AI, app-defined OS, & the ossified Linux kernel

Felix Xiaozhu Lin 林小竹

http://xsel.rocks

PURDUE ENGINEERING

Summary

Describe recent trends of OS research

Present three cases of our investigation

Share personal reflection on OS research

Self intro

- 2014 – now. Asst. prof, Purdue ECE
- 2014 PhD in CS. Rice U (advisor: Lin Zhong)
  - Thesis: OS for mobile computing
- 2008 MS + BS. Tsinghua U

Crossroads Systems Exploration Lab

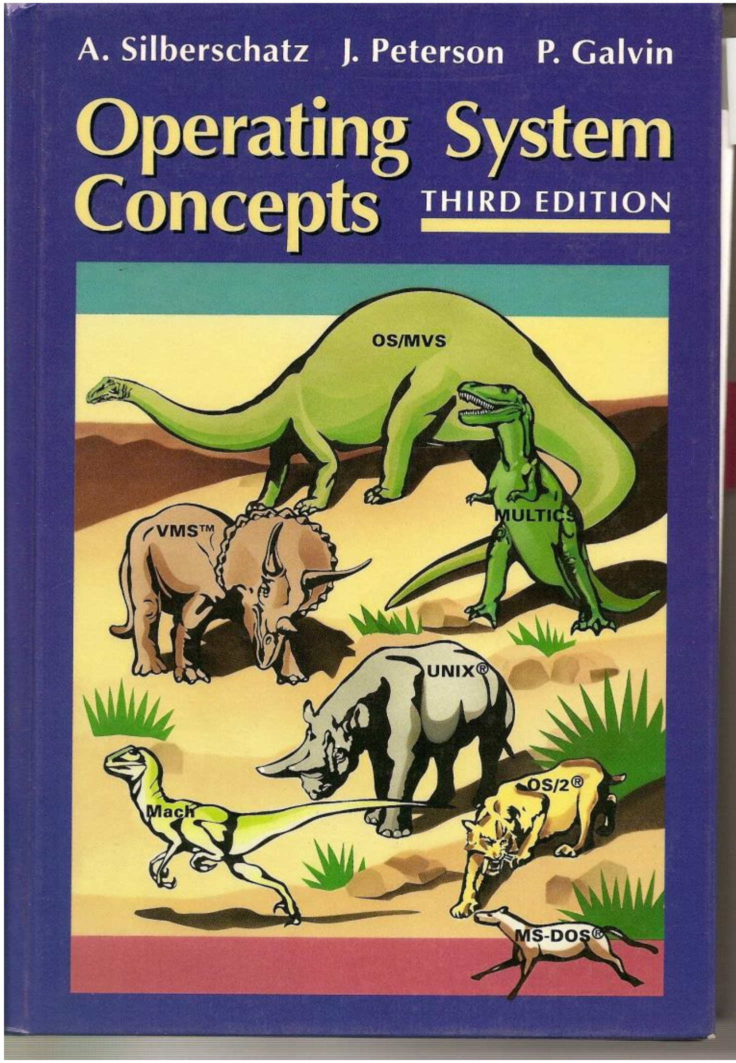
XSEL & Friends, 2017

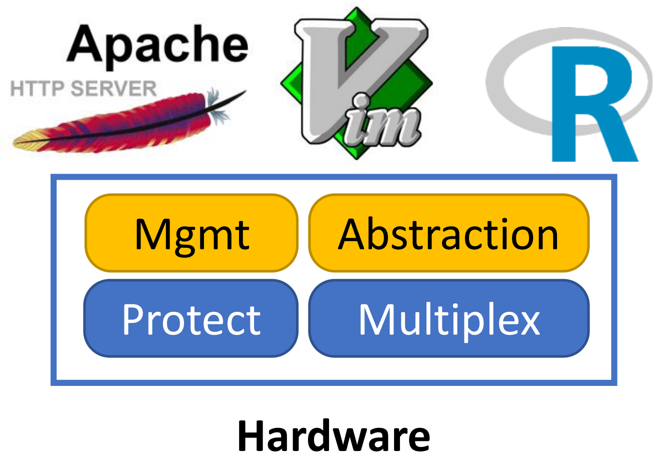
AI + big data: the workloads in our era

How does OS respond?

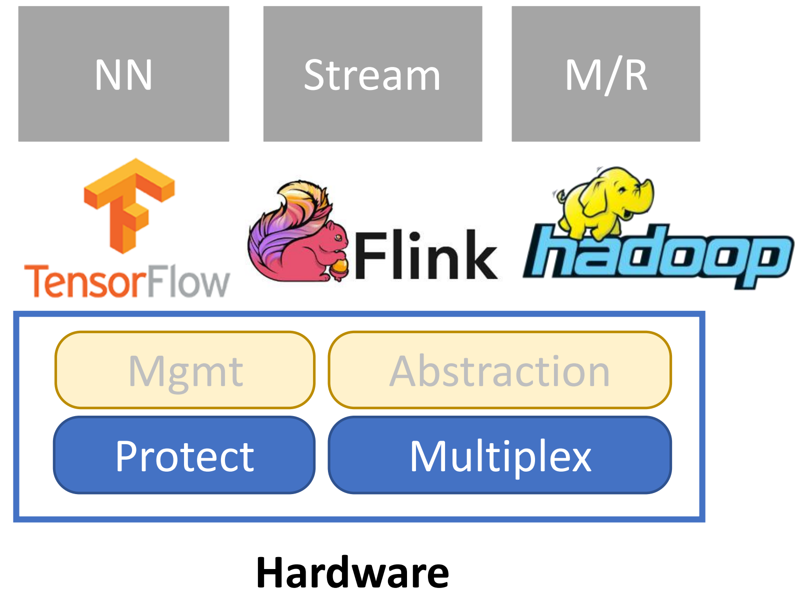
OS as in textbooks

- Resource management
- Abstraction
- Protection
- Multiplexing

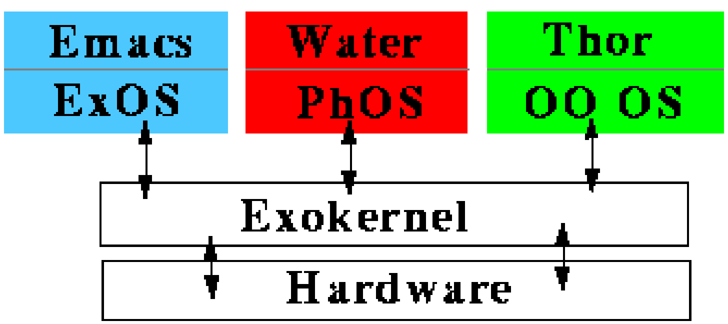




OS in 2000



OS in 2018



Exokernel: An Operating System Architecture for Application-Level Resource Management

Dawson R. Engler, M. Frans Kaashoek, and James O'Toole Jr.
M.I.T. Laboratory for Computer Science

SOSP 1995

Two premises of our research

1. Important apps define OSes

2. The Linux kernel is the new firmware

Like it or not …

- Following the model: widely adopted
    - Various ML frameworks
    - Android
    - DPDK
    - RDMA
- Against the mode: less adopted
    - Library OS
    - Multikernel
    - Unikernels
    - (… and all kinds of research proposals)

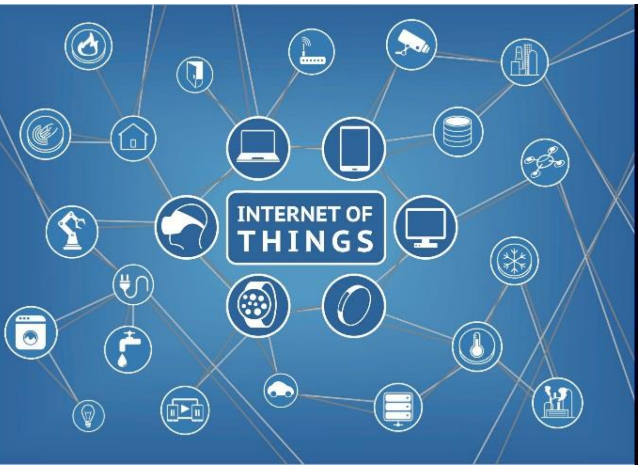
This talk: our exploration in three cases

- App-defined OSes
  - Case 1: Memory mgmt
  - Case 2: Storage
- The ossified Linux Kernel
  - Case 3: Transkernel

Case 1: App-defined memory management

Exploiting hybrid memory for stream analytics

Background: Stream analytics in 1 minute



IoT

Data centers

Humans

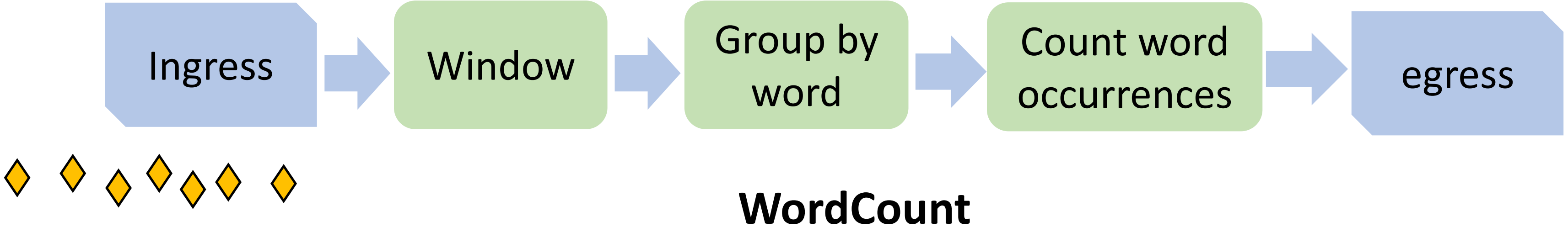
High input throughput: millions of events per sec

