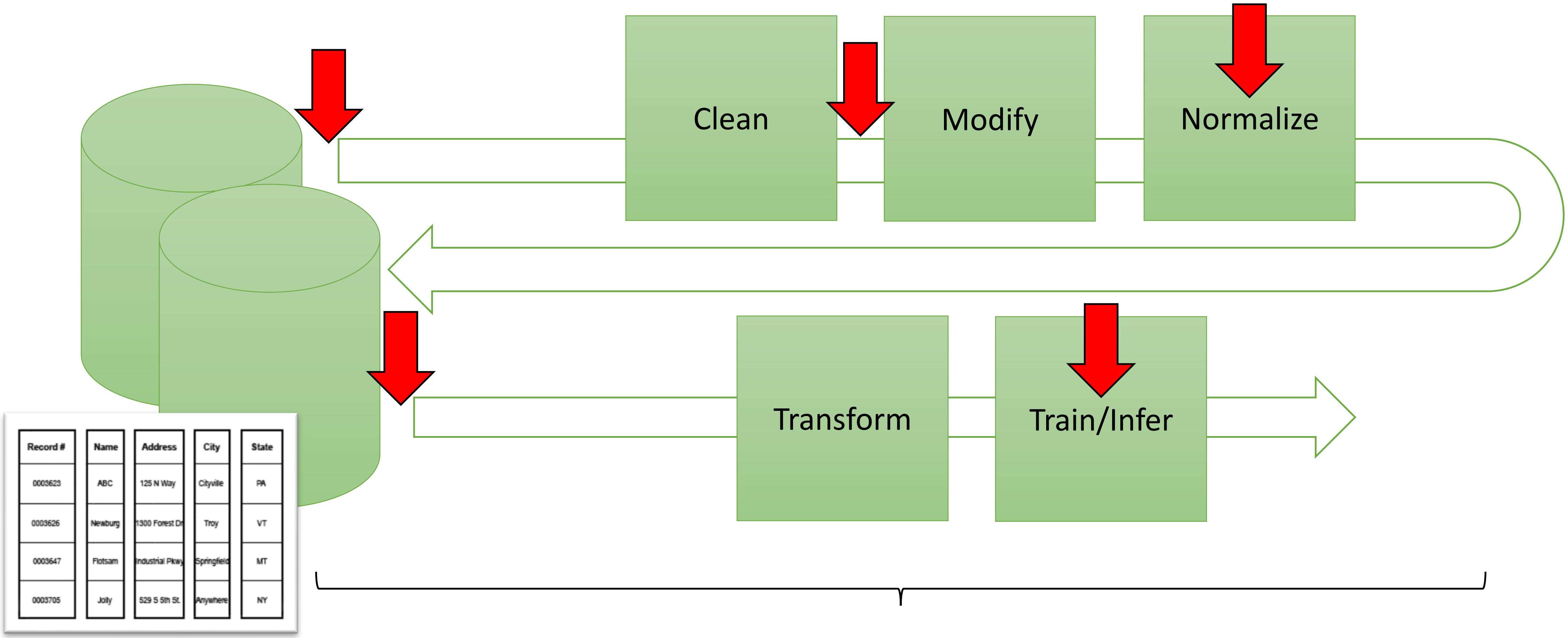
Specialize in Moderation

Building Application-aware Storage Services using FPGAs in the Datacenter

Lucas Kuhring, Eva Garcia, Zsolt István

IMDEA Software Institute, Madrid

ML Pipelines in the Cloud



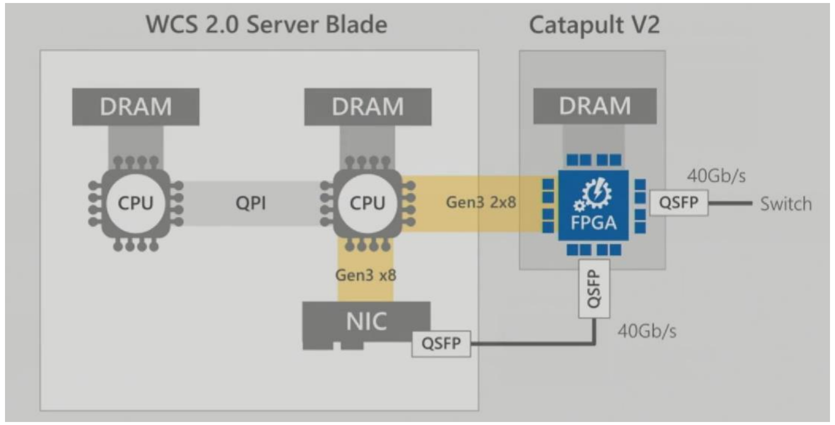
Record #	Name	Address	City	State
0003623	ABC	125 N Way	Cityville	PA
0003626	Newburg	1300 Forest Dr	Troy	VT
0003647	Flotsam	Industrial Pkwy	Springfield	MT
0003705	Jolly	529 S 5th St.	Anywhere	NY

