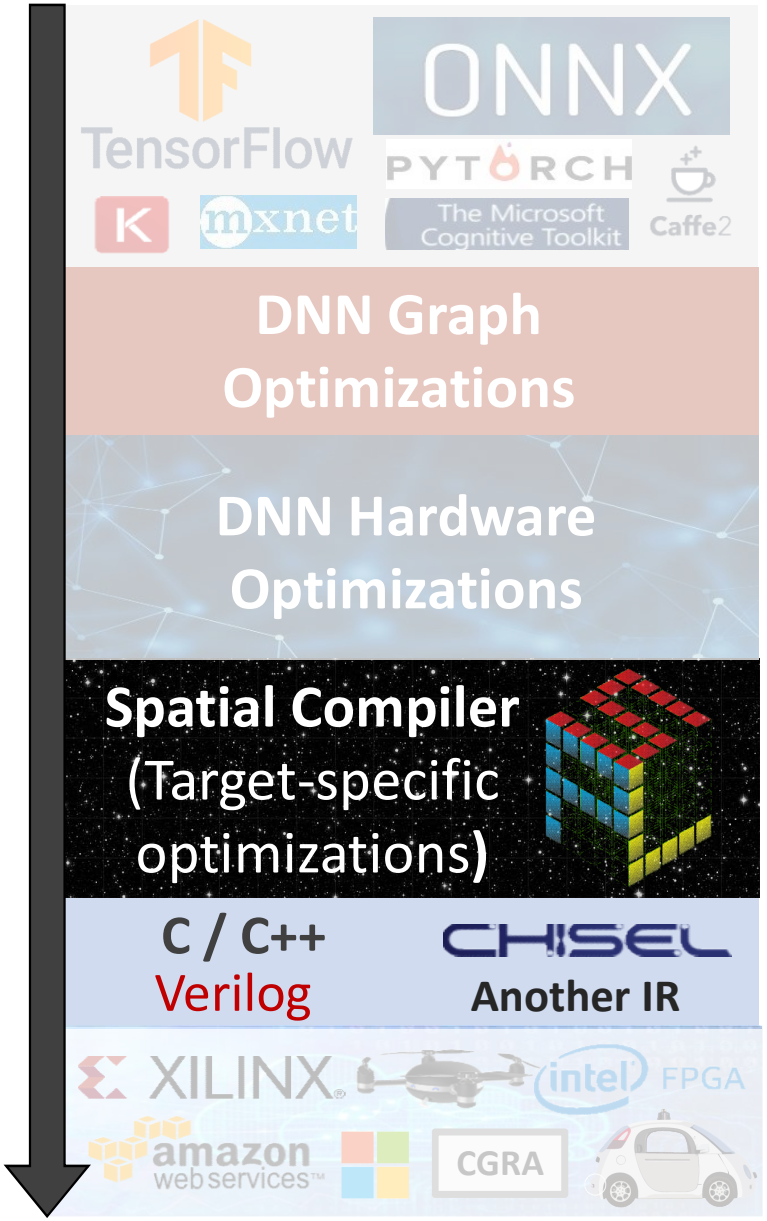
Uses "**Spatial**", a language and compiler for application accelerators [PLDI 2018]