Low delay processing: sub-seconds

Background: streaming pipeline

Operators Computations consuming & producing streams

Pipeline A graph of operators

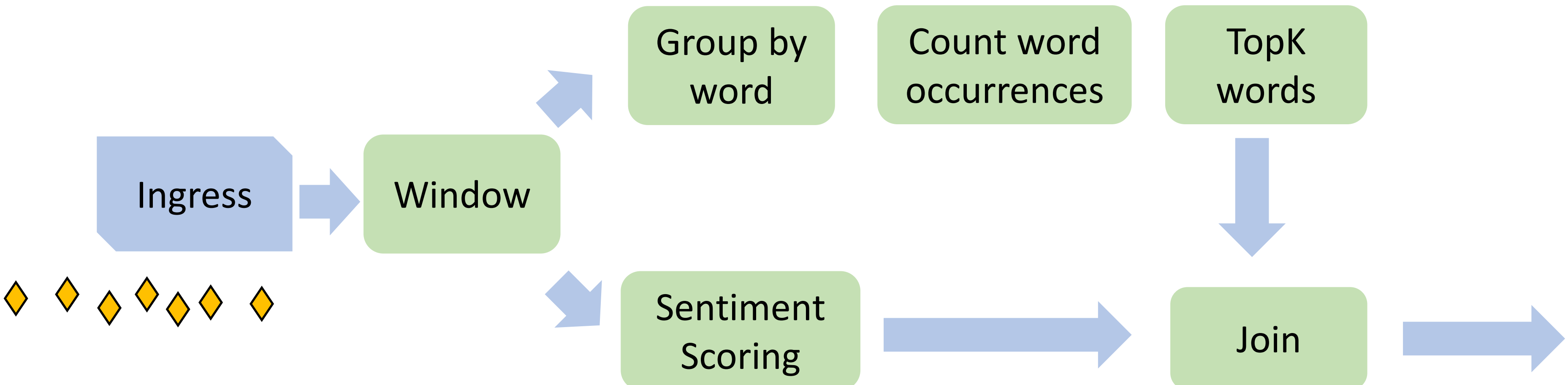


WordCount

Background: streaming pipeline

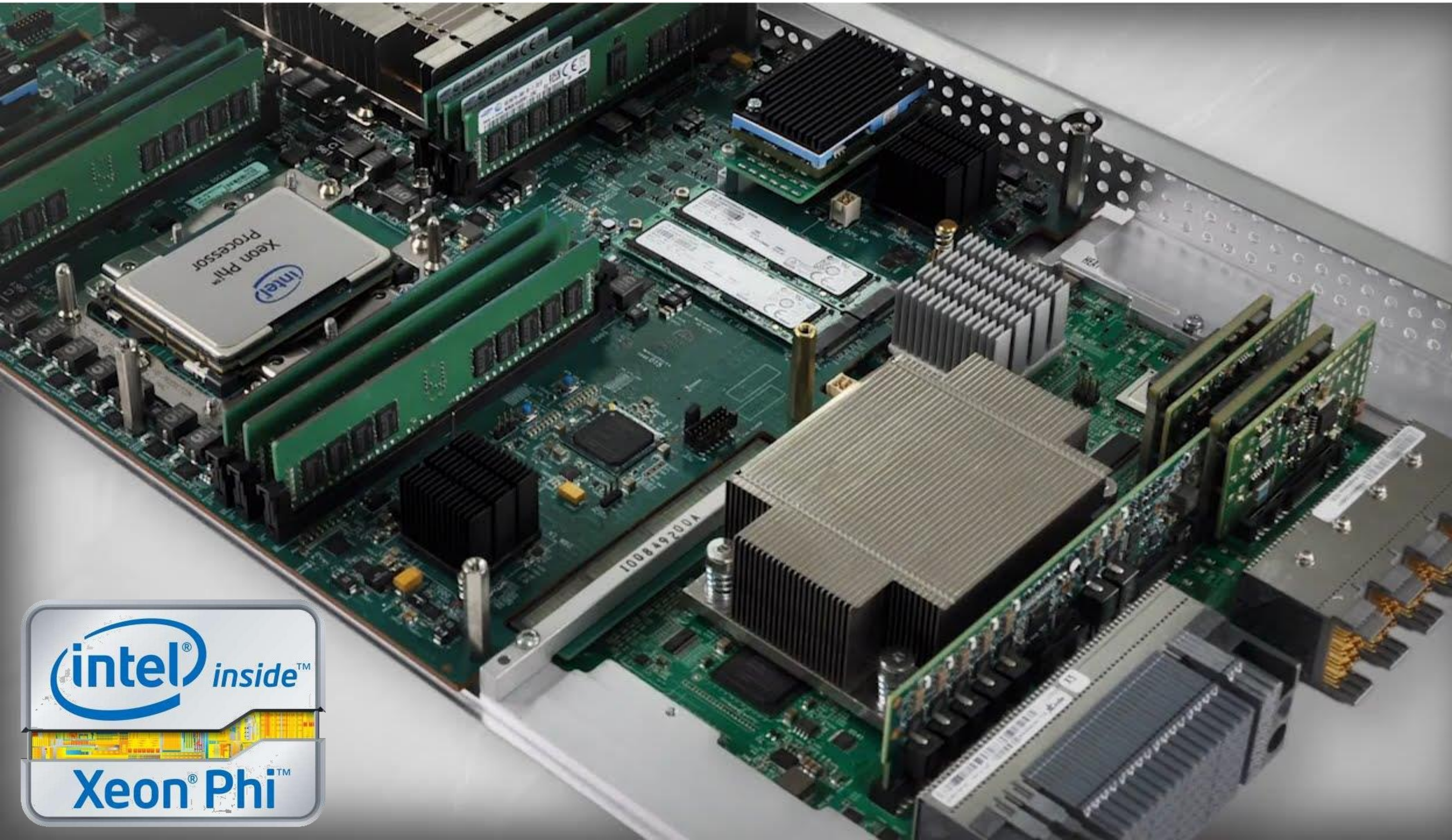
Operators Computations consuming & producing streams

Pipeline A graph of operators

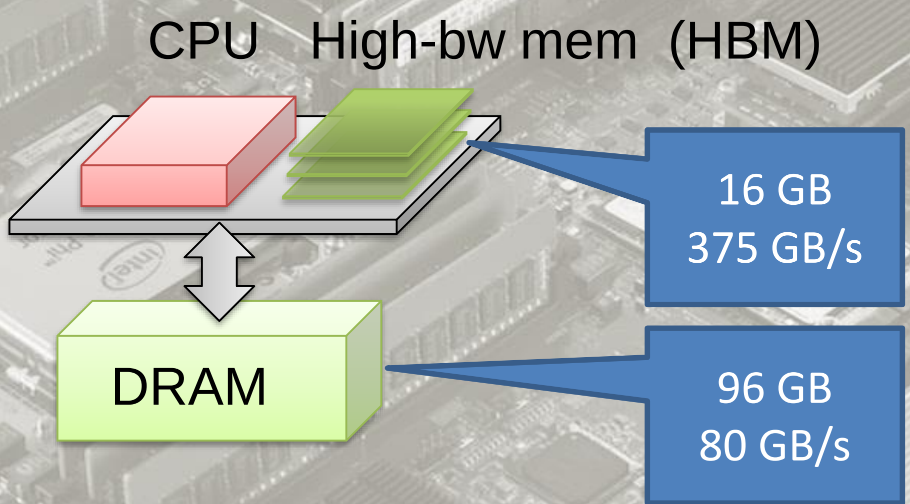


Sentiment Detection

Hybrid high-bandwidth memory



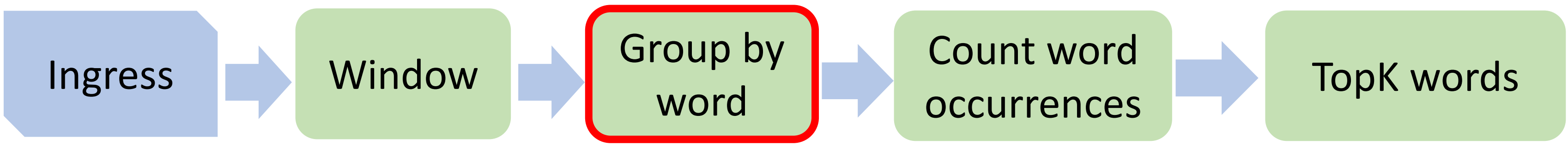
Hybrid high-bandwidth memory



- Tradeoffs: capacity vs bandwidth
- Untraditional memory hierarchy
  - No latency benefit (Unlike SRAM+DRAM)
  - Configurable: sw-managed or hw-managed

Accelerate stream analytics with HBM?

- The most expensive stream operators: **grouping**



Grouping traditionally built as Hash
Mismatch HBM!

Grouping in Hash	**High bandwidth memory**
High capacity demand	Capacity limited
Extensive random access	Seq access + parallelism

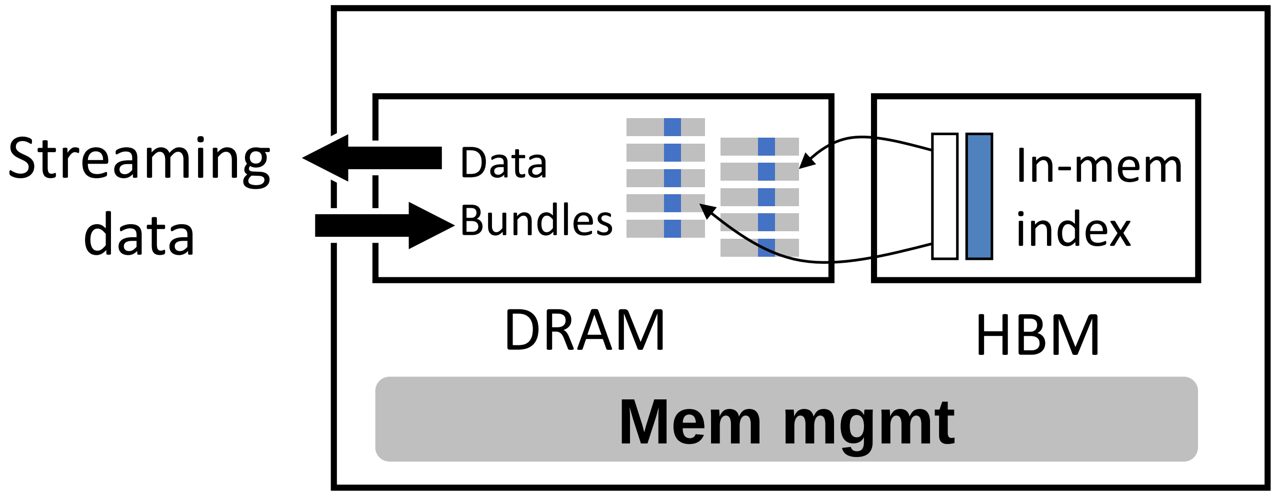
HBM can accelerate grouping!

- … with an unconventional algorithm choice
- Grouping: ~~hash~~ → sort

- Sort is **worse** than Hash on algorithmic complexity
  - O(NlogN) vs. O(N)

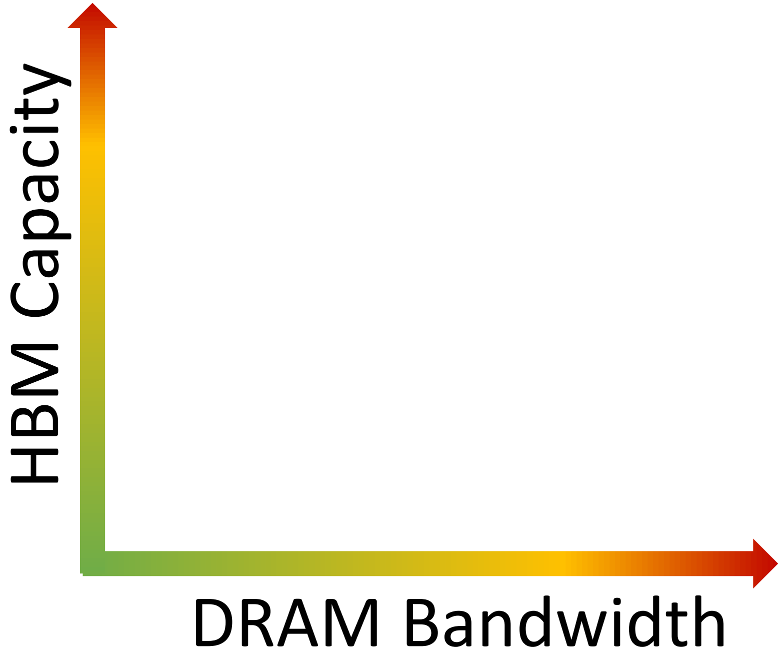
- Yet, Sort **beats** Hash with …
  - Abundant mem bw
  - High task parallelism
  - Wide SIMD (avx512)

