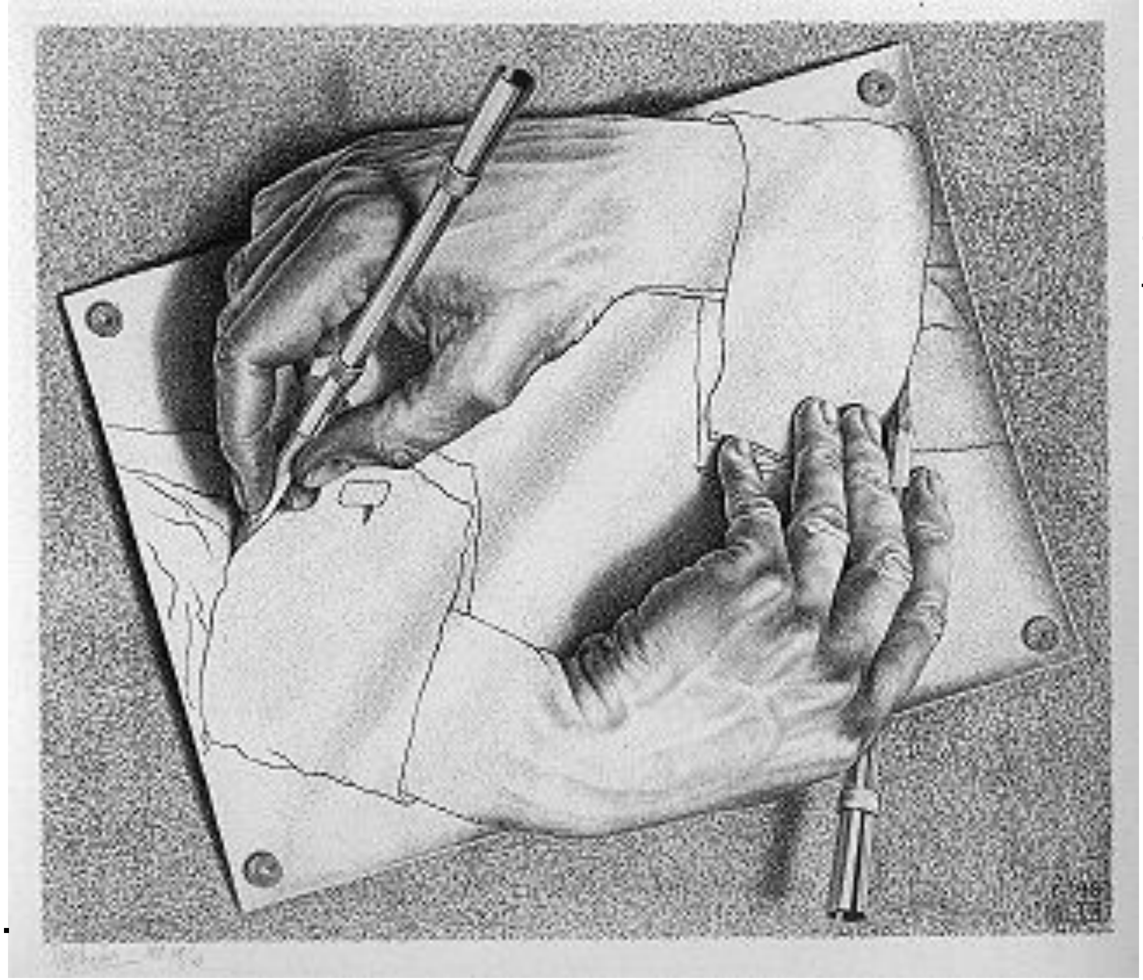
Coevolution of Neural Network and Computer Architecture

zsc@megvii.com

Aug. 2019

Propose new kind of Neural Network for hardware.

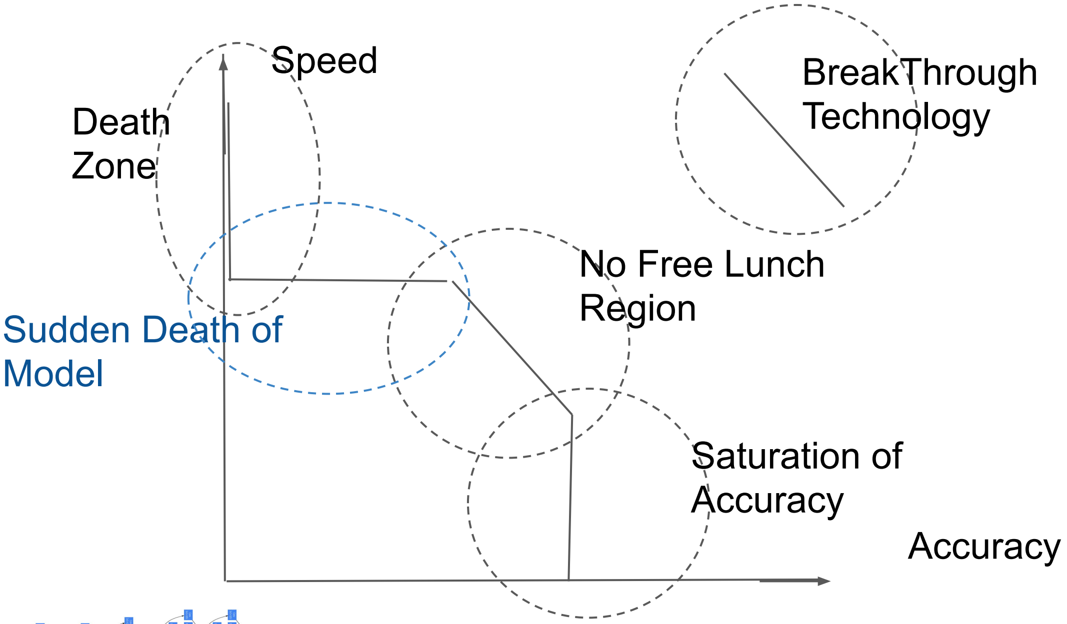


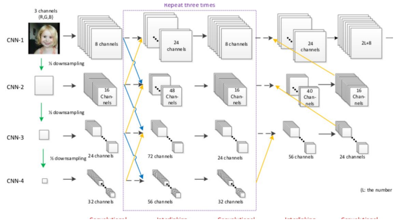
Define new hardware for Neural Network.