Column-oriented file formats (e.g. Parquet)

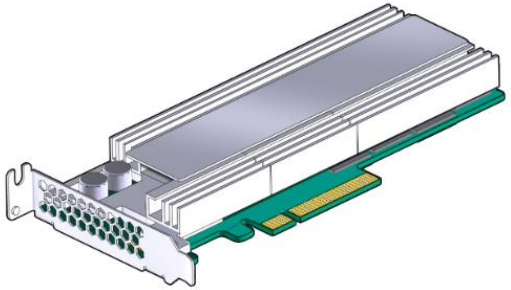
ML Framework/Service

Emerging Programmable Hardware

- Networking
  - Programmable switches
  - SmartNICs
  - Microsoft Catapult



- Compute accelerators
  - FPGA
  - TPU
  - Intel Xeon+FPGA

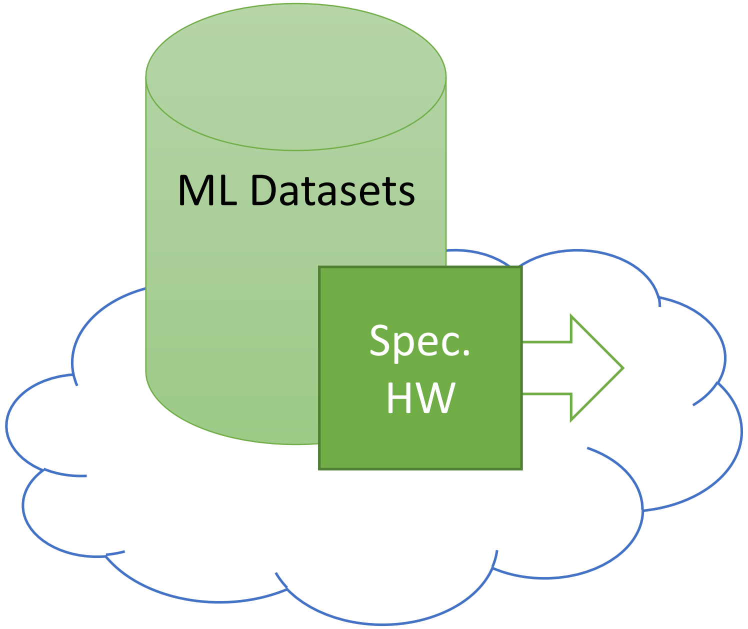


Let’s build Specialized Nodes…

- KVS on FPGAs well studied in related work
  - Several pipelined designs, saturate network link (e.g., Caribou, KV-Direct, BlueCache, …)
  - Can provide replication for fault-tolerance (Consensus in a box)