A bold design: only use HBM for in-mem index of parallel sorting



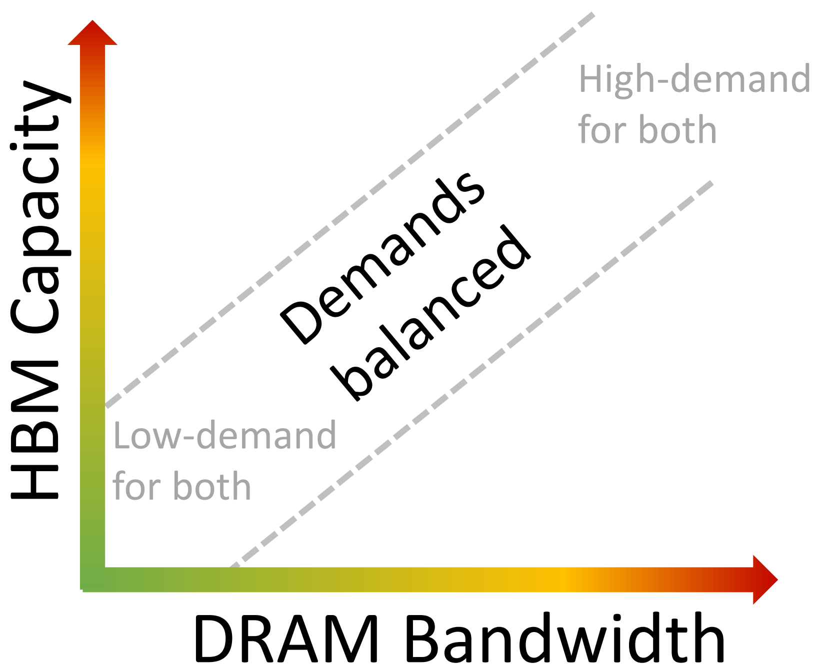
Memory allocator: balancing two limited resources

- Prevent either from becoming the bottleneck
- A single knob: where to allocate new index: HBM or DRAM?



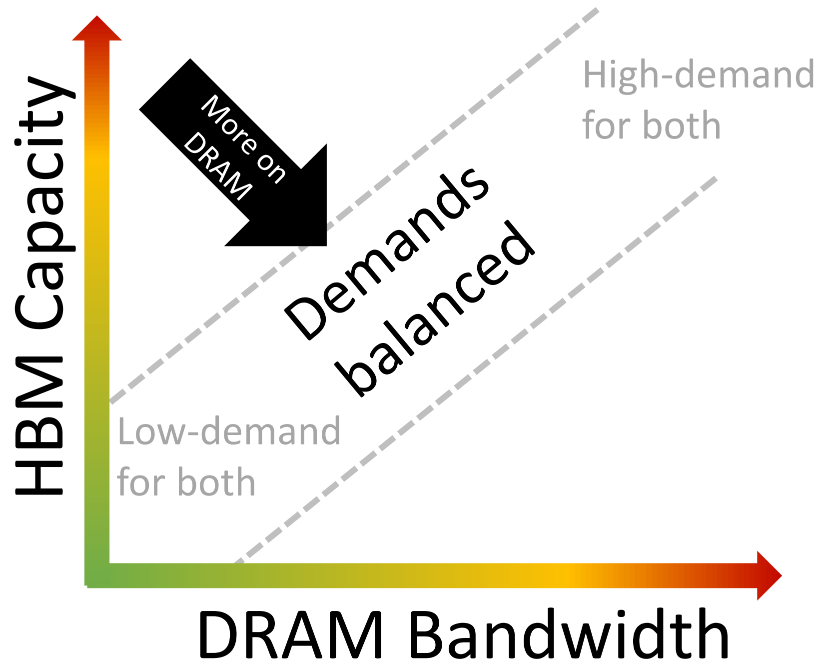
Memory allocator: balancing two limited resources

- Prevent either from becoming the bottleneck
- A single knob: where to allocate new index: HBM or DRAM?



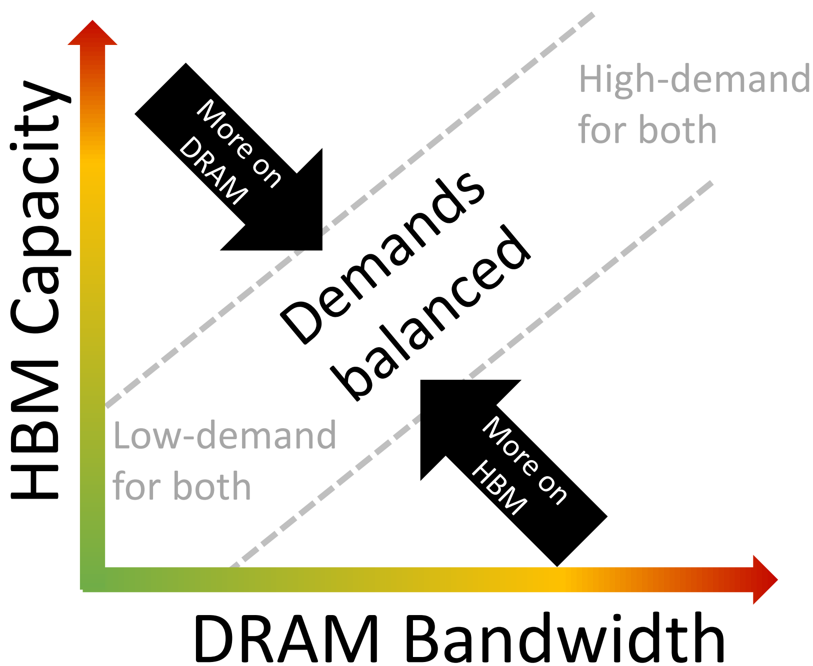
Memory allocator: balancing two limited resources

- Prevent either from becoming the bottleneck
- A single knob: where to allocate new index: HBM or DRAM?



Memory allocator: balancing two limited resources

- Prevent either from becoming the bottleneck
- A single knob: where to allocate new index: HBM or DRAM?



Case 1: First streaming engine optimized for hybrid memory

- Works on real hardware
- Best stream analytics performance on a single machine
- Generic memory allocators are inadequate!

This talk: our exploration in three cases

- App-defined OSes
  - Case 1: Memory mgmt
  - Case 2: Storage
- The ossified Linux Kernel
  - Case 3: Transkernel

Case 2: App-defined storage

A data store for video analytics


/Shutterstock

Pervasive cameras. Large videos.

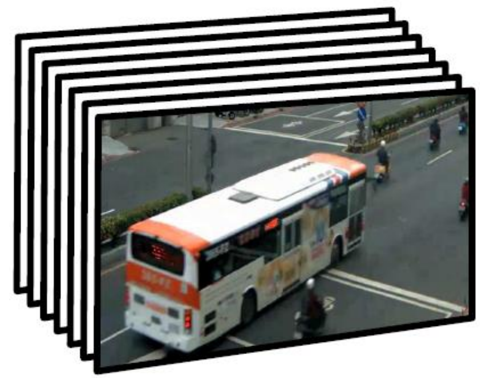
- 130M surveillance cameras shipped per year
- Many institutions run > 200 cameras 24x7
- A single camera produces 24 GB video per day

Retrospective video analytics in 30 seconds

- A query: “find all white buses appeared yesterday”
- A cascade of operators

Frame diff detector | Specialized neural net | Full neural net

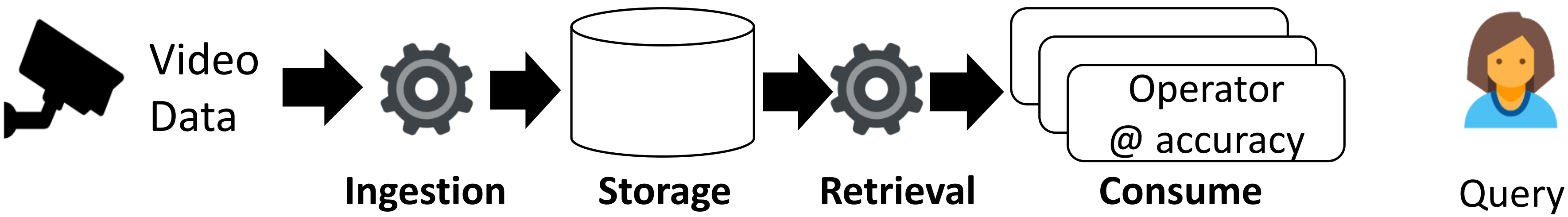
~10,000x | **~1,000x** | **~10x**



- Query picks operator accuracies
  - Lower accuracy → lower cost → faster execution

Image credits: NoScope: Optimizing Neural Network Queries over Video at Scale, Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, Matei Zaharia, VLDB 2017

Need for a video store for retrospective analytics



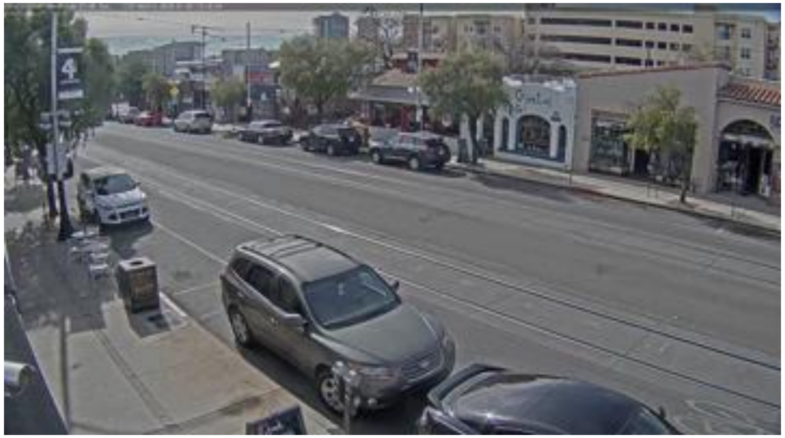
“Aren’t there many video databases already?”
Yes.
But they are for **human** consumers.
Not for **algorithmic** consumers.

Central design issue: choose storage formats catering to operators

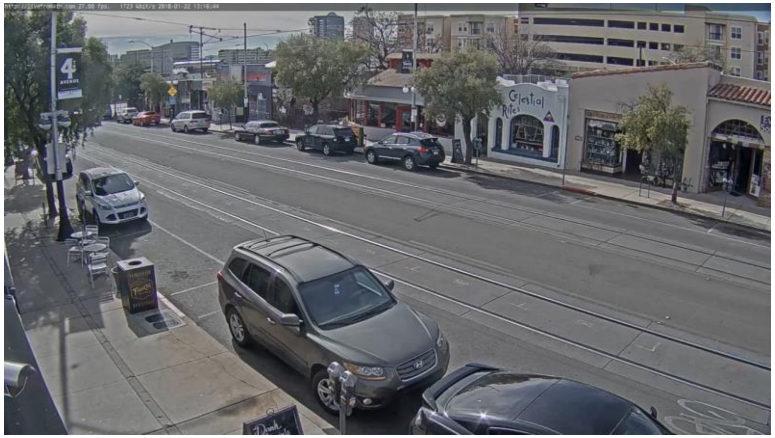
- Video format: resolution, frame rate, crop, compression level, color channel… **Many knobs!**

Central design issue: choose storage formats catering to operators

- Video format: resolution, frame rate, crop, compression level, color channel… **Many knobs!**
- Higher accuracy → richer format & higher cost



200p 1FPS
(~100KB/s)



720p 30FPS
(~400KB/s)

Actual tuning of formats is complex!