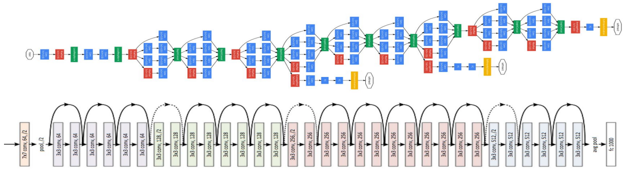
Software-hardware co-evolution

Tradeoff between Accuracy and Speed

- Breakthroughs improve both accuracy and speed
  - Factorized Convolution (GoogleNet)
  - Skip connection (ResNet)
  - Fully Convolutional Network
  - Batch Normalization
  - Cyclic Learning Rate
  - NAS
  - Transfer Learning in NLP

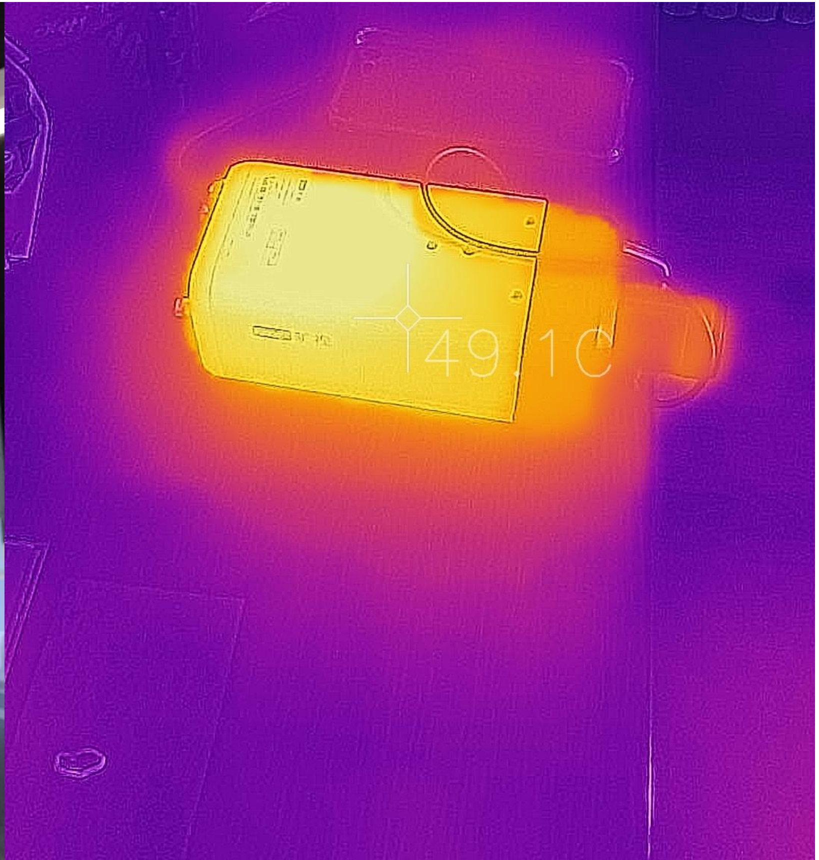






6, QA: 0.86
7, QA: 0.87
2, QA: 0.86
9, QA: 0.84
3, QA: 0.88
13, QA: 0.9
4, QA: 0.9
5, QA: 0.9
15, QA: 0.88
16, QA: 0.81
17, QA: 0.86
24, QA: 0.91
27, QA: 0.87
30, QA: 0.9
32, QA: 0.88
34, QA: 0.89
44, QA: 0.9
46, QA: 0.9
50, QA: 0.84
53, QA: 0.83
55, QA: 0.8
60, QA: 0.87
65, QA: 0.89
71, QA: 0.91
81, QA: 0.85
84, QA: 0.86
86, QA: 0.9
87, QA: 0.88
89, QA: 0.8
92, QA: 0.87
94, QA: 0.83

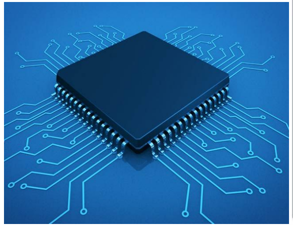
User cases: Deep Learning



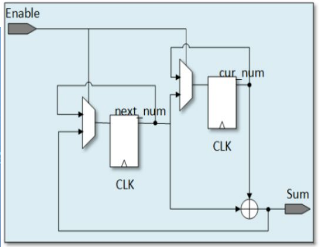
Computer Architecture answer to Deep Learning Challenge

- Make it start: Conceptual Breakthrough
  - GPU: flexible powerhouse
- Make it work: Building product
  - ISA & Programming models: Graph Compiler and Execution Engine
- Make it cheap: Democratize
  - ASIC, Edge Computing, Cloud computing: mass production of all-in-one chips

Computation Stack

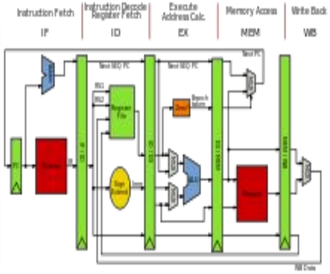


Silicon
- Partitioning & Planning
- Place & Route
- Timing Closure

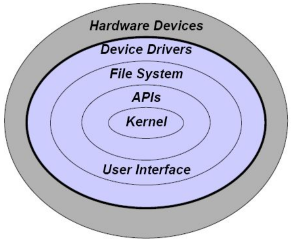


Circuit to Generate Fibonacci Series *(Fig. 13)*

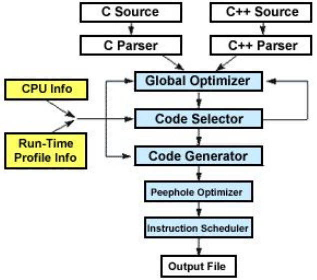
Verilog
- Karnaugh map
- Finite State Machine