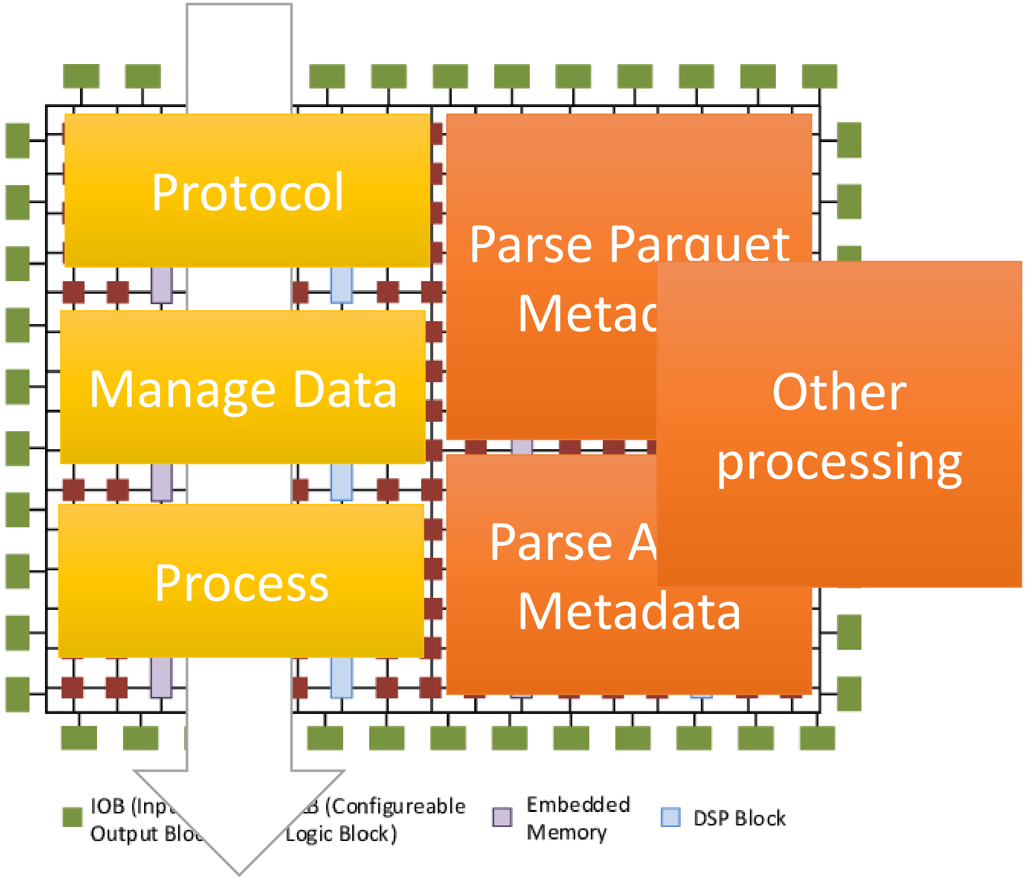
- “ML Store” microserver
  - ➢Low latency and high throughput access to data
  - ➢Low energy and small physical footprint
  - ➢Near data processing to filter/transform data
  - ➢Predictable behavior even with processing

- But… Needs to be shared! Need to support other file types/apps! Software evolves!



Code & Programmable Hardware

- Hardware is different from Software: <u>code</u> is converted to <u>circuits</u>
  - FPGAs synthesize logic gates
  - P4 switches have bounded pipeline, …
- Sharing is difficult if tenants require different functionality
  - Even parsing can be expensive
  - Can lead to reduced usefulness for all!



Parquet-aware KVS on FPGA

Multes++

- Efficient multi-tenant use of KVS logic – data & performance isolation [FPL18]
- Add deduplication in hardware – seamless processing in-storage
- Add software library to parse/manage Parquet files – easy to evolve