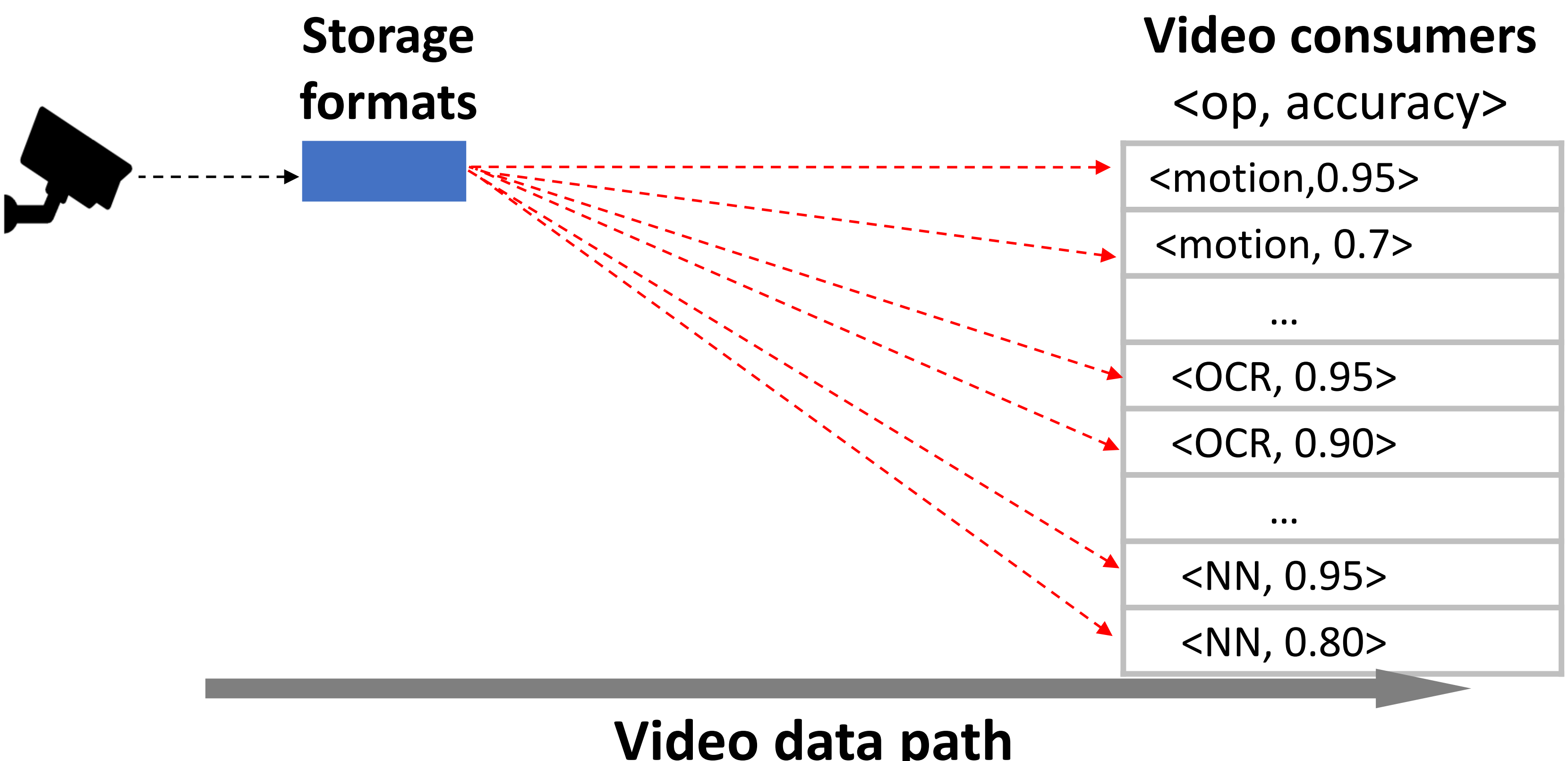
Key problem: What video formats to store?

1. Retrieving formats must be fast enough to feed operators

2. Formats must be rich enough to satisfy all accuracy needs

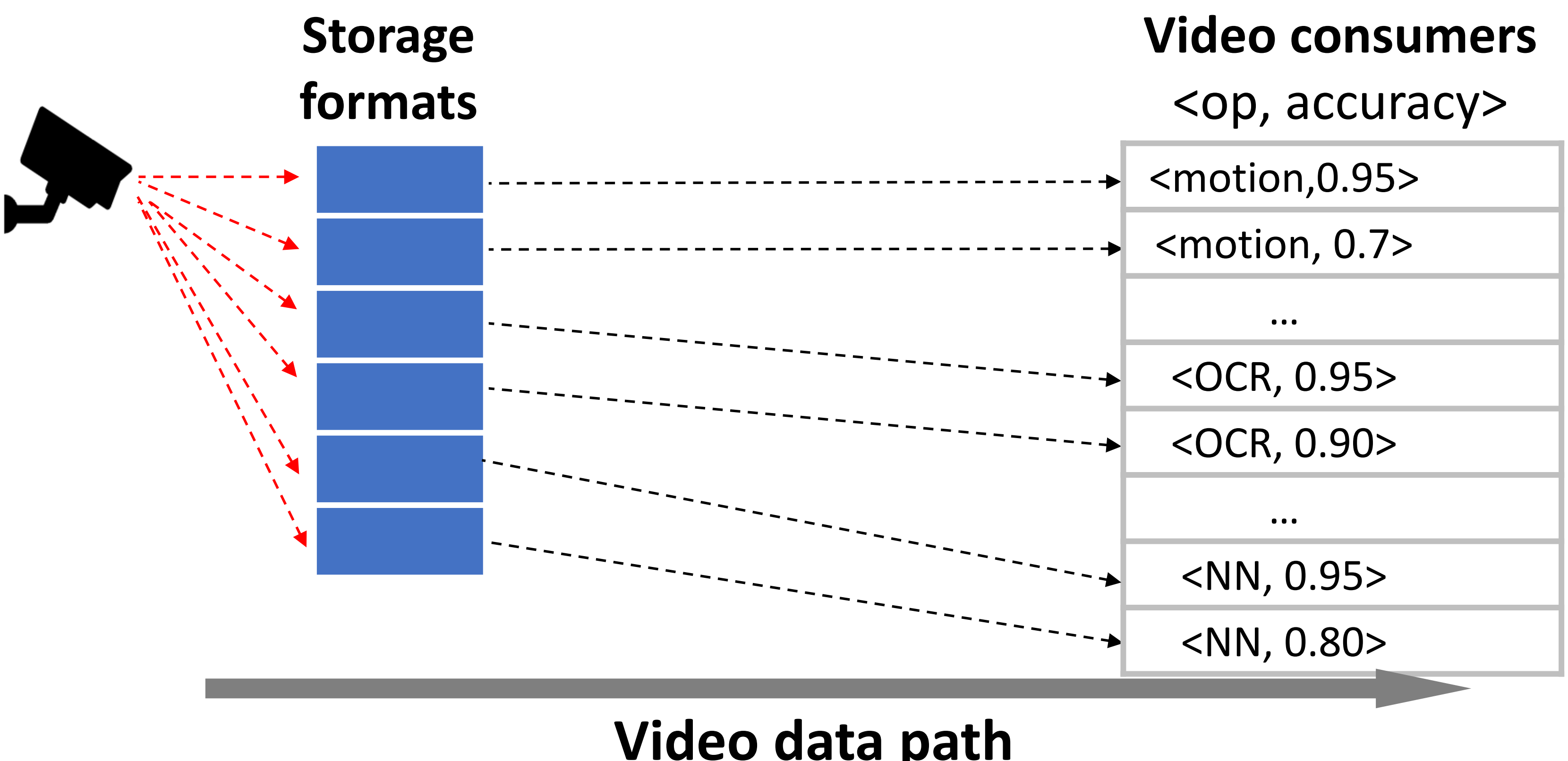
Storing one unified format?

Retrieving & decoding slows down operators

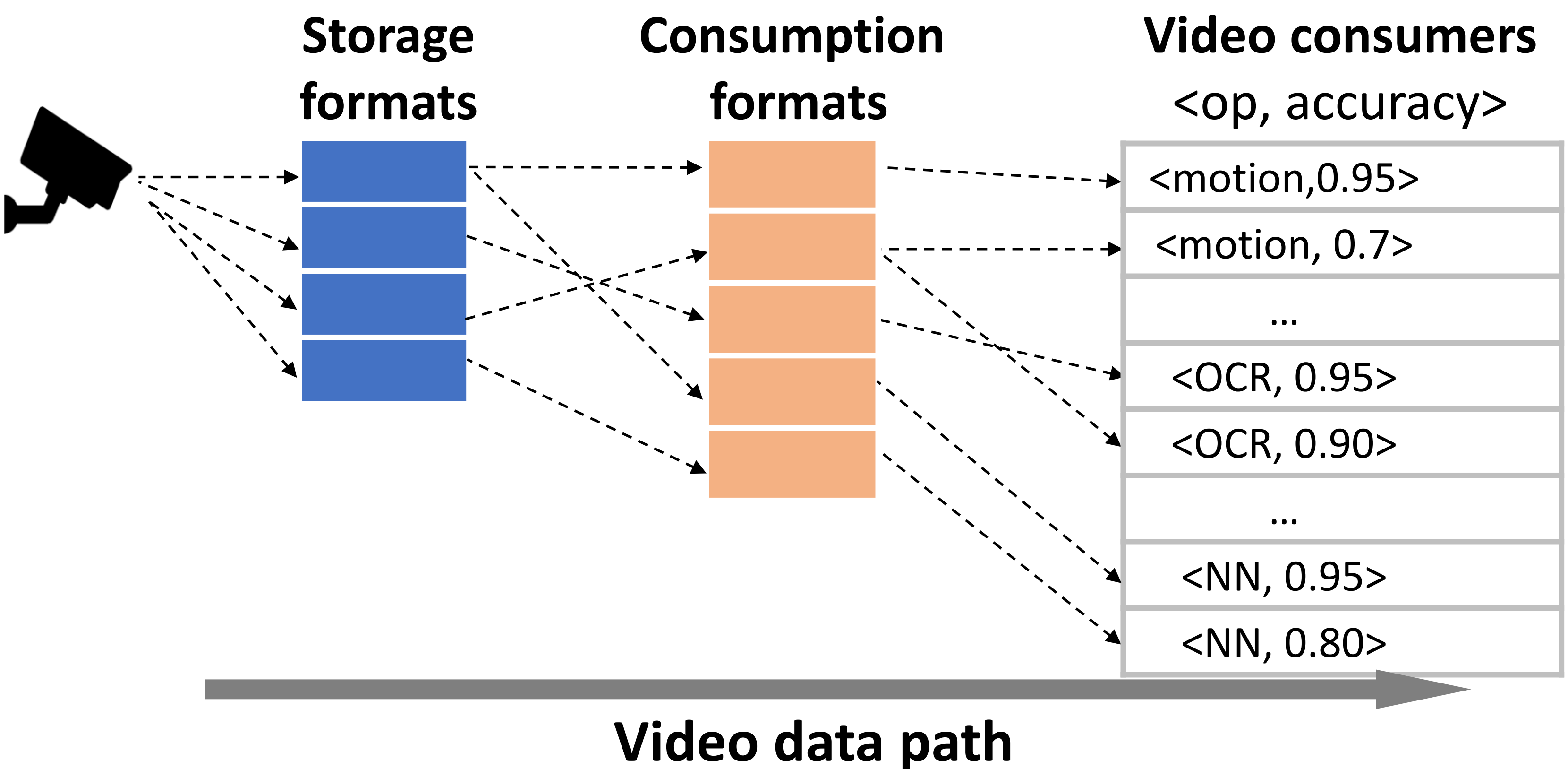


Storing all needed formats?