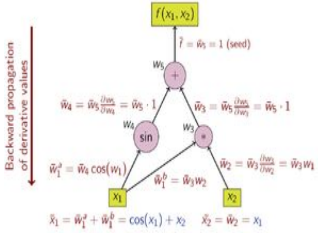
Architecture
- ISA
- Micro-code
- Resource allocation



Operating System
- Page table
- File system
- Interrupts

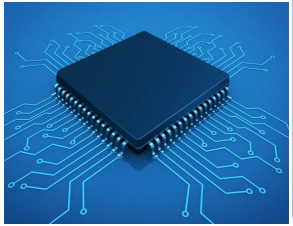


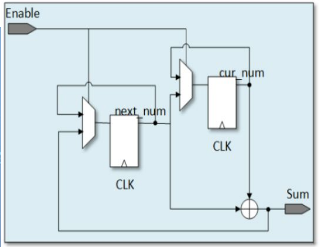
Compiler
- Parallelism mining
- Memory latency hiding



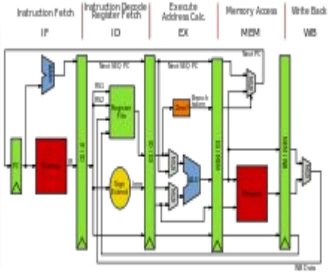
Computation Graph Engine
- Kernels
- Execution Plan

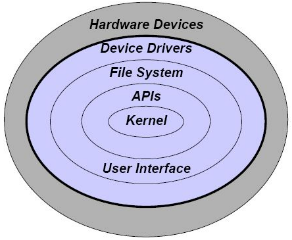
Computation Stack

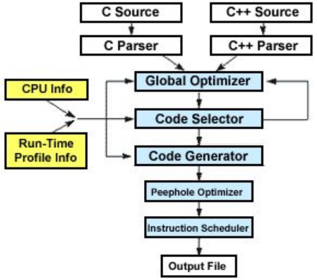


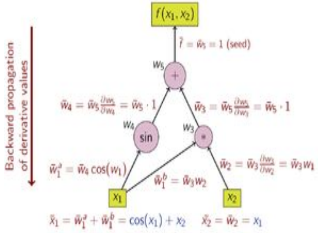


Circuit to Generate Fibonacci Series *(Fig. 13)*









Silicon
- Partitioning & Planning
- Place & Route
- Timing Closure

Verilog
- Karnaugh map
- Finite State Machine

Architecture
- ISA
- Micro-code
- Resource allocation

Operating System
- Page table
- File system
- Interrupts

Compiler
- Parallelism mining
- Memory latency hiding

Computation Graph Engine
- Kernels
- Execution Plan

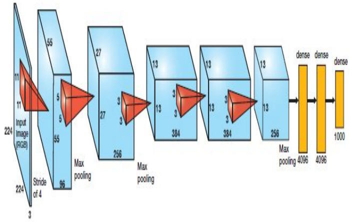
How will this stack deal with changes?