Overview

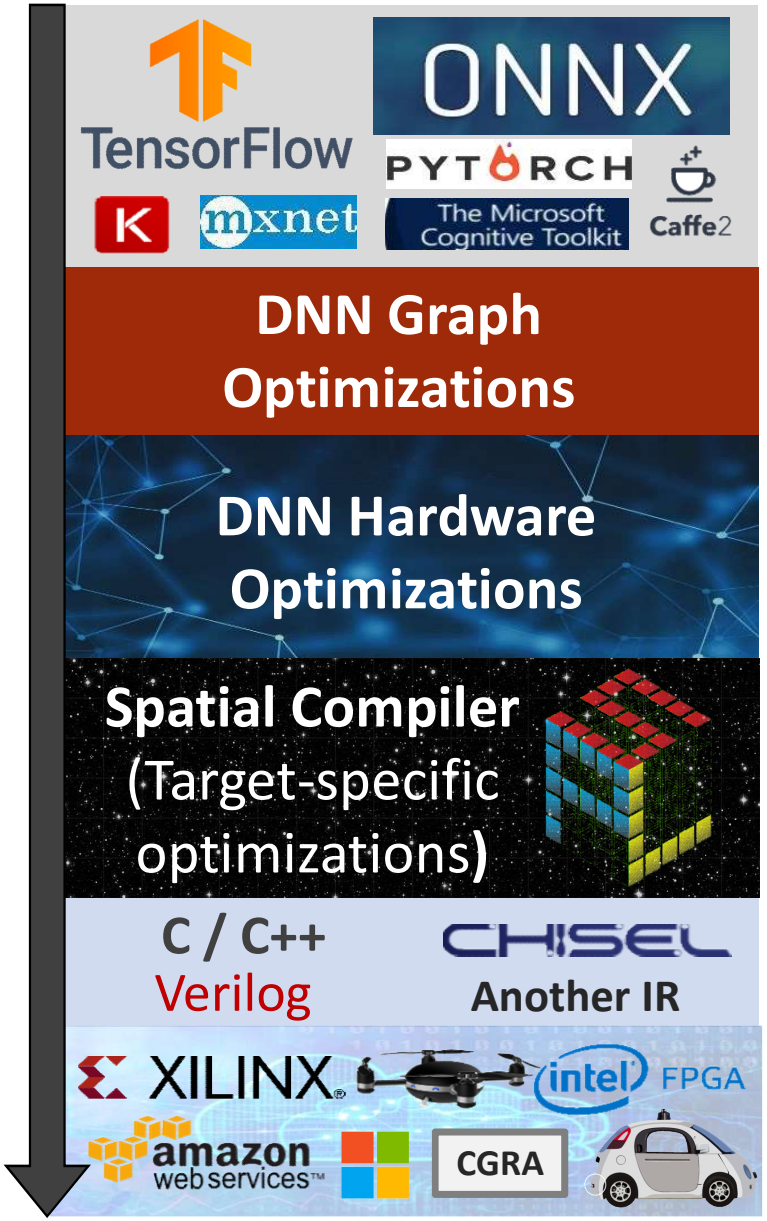
- **Solution:** An end-to-end toolchain to go from high-level DNN models to low-level hardware
  - Input: TensorFlow model
  - Output: Optimized FPGA design
- Three Optimization Levels:
  - DNN Graph optimizations
  - Optimizations for DNN hardware accelerators
  - DNN-agnostic, target-specific optimizations

Uses "**Spatial**", a language and compiler for application accelerators [PLDI 2018]



Overview

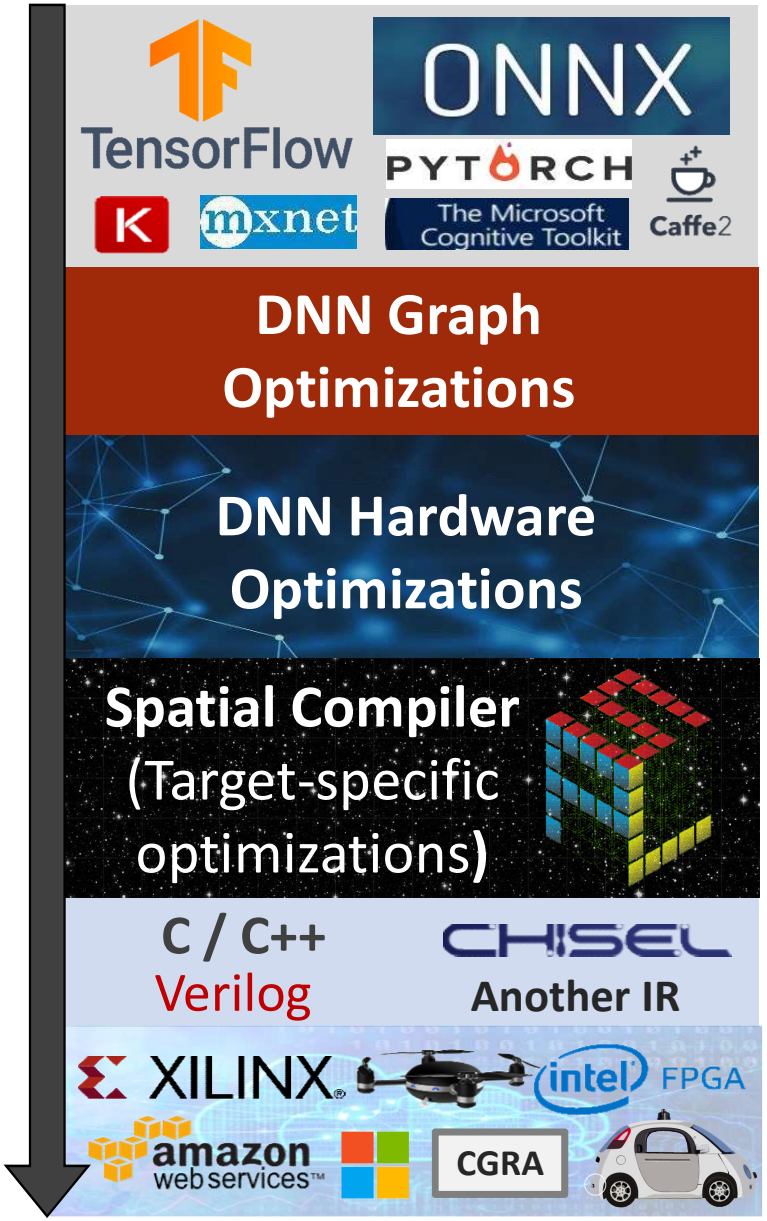
- **Solution 1:** Allows experimenting with architectures, algorithms and design parameters to explore large design spaces
- **Solution 2:** Performs required optimizations at each level of the stack so DNNs expressed in high-level frameworks can be deployed to a variety of hardware targets