Ingestion & storage are expensive

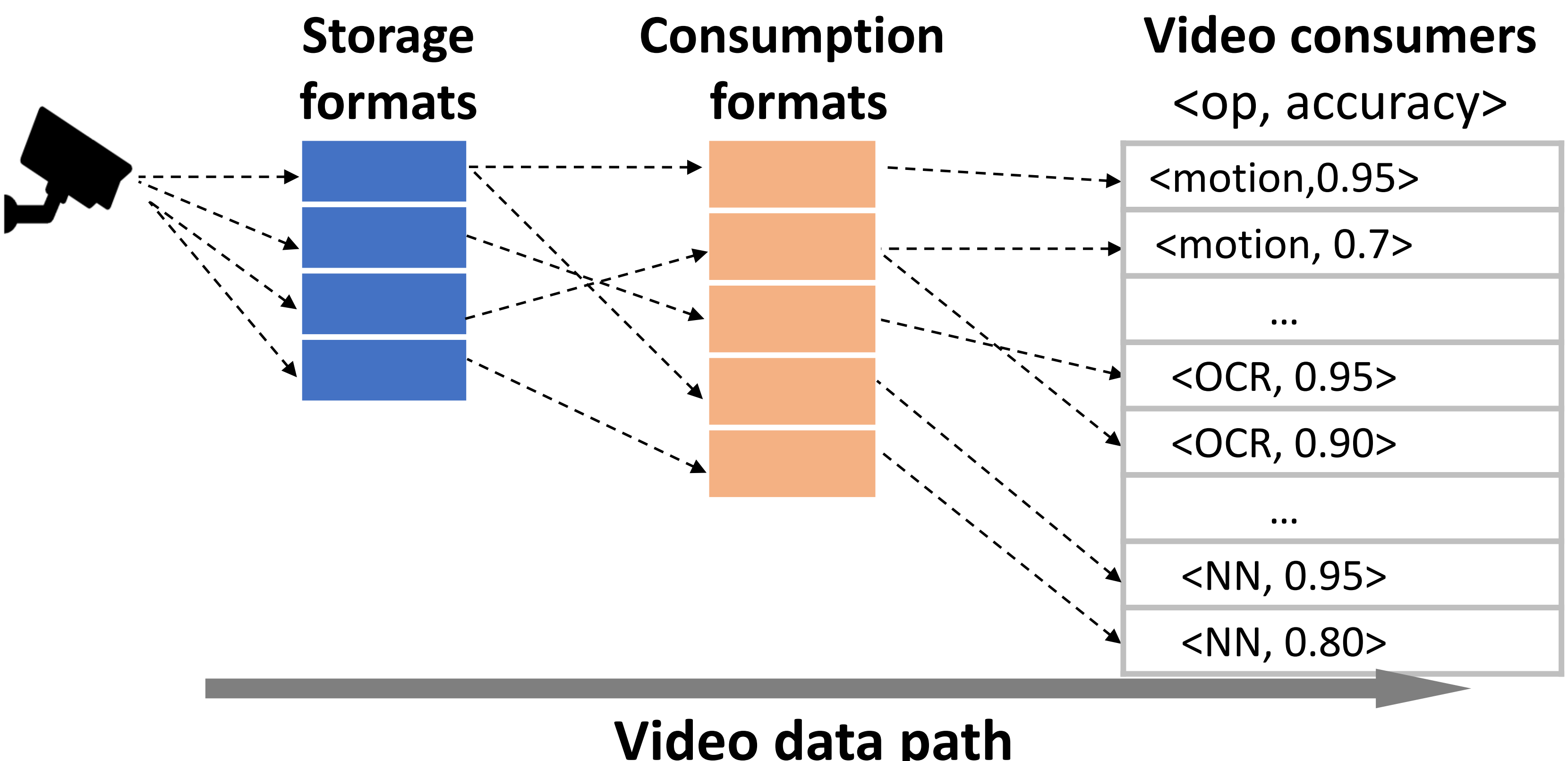


Proposal: Store minimum necessary formats

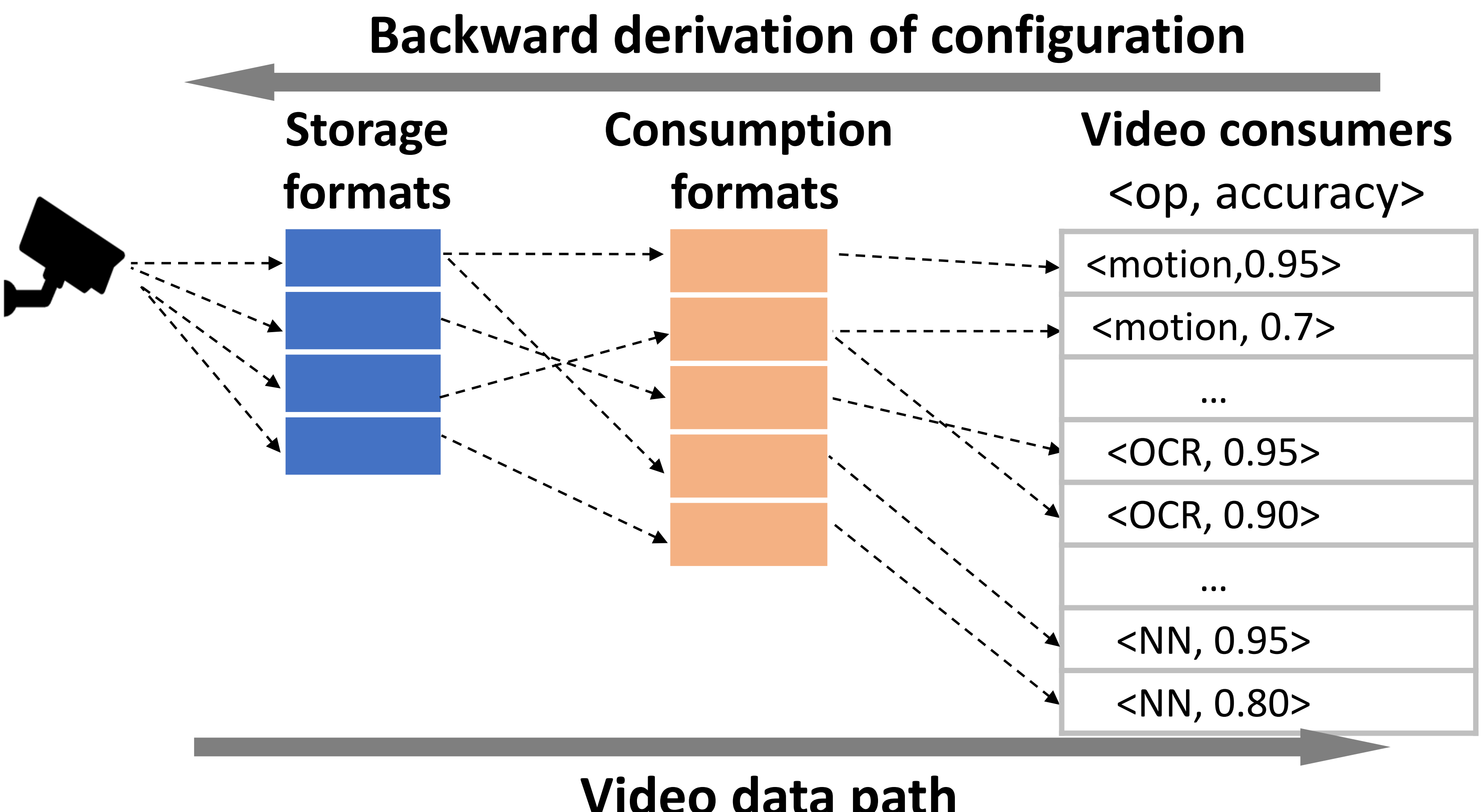


Proposal: Store minimum necessary formats

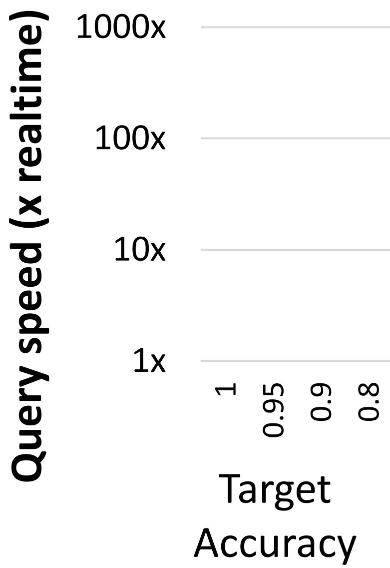
Key Challenge: Too many knobs to tune!



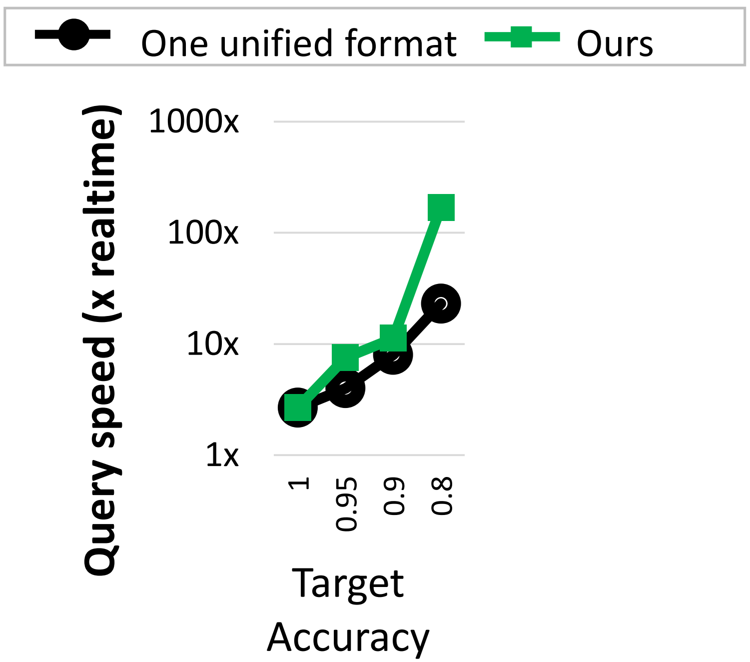
Proposal: Store minimum necessary formats & derive them automatically!