MEGVII 旷视

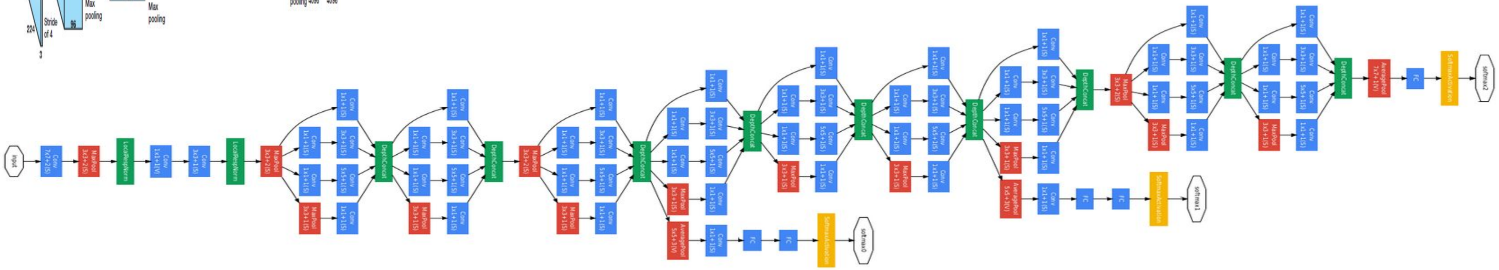
Case study: Large Neural Networks

Characteristics: many channels + side-branches + many layers

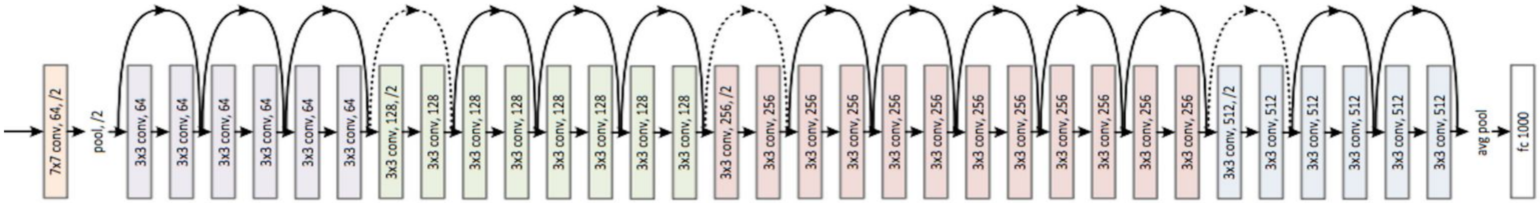
AlexNet



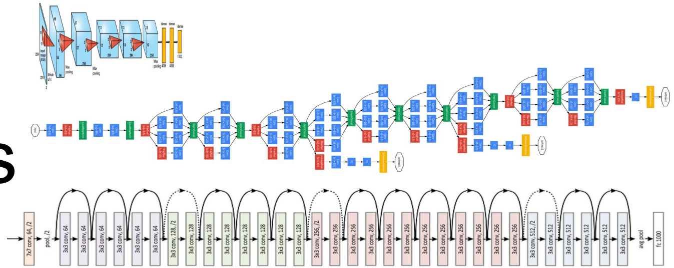
GoogLeNet



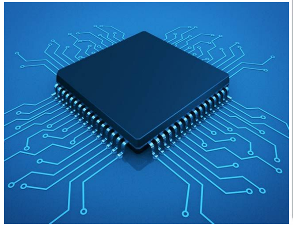
ResNet



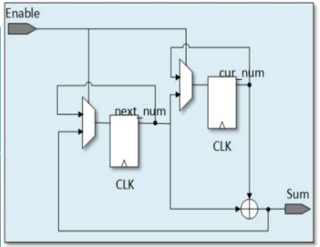
Case study: Large Neural Networks



On-Chip-Memory for caching feature maps



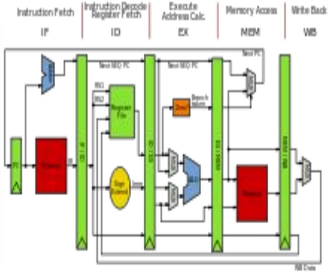
Silicon



Circuit to Generate Fibonacci Series *(Fig. 13)*

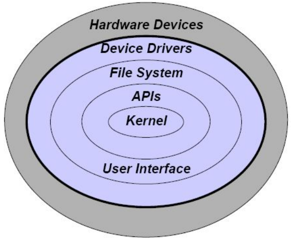
Verilog

- Instructions for convolutions & non-linearity
- Systolic Array



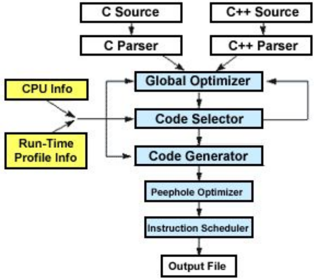
Architecture

Large page-table



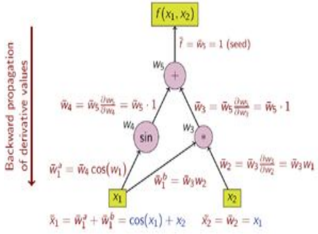
Operating System

Auto-SIMD



Compiler

Static analysis + dynamic profiling for kernel selection + execution plan

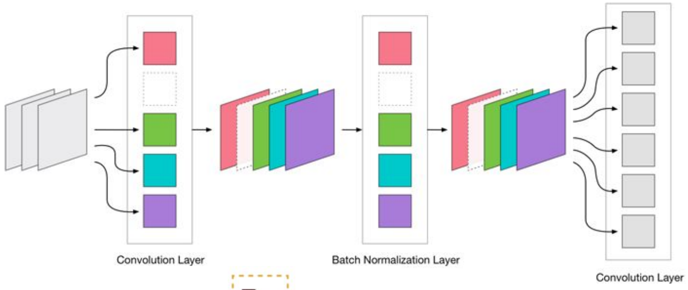


Computation Graph Engine

Case study: Small Neural Networks

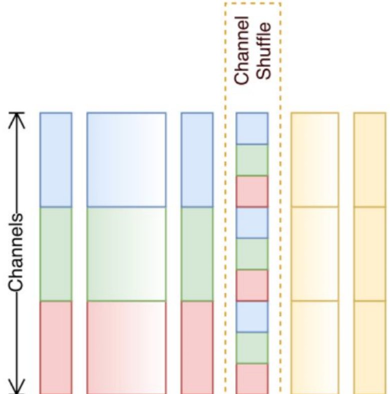
Characteristics: few channels + 1x1 convolutions

MobileNet



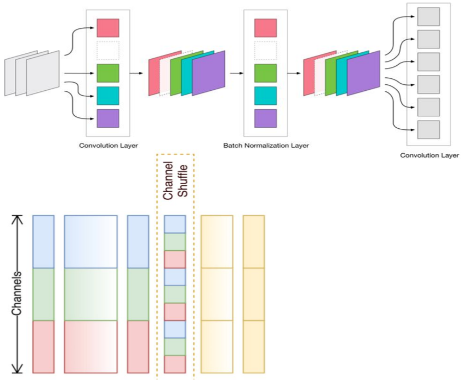
Lack of shortcut hurts its transfer learning ability.

ShuffleNet



The shuffle operation is an efficient way of information mixing, but its uniqueness slows its adoption.

Case study: Small Neural Networks



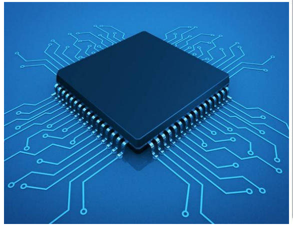
On-Chip-Memory may be more important.

- Specialized support for few channel layers and 1x1 convolutions.
- Different batching

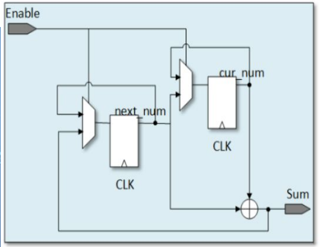
- Lower overhead
- Non-batch perf.
- Page coloring