Overview

Three Optimization Levels:

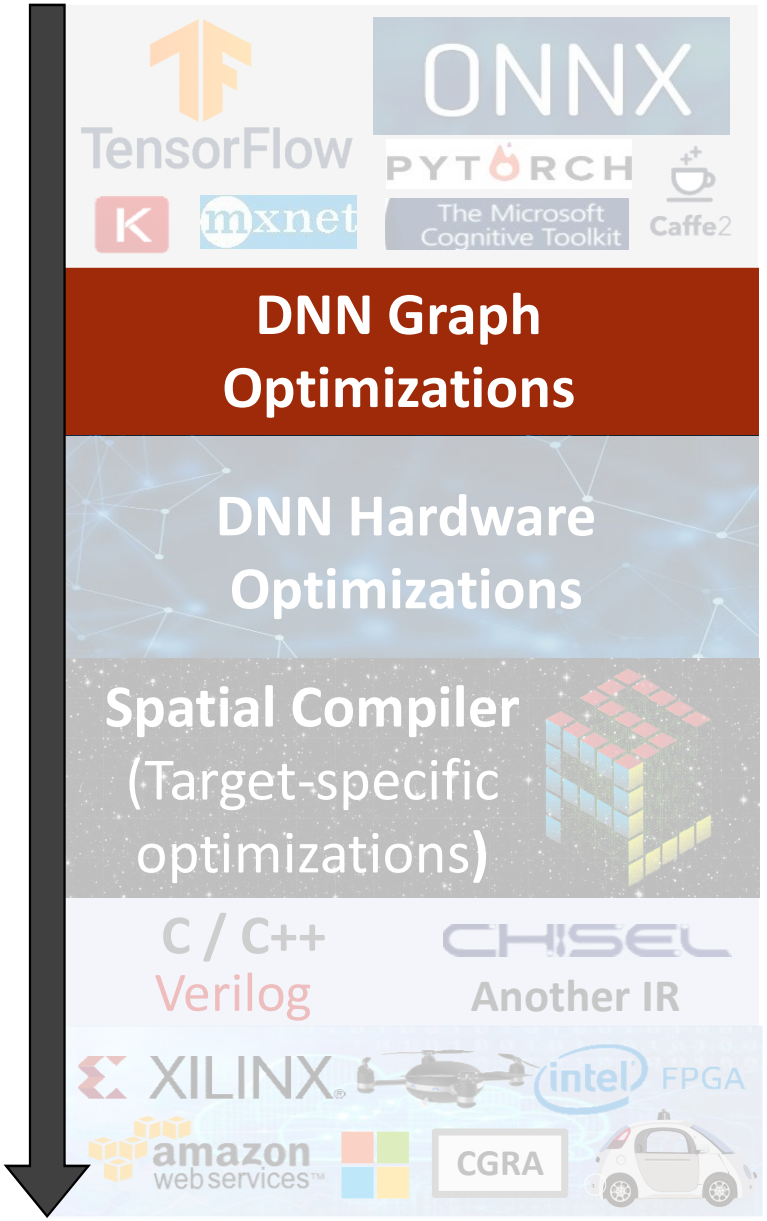
- DNN Graph optimizations
- Optimizations for DNN hardware accelerators
- DNN-agnostic, target-specific optimizations