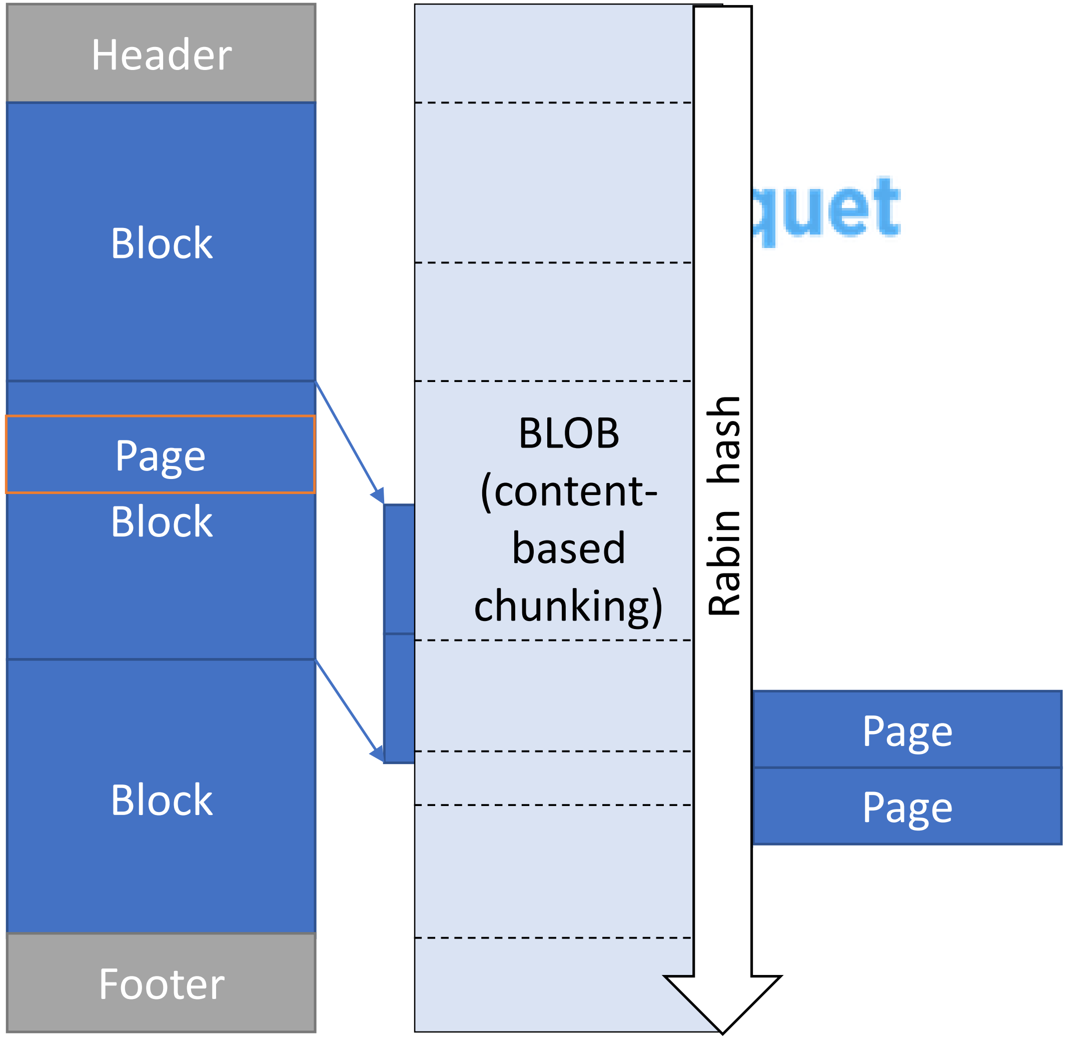
✓Benefit from app-knowledge, while storage node remains general purpose

[FPL18] Z. Istvan, G. Alonso, A. Singla: Providing Multi-tenant Services with FPGAs: Case Study on a Key-Value Store. FPL'18

Parquet Files

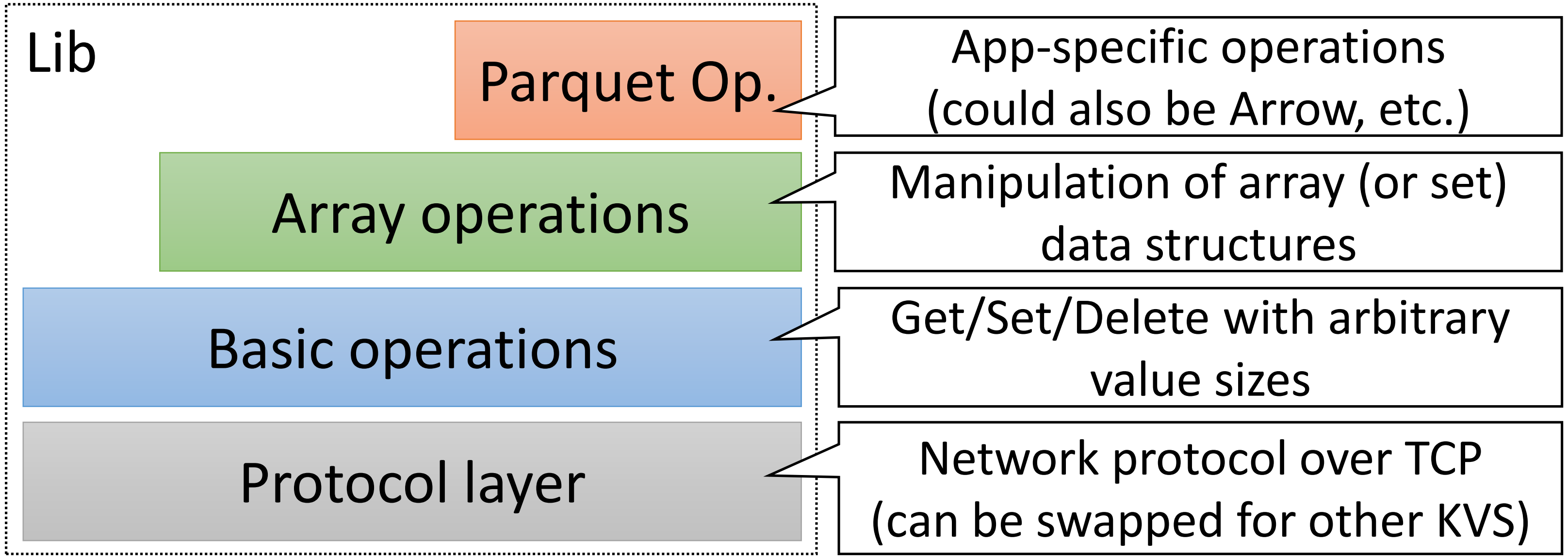
- Data stored as pages, organized by columns
  - Flexible access
  - Efficient processing
  - Compression