Auto-SIMD

Fusion of layers + handcrafted kernels

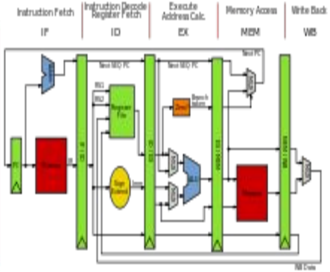


Silicon

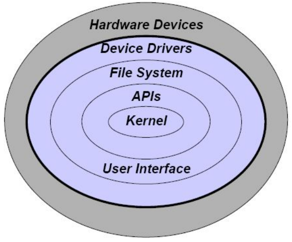


Circuit to Generate Fibonacci Series *(Fig. 13)*

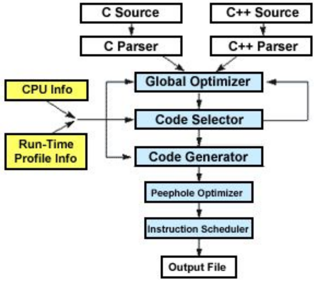
Verilog



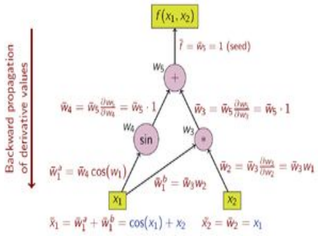
Architecture



Operating System



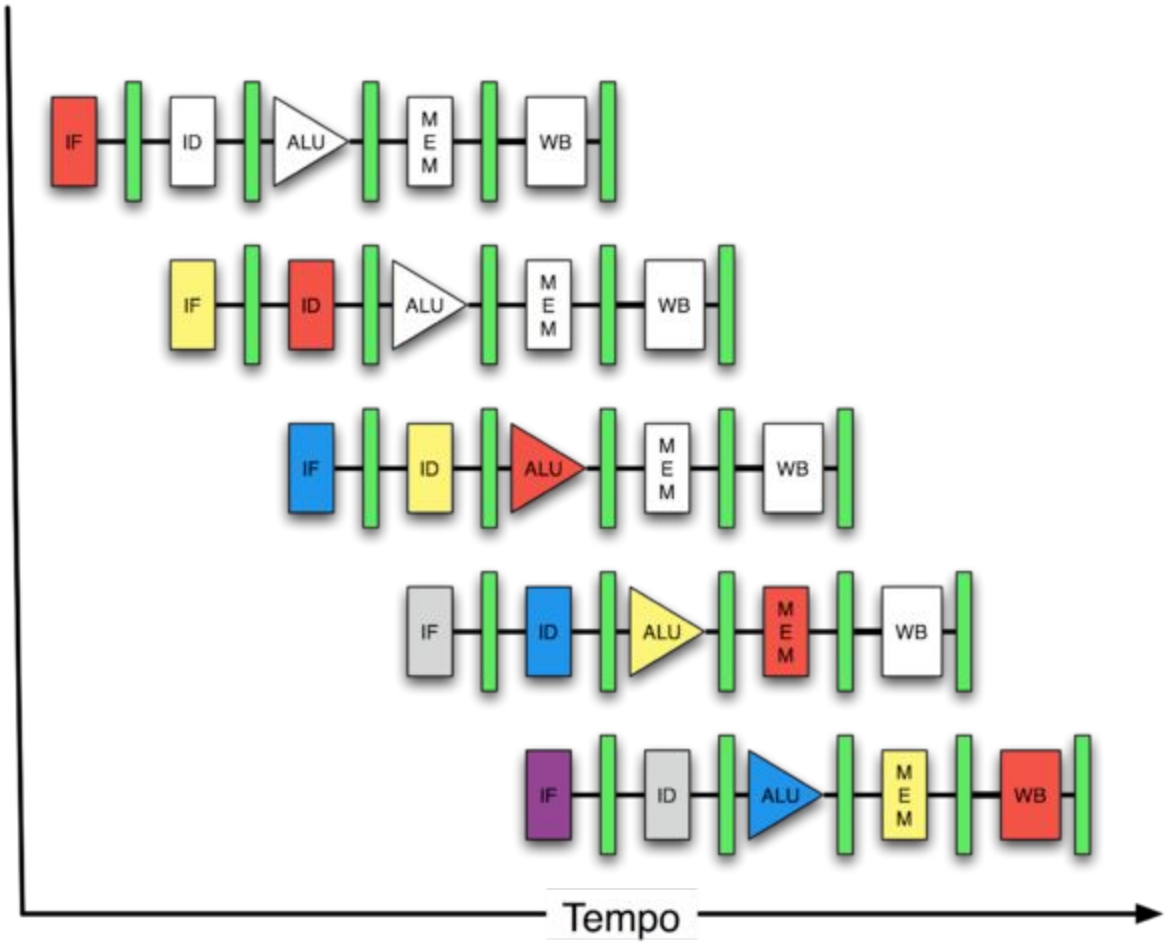
Compiler



Computation Graph Engine

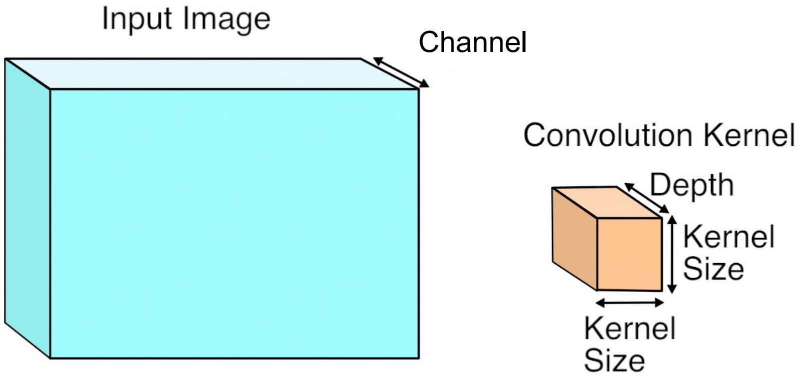
Pipeline approach

- With DL, Need to deal with batch data to improve computation / memory ratio



Instruction Size vs. Feature Size

- Composition of Instruction
  - Type: conv, concat, unpool, stride
  - Weights
- Feature / Weight = N * H * W / (K * Kh * Kw)
  - Feature = N * C * H * W
  - Weight = K * C * Kh * Kw
  - Getting smaller when later in Detection/Recognition NN
  - Relatively stable in image generation NN
- Efficiency suffers from low feature / weight ratio when small batch size



When a Neural Network Designers, a Computer Architect, a Compiler Expert and an OS Guru meet

- Designer wants
  - A reliable performance model
    - Open architecture design and assembly/microcode level exposure
  - Better profilers for runtime diagnostics and analyzers
  - Support for sparse matrices, dynamic operations
- Architect wants
  - Batch operations with constant delays
  - Regular memory access pattern subject to locality and many reuses
  - Streamlined memory/computation usage, no overwhelming peaks
  - Less number of operators
- Compiler Expert and OS Guru wants
  - To broker between the Designer and the Architect
    - Have a slow fallback for bizarre operators
    - Cutting peaks

MEGVII 旷视

Coevolution

Neural Network Designer

We train our DL models and design our networks!