Benefit: faster analytics



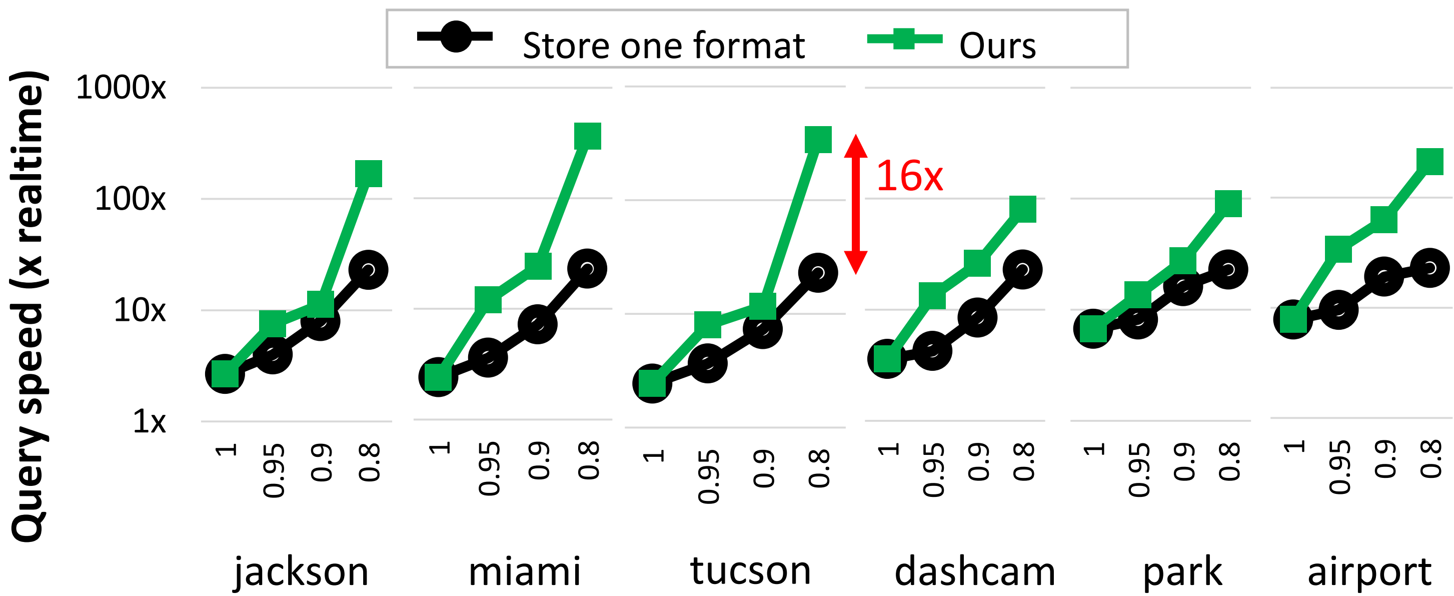
Benefit: faster analytics



Why?

- Lower accuracy → analytics tolerates cheaper video formats
- Video retrieval should be proportionally cheaper!

Benefit: faster analytics



Why?

- **Lower** accuracy → analytics tolerates **cheaper** video formats
- Video retrieval should be proportionally cheaper!

Case 2: First data store for retro. video analytics

- Baking analytics demands into storage decisions

- Showing clear advantage over generic file systems or storage layers

This talk: our exploration in three cases

- App-defined OSes
  - Case 1: Memory mgmt
  - Case 2: Storage
- The ossified Linux Kernel
  - Case 3: Transkernel

Case 3: enhancing the ossified Linux kernel

A new OS structure for device driver offloading

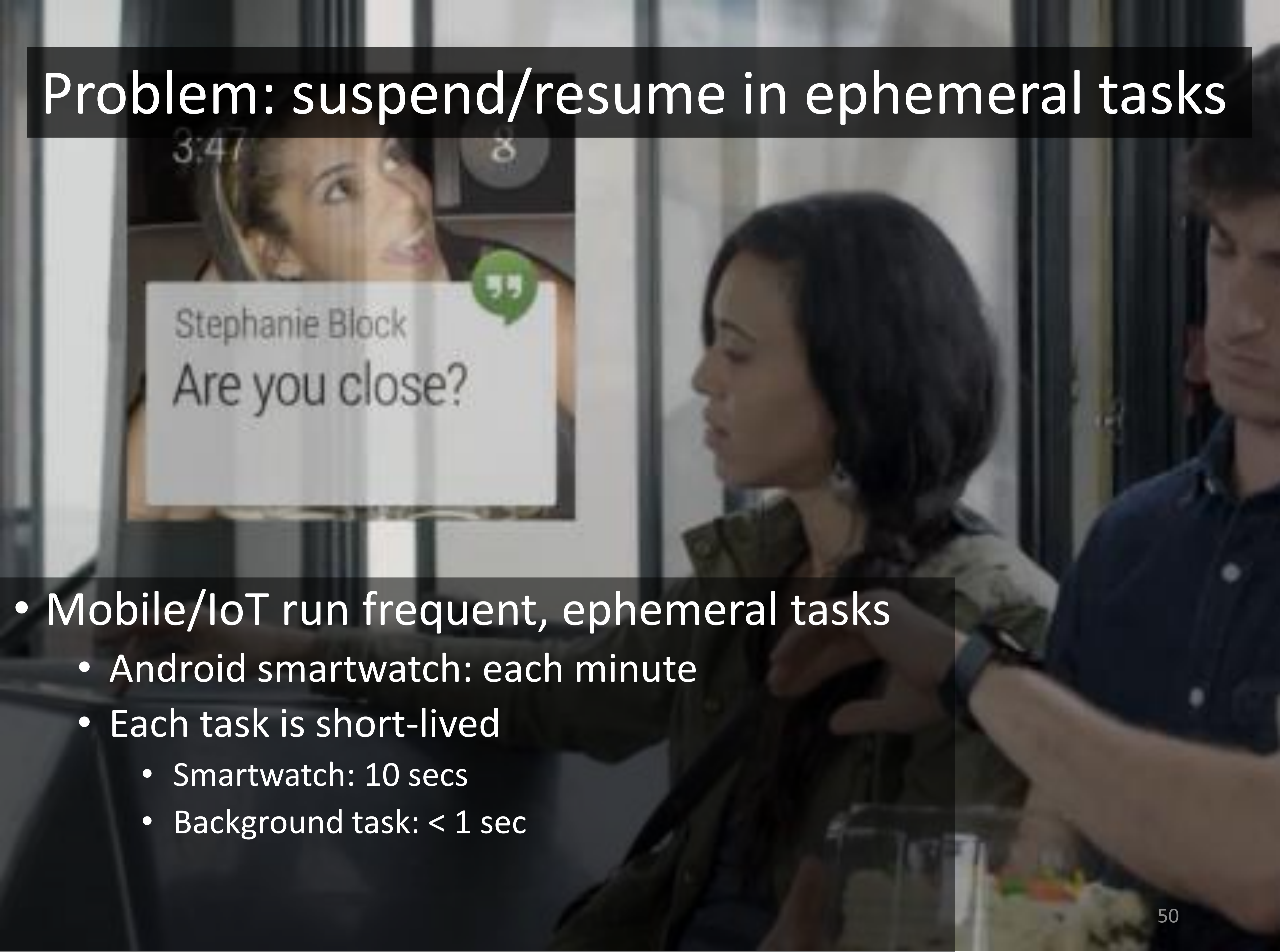
A commodity kernel still irreplaceable

- The major reason: **device drivers**
  - Linux: a common environment for all drivers
  - >70% Linux kernel source are device drivers

- All other OS functions have alternatives!

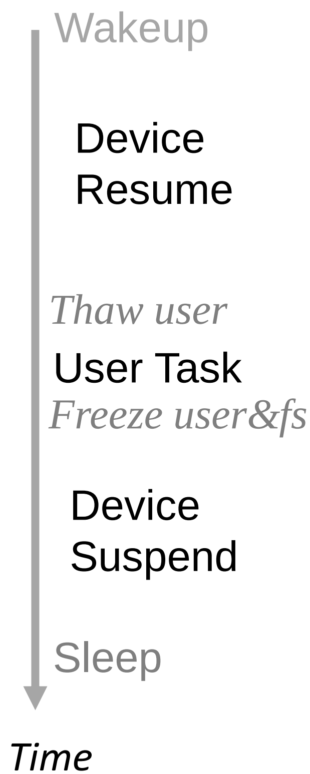
1. Understanding Modern Device Drivers, Asim Kadav and Michael Swift. ASPLOS'12.

Problem: suspend/resume in ephemeral tasks



- Mobile/IoT run frequent, ephemeral tasks
  - Android smartwatch: each minute
  - Each task is short-lived
    - Smartwatch: 10 secs
    - Background task: < 1 sec

Under the hood…

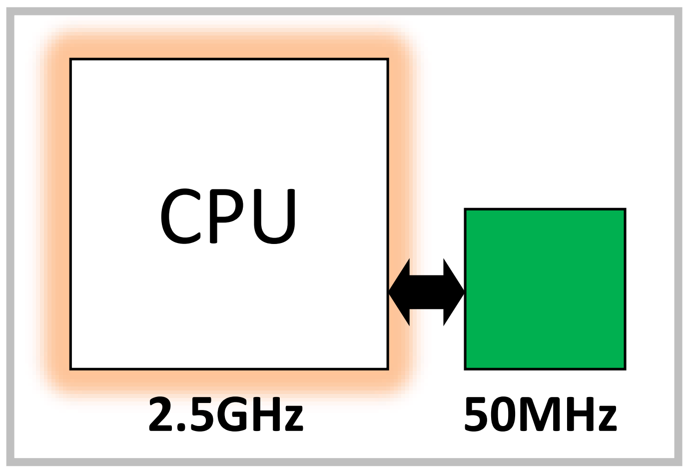


Suspend/Resume Is Slow

	Nexus 5	Gear	Note 4	Panda
Suspend	119 ms	191 ms	231 ms	262 ms
Resume	88 ms	159 ms	316 ms	492 ms