Overview

Three Optimization Levels:

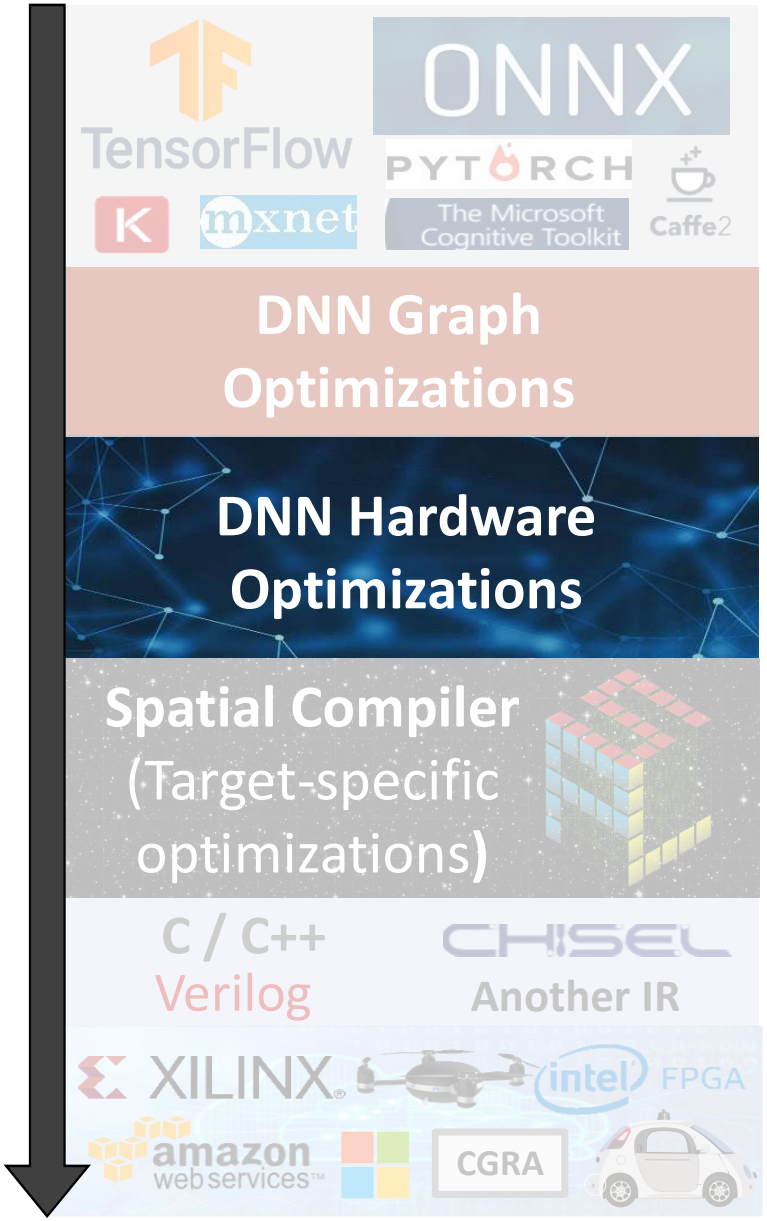
- **DNN Graph optimizations**
- Optimizations for DNN hardware accelerators
- DNN-agnostic, target-specific optimizations



Overview

Three Optimization Levels:

- DNN Graph optimizations
- **Optimizations for DNN hardware accelerators**
- DNN-agnostic, target-specific optimizations