Computer Architects

We design our processors and computers, from ISA to PCB!

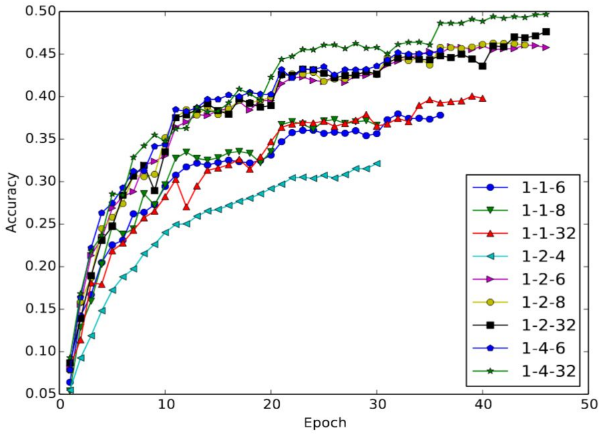
AI-product Programmers

We build our AI-products! From Javascript to Linux Kernels!

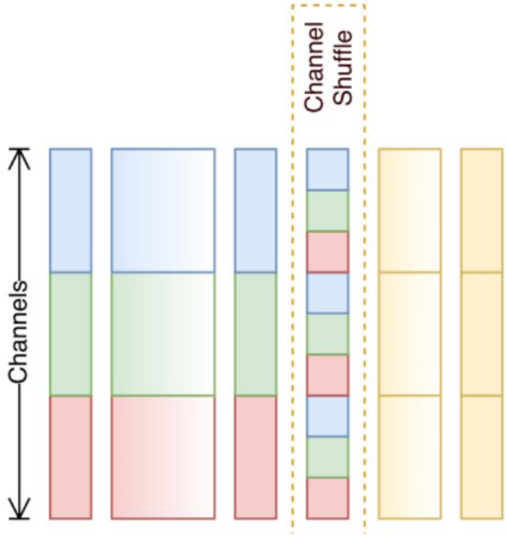
S-platform
Edge-computing platform

A-firmware
Edge Device Firmware

Neural Network Designer



DoReFa-net



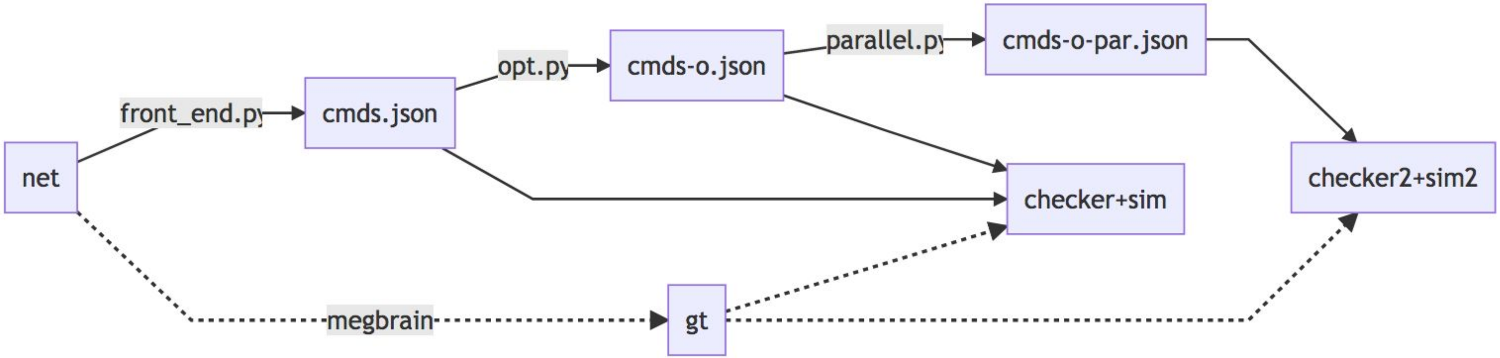
ShuffleNet

Example: self-adjustable global channel quota

- k samples from n without replacement
  - follows multinomial hypergeometric distribution
  - satisfies EX_i = p_i
- SVHN-cropped:
  - channel distribution
    - base: 3 96, 128, 128, 256 | 512, 10; train 19/sec, misclassify 0.027
    - self-adjusted: 3, 17, 26, 100 | 409, 10; train 62/sec, misclassify 0.031
      - 50% less #channel
    - self-adjusted: 4, 23, 51, 191 | 512, 10; train 41/sec, misclassify 0.028
      - 30% less #channel