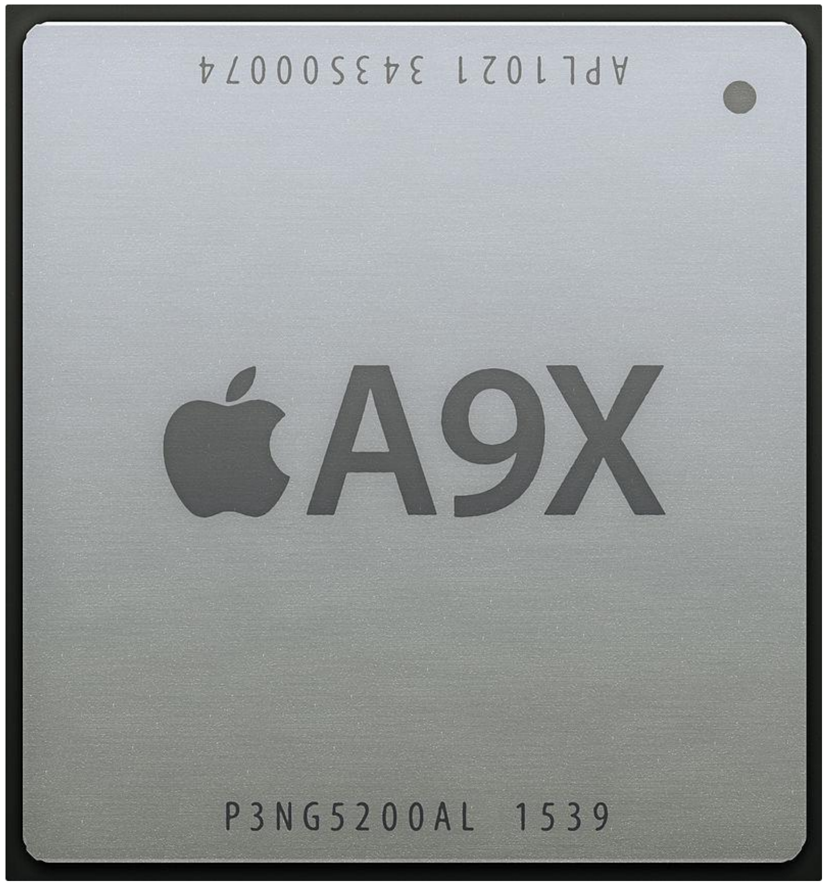
Why? IO devices are slow

Proposal: run these driver paths on a low-power tiny core

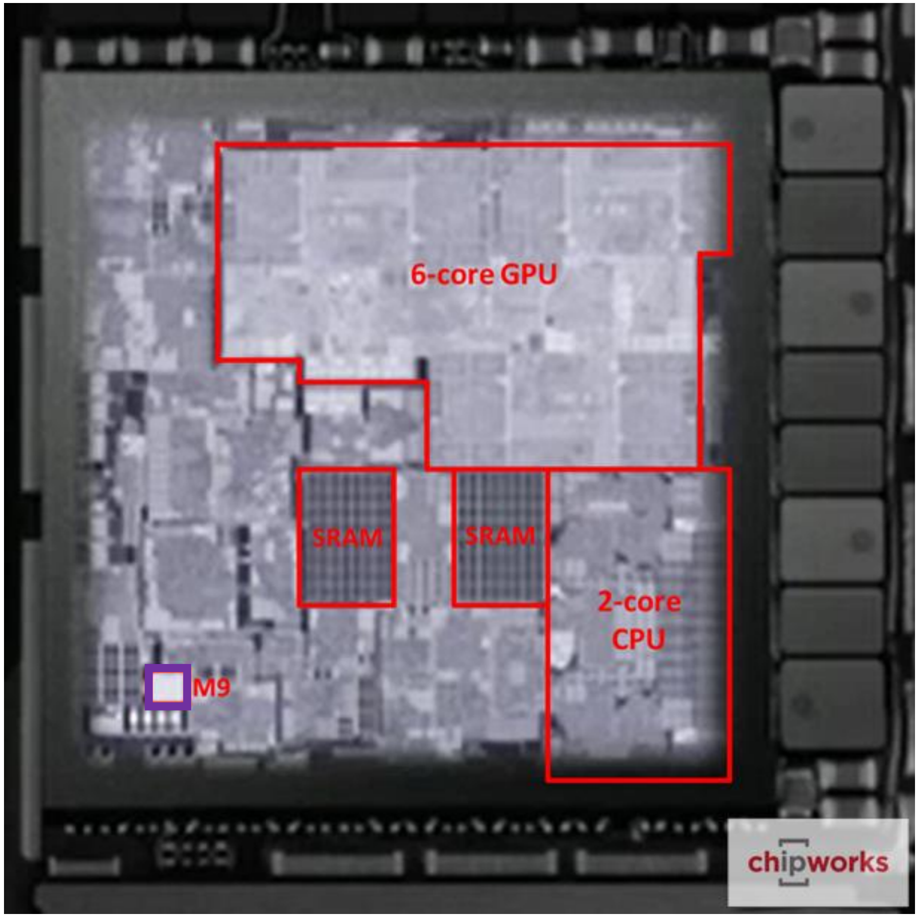


What are these tiny cores?

What are these tiny cores?



What are these tiny cores?

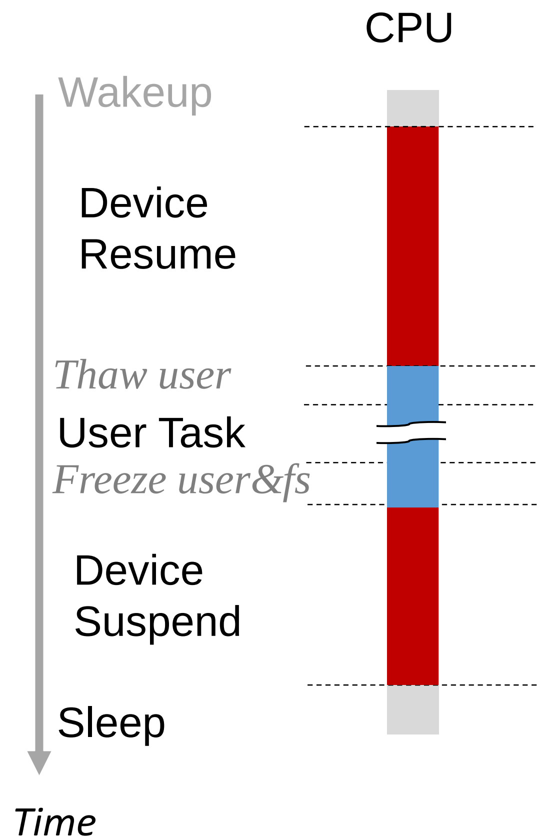


How can a tiny core help?

- Idling more efficiently
  - 3 mW vs 30 mW
- Executing kernel more efficiently
  - Small code working set
  - Less predictable control flow

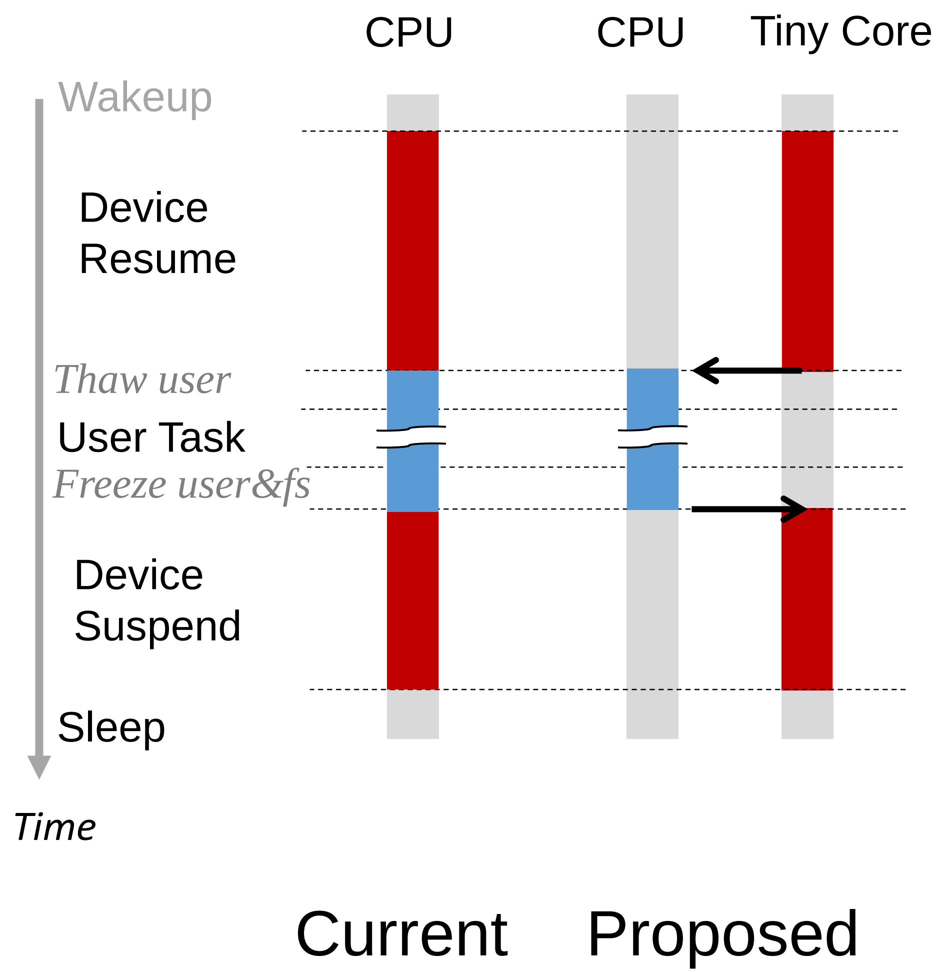
1. J. Mogul, J. Mudigonda, N. Binkert, P. Ranganathan, and V. Talwar. Using asymmetric single-isa cmps to save energy on operating systems. Micro, IEEE, 2008

The workflows



Current

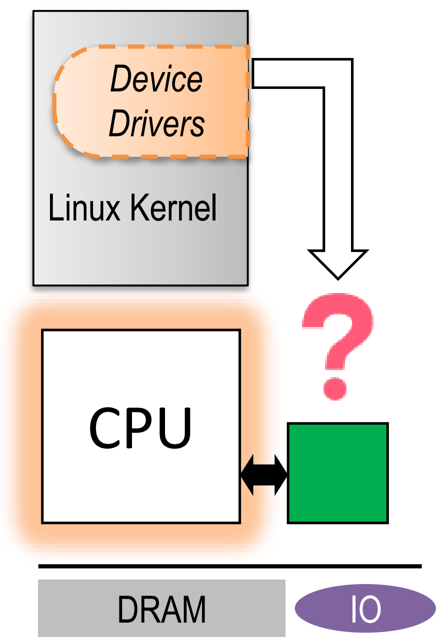
The workflows



How?