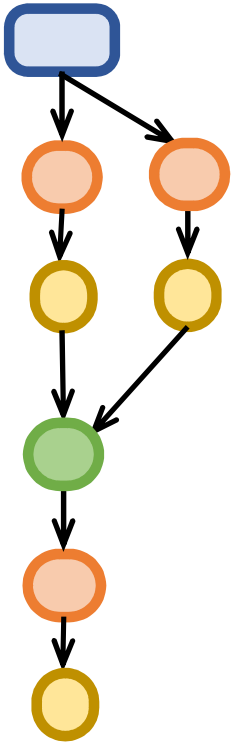


Convert DNN Graph → Spatial Language

- **Goal:** specify architecture layout of DNN on-chip

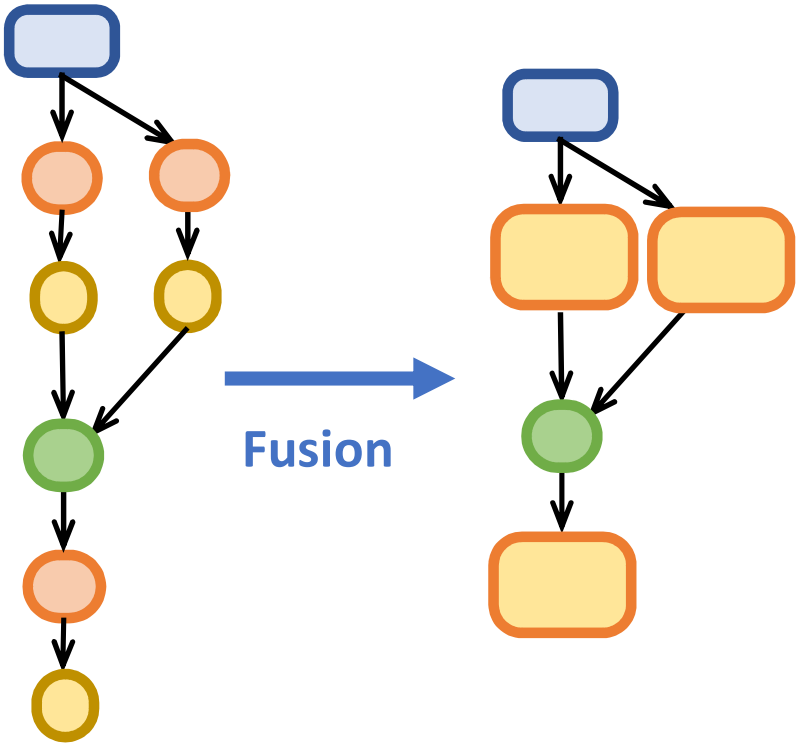
Convert DNN Graph → Spatial Language

- **Goal:** specify architecture layout of DNN on-chip



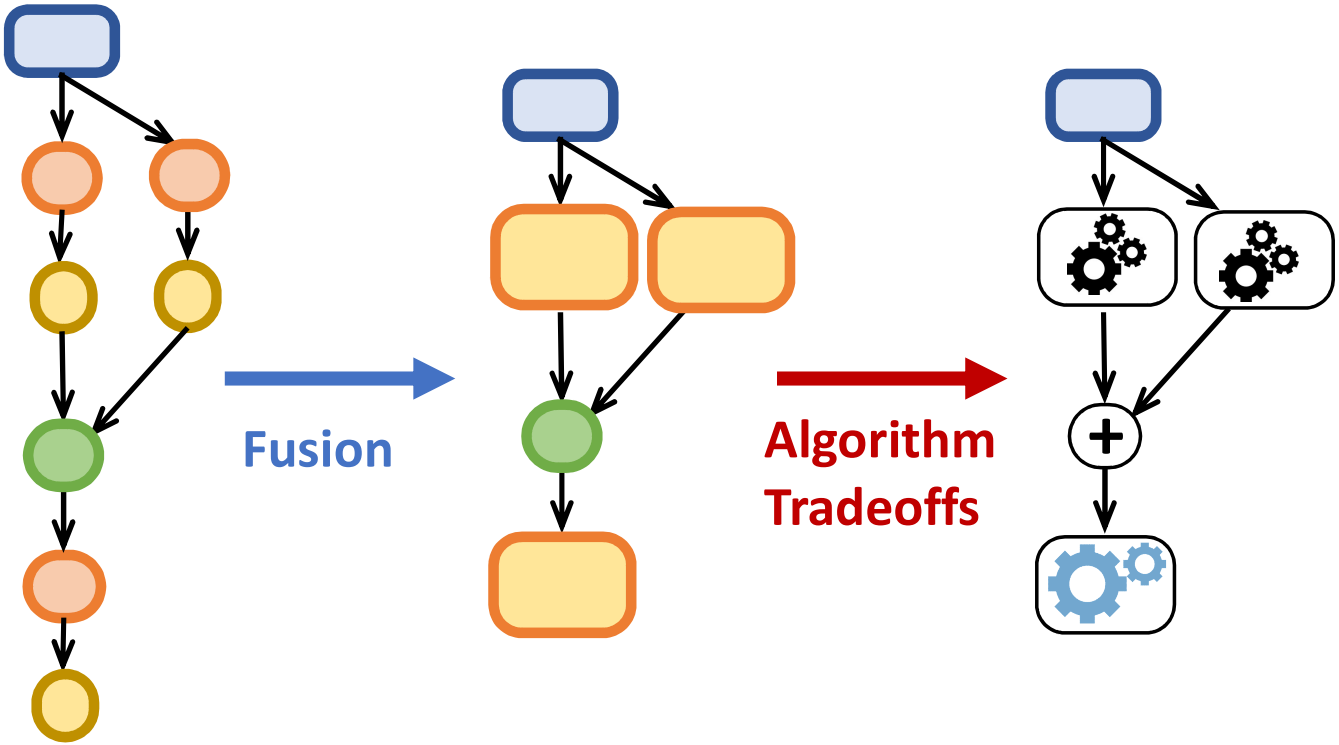
Convert DNN Graph → Spatial Language

- **Goal:** specify architecture layout of DNN on-chip



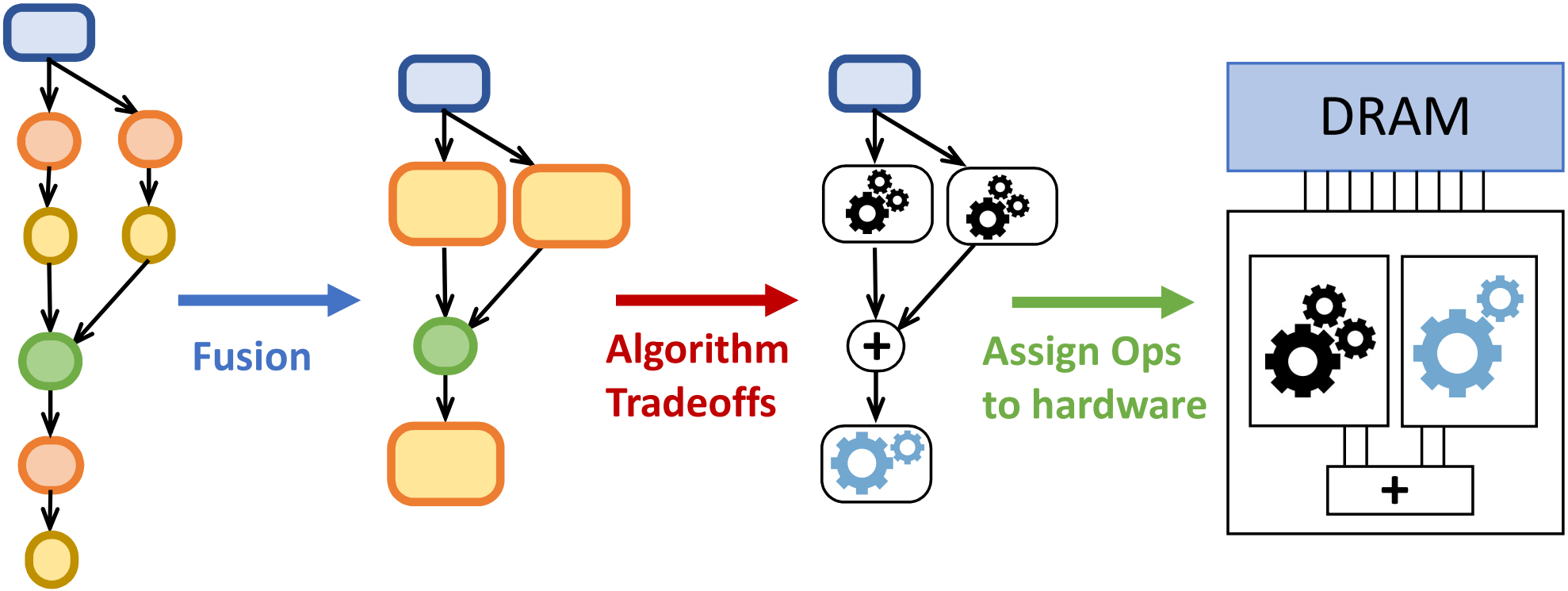
Convert DNN Graph → Spatial Language

- **Goal:** specify architecture layout of DNN on-chip