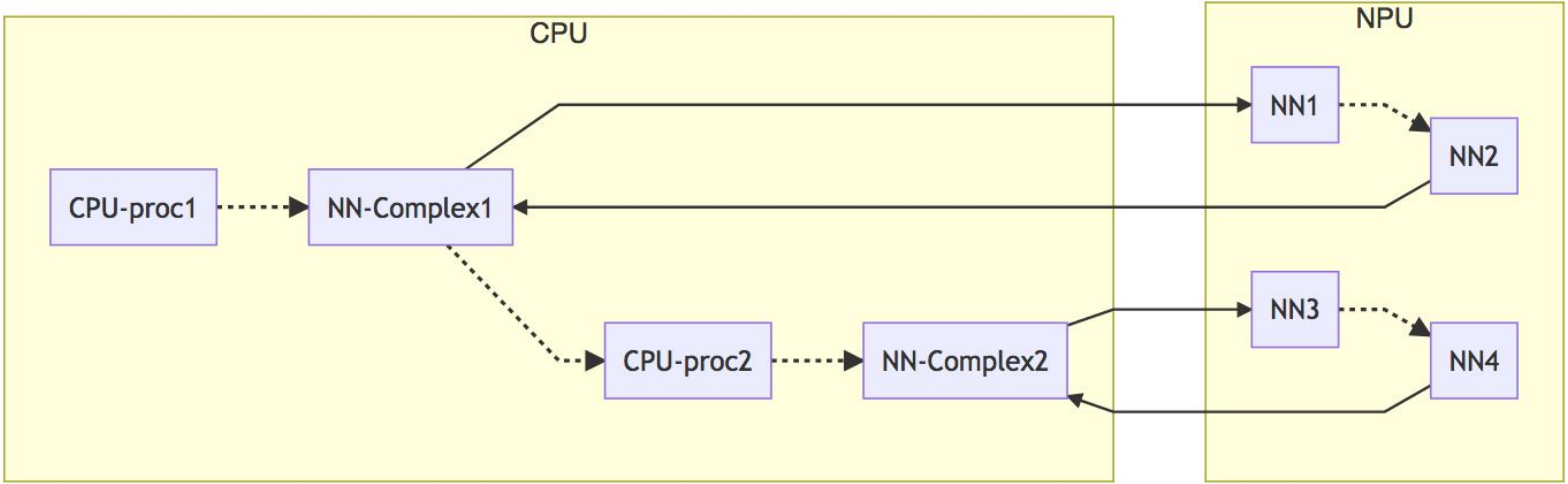
Computer Architect

"X Compiler": Optimizing & Autopar Compiler



MEGVII 旷视

Pipeline Scheduling

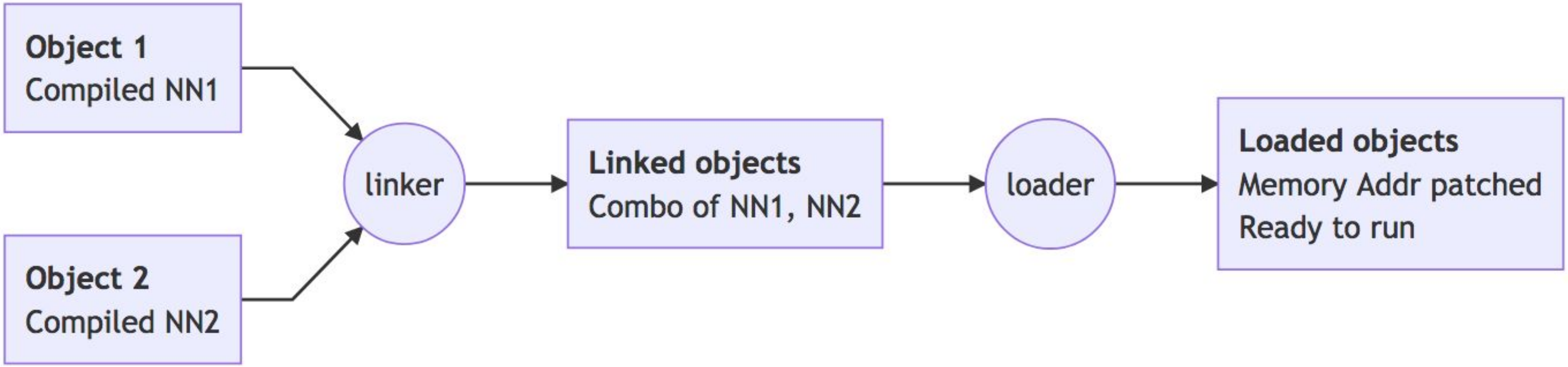


Single Neural Network: Semi-Static Scheduling

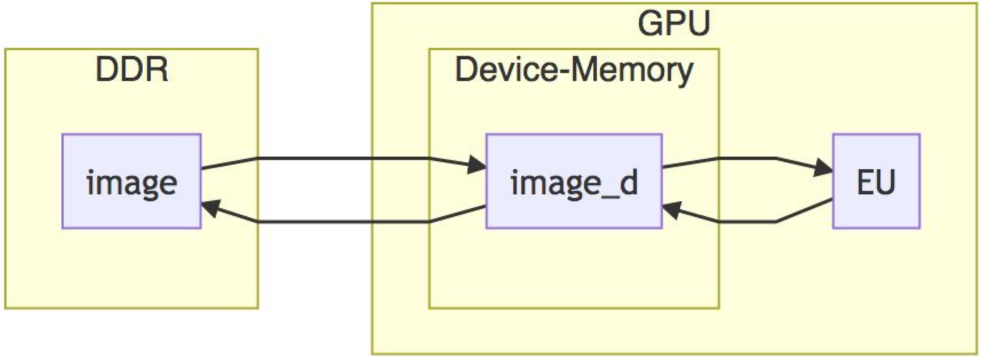
- Neural Networks are almost static
  - No branching
  - (almost always) Fixed length data: fixed input/output/intermediate size
  - Regular computation
- But there are "clouds"
  - DDR latency / bandwidth
  - Cache
- Dynamic Scheduling inevitable?