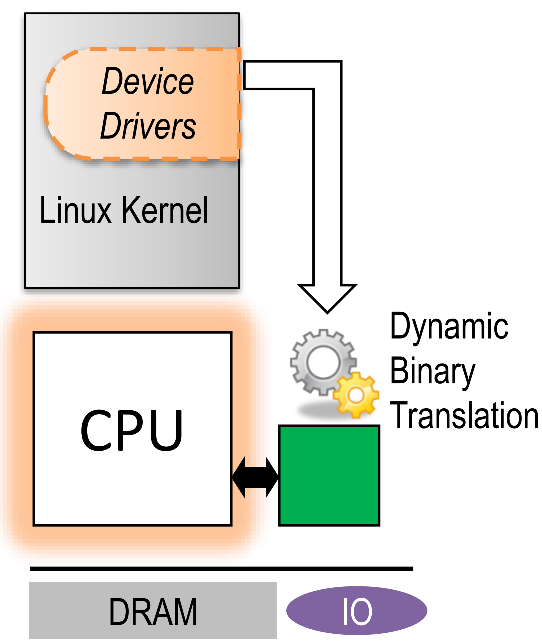
- The tiny core is very wimpy
  - Essentially a microcontroller
- Has a different ISA
  - More aggressive than big.LITTLE

- Re-engineering the kernel?
- “Linux kernel is the new firmware”



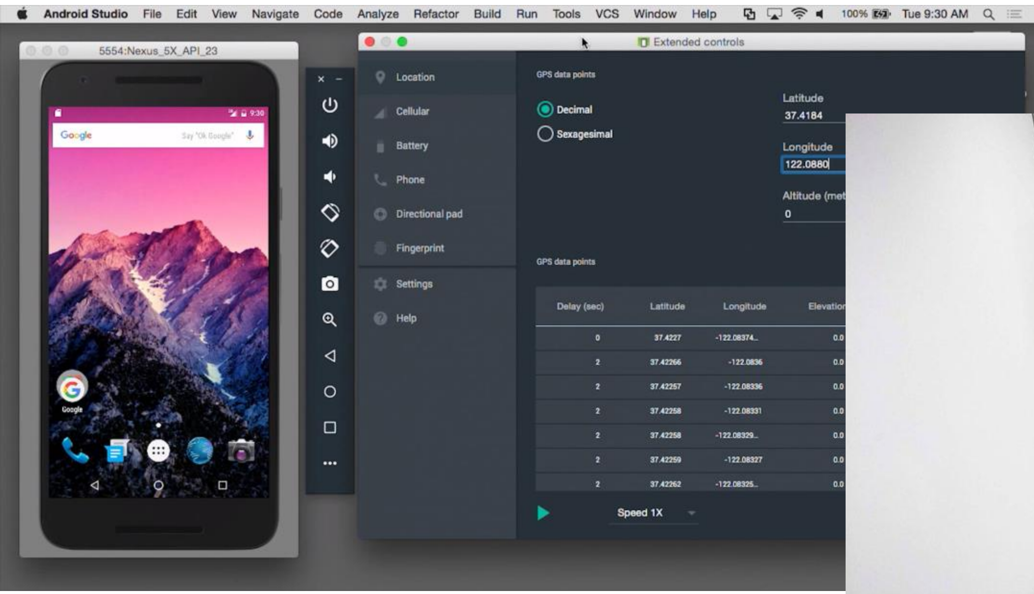
Proposal: use dynamic binary translation

- Empower the tiny core to execute ***unmodified*** kernel binaries



Dynamic binary translation in 30 sec

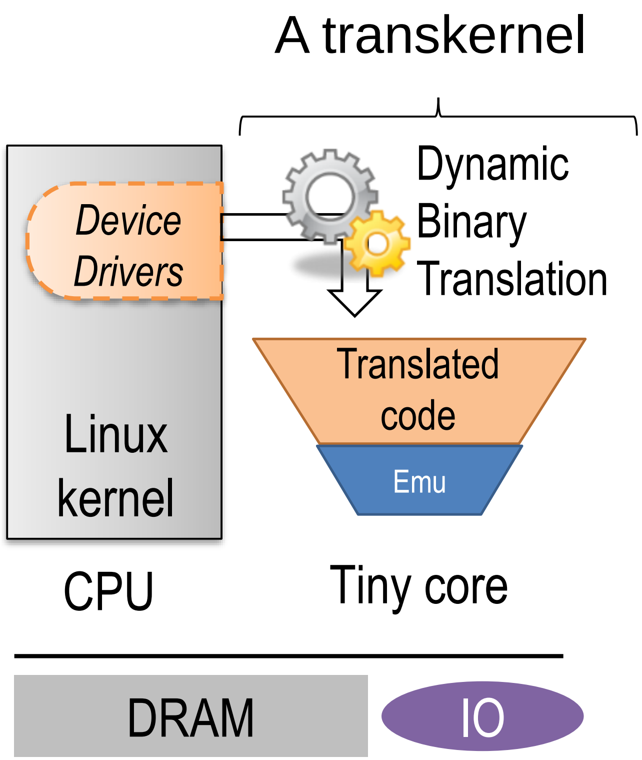
- The technology behind QEMU and a PS emulator



“This sounds impractical”

- Dynamic bin translation (DBT) is known expensive
  - > 10x overhead
- No one runs DBT on microcontrollers
  - Always weak over strong

Solution: the Transkernel model

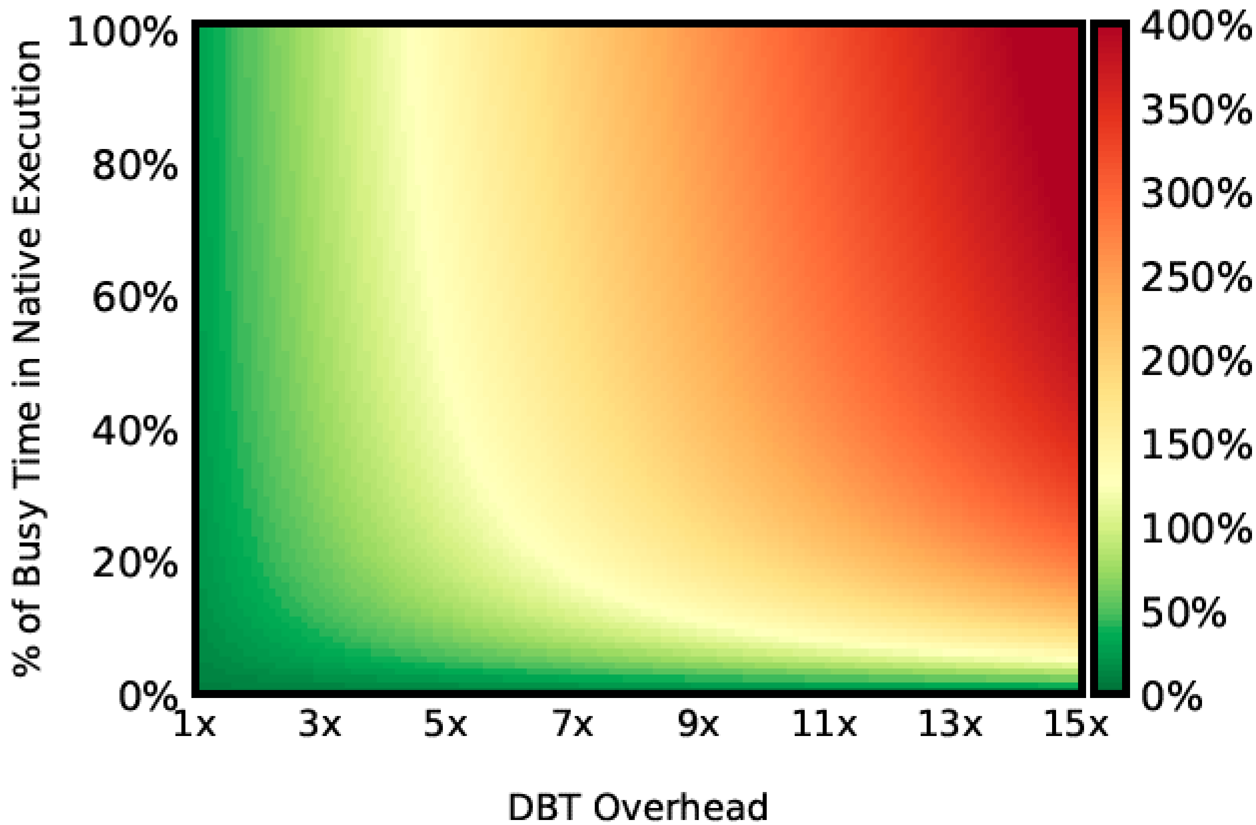


Key ideas:

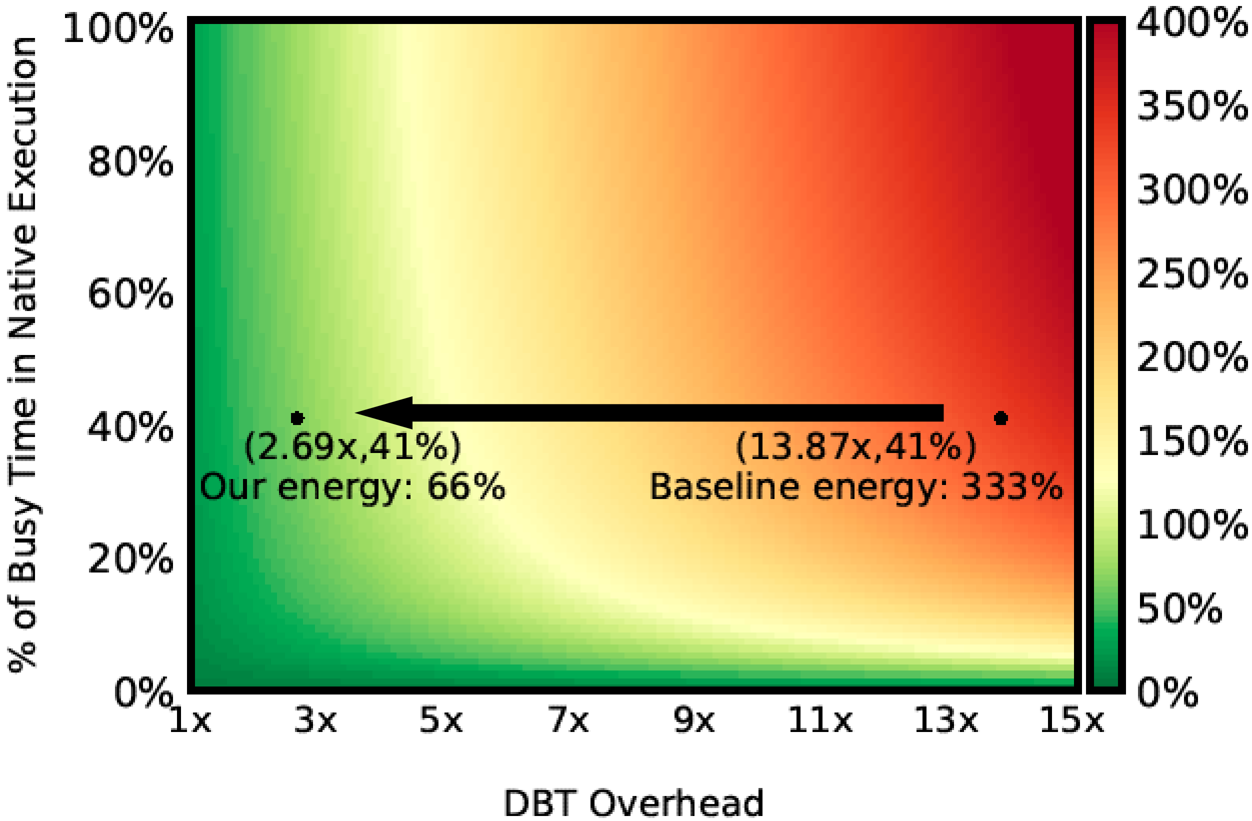
1. Translate most of driver code
2. Emulate the remaining kernel infrastructure

Essentially a tiny VM for drivers!

Is this practical?



Only through careful optimization!



Case 3: Transkernel: tiny VMs for device driver paths

First DBT engine running on a microcontroller-like core!

Demonstrated:

- One can use DBT for efficiency gain
- The ossified kernel can be tamed!

This talk: our exploration in three cases

- App-defined OSes
  - Case 1: Memory mgmt for stream analytics
  - Case 2: Storage for retro. video analytics
- The ossified Linux Kernel
  - Case 3: Transkernel

Reflection: a tale of two inquiries

- App-defined OS
  - Brave new world
  - Usefulness depends on the target apps
  - OS as a tool: serving AI and big data
- Kernel as the new firmware
  - Still lots of opportunities
  - Unique constraints. Require unconventional techniques
  - Like fixing an engine of an airplane in the air
  - OS as a craft. Fun!