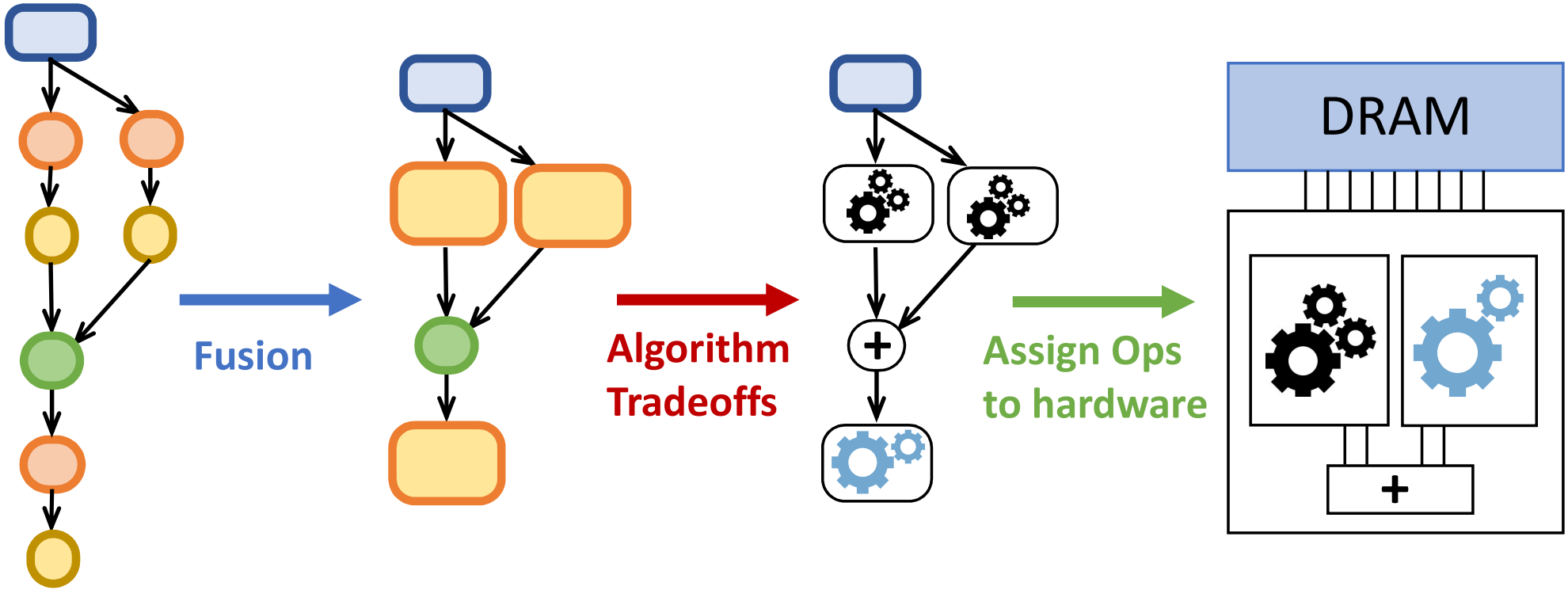
Convert DNN Graph → Spatial Language

- **Goal:** specify architecture layout of DNN on-chip



Convert DNN Graph → Spatial Language

- **Goal:** specify architecture layout of DNN on-chip

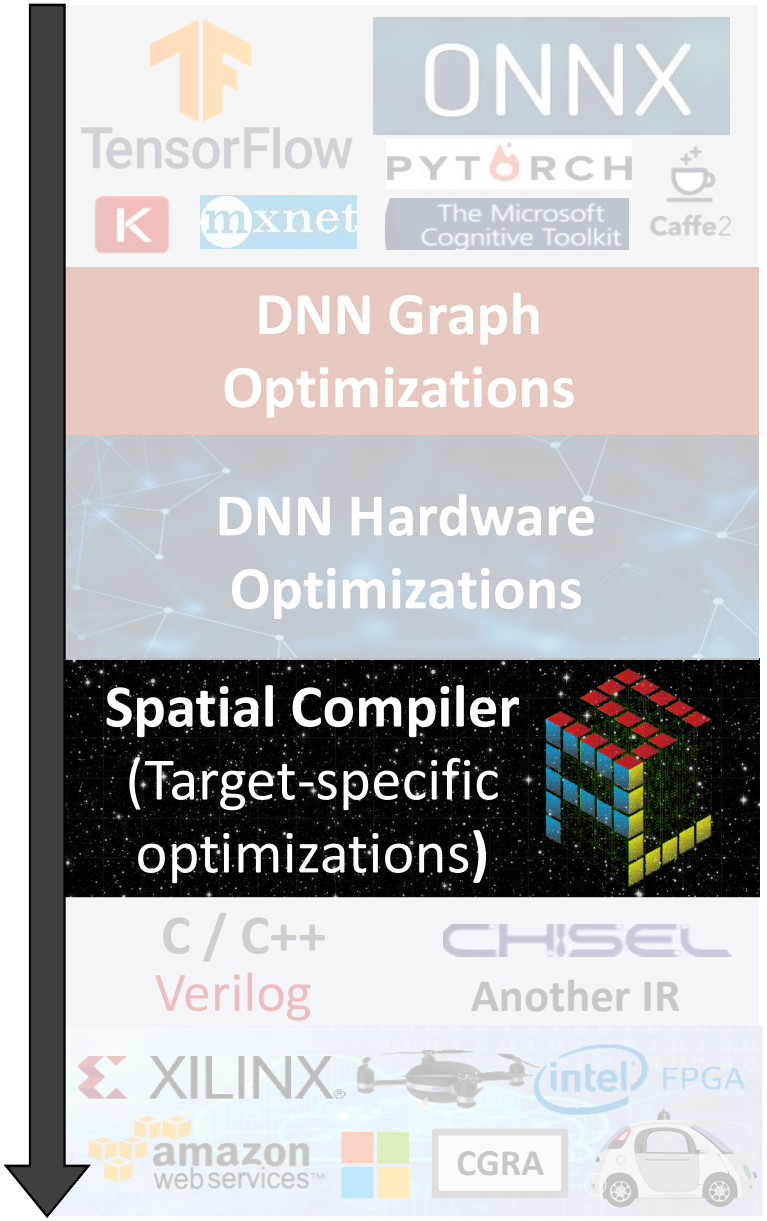


- Then generate Spatial Language program for this layout

Overview

Three Optimization Levels:

- DNN Graph optimizations
- Optimizations for DNN hardware accelerators
- **DNN-agnostic, target-specific optimizations**



Spatial Language / Compiler

- Language/compiler for application accelerators
  - Uses hardware abstractions, e.g. like Verilog but at higher level of abstraction
  - Makes experimenting with algorithms, architectures and design parameters easier (vs. HDL)
- Single source program can be mapped to many hardware targets
  - Optimizes parameters for a target (e.g. operator latencies)
  - Generates a C++ host program and Verilog design for the target FPGA

Current Support

- We currently support CNNs and MLPs
  - Cloud multimedia applications (speech-to-text, ResNet object recognition)
- Working on broader application support and support for edge devices
- Long-term goal: to be an architecture exploration tool like VPR, but for machine learning accelerators
  - Describe DNN / ML graph in a standard high-level format
  - Perform necessary optimizations at each level to allow experimentation with different design strategies

More Information

- Spatial Language and Compiler:
  spatial-lang.org

- TensorFlow to Cloud FPGAs:
  github.com/stanford-ppl/spatial-multiverse