Multiple Neural Network: Semi-Static Scheduling

- Combo-NN
  - Multiple NN's may be triggered by the same chunk of input data
  - Though logically separate, can be "linked" together
  - Ad-hoc on-the-fly combo by JIT

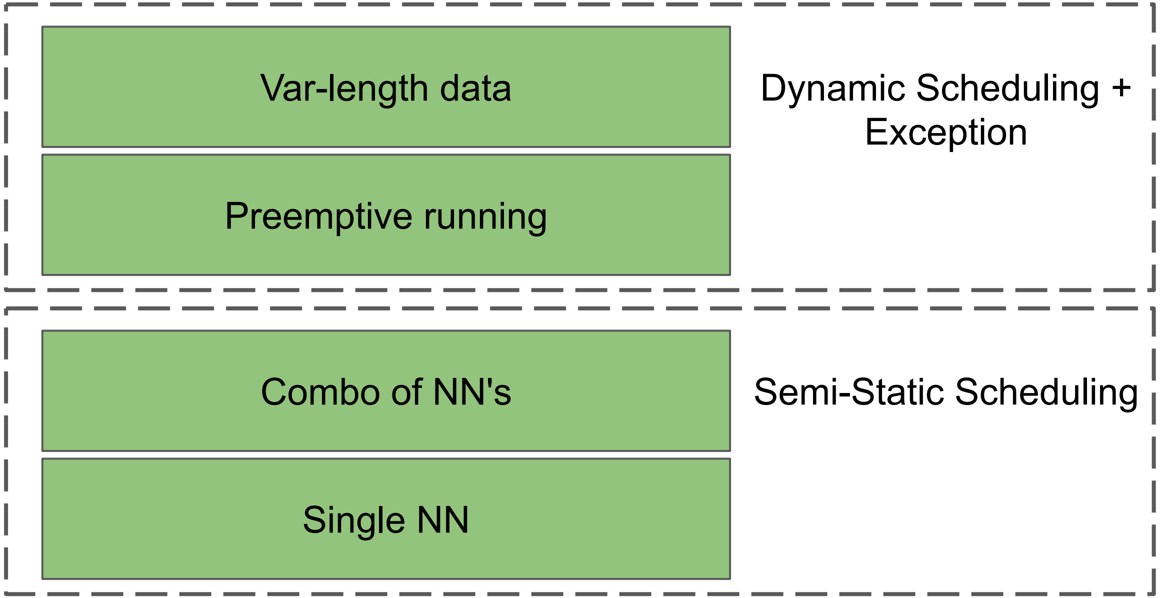


Dyanmic Scheduling



- Complex memory state
  - Can Scheduling ensure OOM-free?
    - Interval analysis
  - Ensure proper recycling of resources when preemption
  - Exception mechanism
    - Spilling data to DDR when below watermark
    - May still not be safe
- Complex running time
  - Interrupts to CPU
  - Unbounded running time: Disk-level access

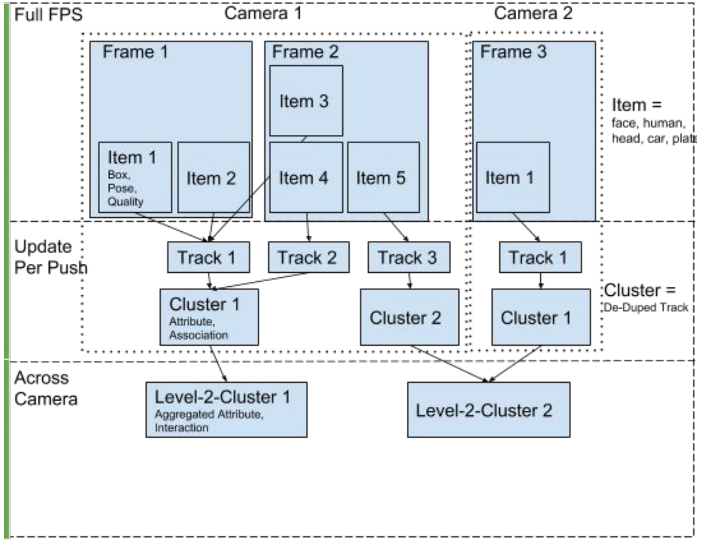
Scheduling Overview



MEGVII 旷视

AI-product Programmers

S-platform: Edge-computing platform

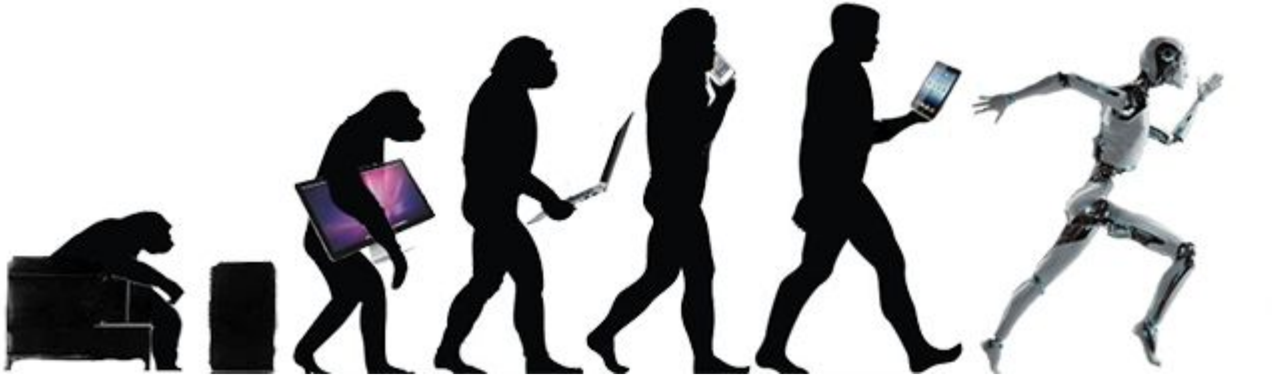


A-firmware: Edge Device Firmware

Backup after this slide

Deep Learning Challenge

- Make it start: Conceptual Breakthrough
- Make it work: Building product
- Make it cheap: Democratize



https://medium.com/global-silicon-valley/the-evolution-of-mobile-computing-d273f23eda61

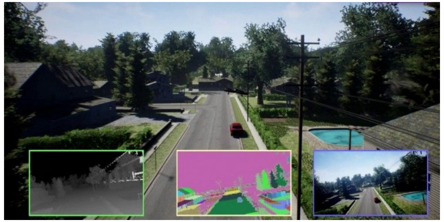
User cases: Reinforcement Learning

Characteristics: require fast & complex simulations

OpenSim

A human skeleton model for locomotive task modeling.

GTA 5
AirSim



Simulation for self-driving car/ADAS and Drones.