- Opportunity for deduplication:
  - Changes in columns do not impact others
  - Changes in rows often local to subset of pages
- We can deduplicate without having to chunk the file inside the storage node!
  - Alternative requires more compute (e.g. Rabin-Karp hash)



SW Library

- Written in Go
- Hides FPGA idiosyncrasies, takes advantage of pipelining, low latency



Lib

Parquet Op.

Array operations

Basic operations

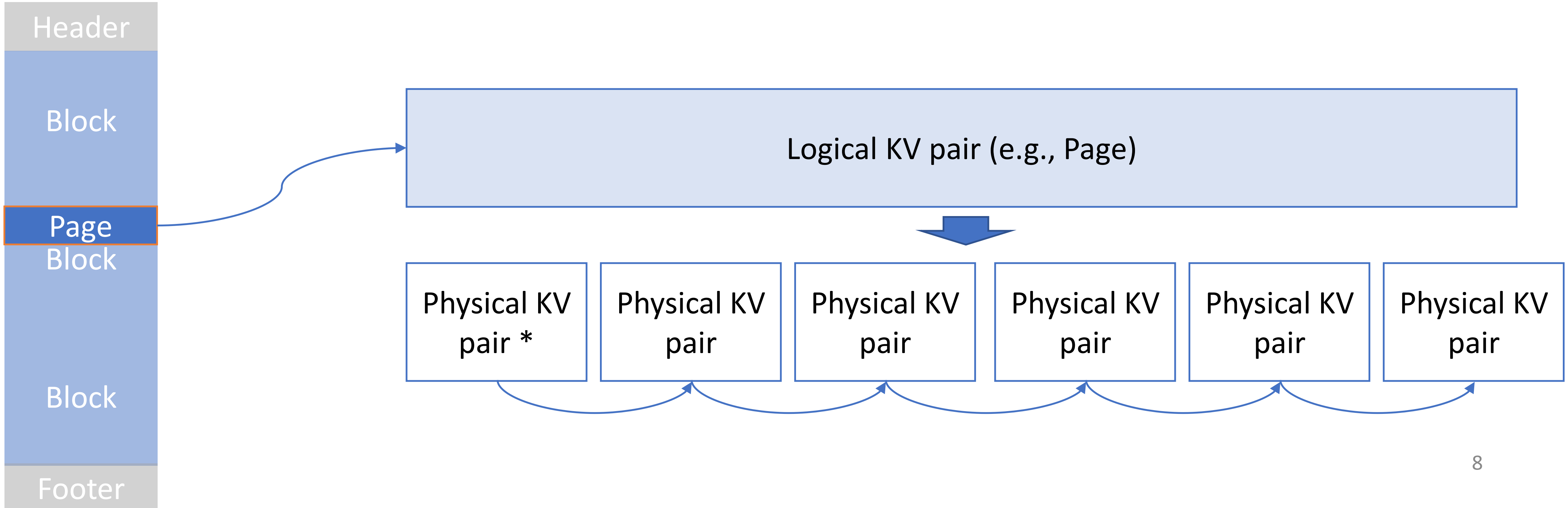
Protocol layer

App-specific operations (could also be Arrow, etc.)

Manipulation of array (or set) data structures

Get/Set/Delete with arbitrary value sizes

Network protocol over TCP (can be swapped for other KVS)



Header

Block

Page

Block

Block

Footer

Logical KV pair (e.g., Page)

Physical KV pair *

Physical KV pair

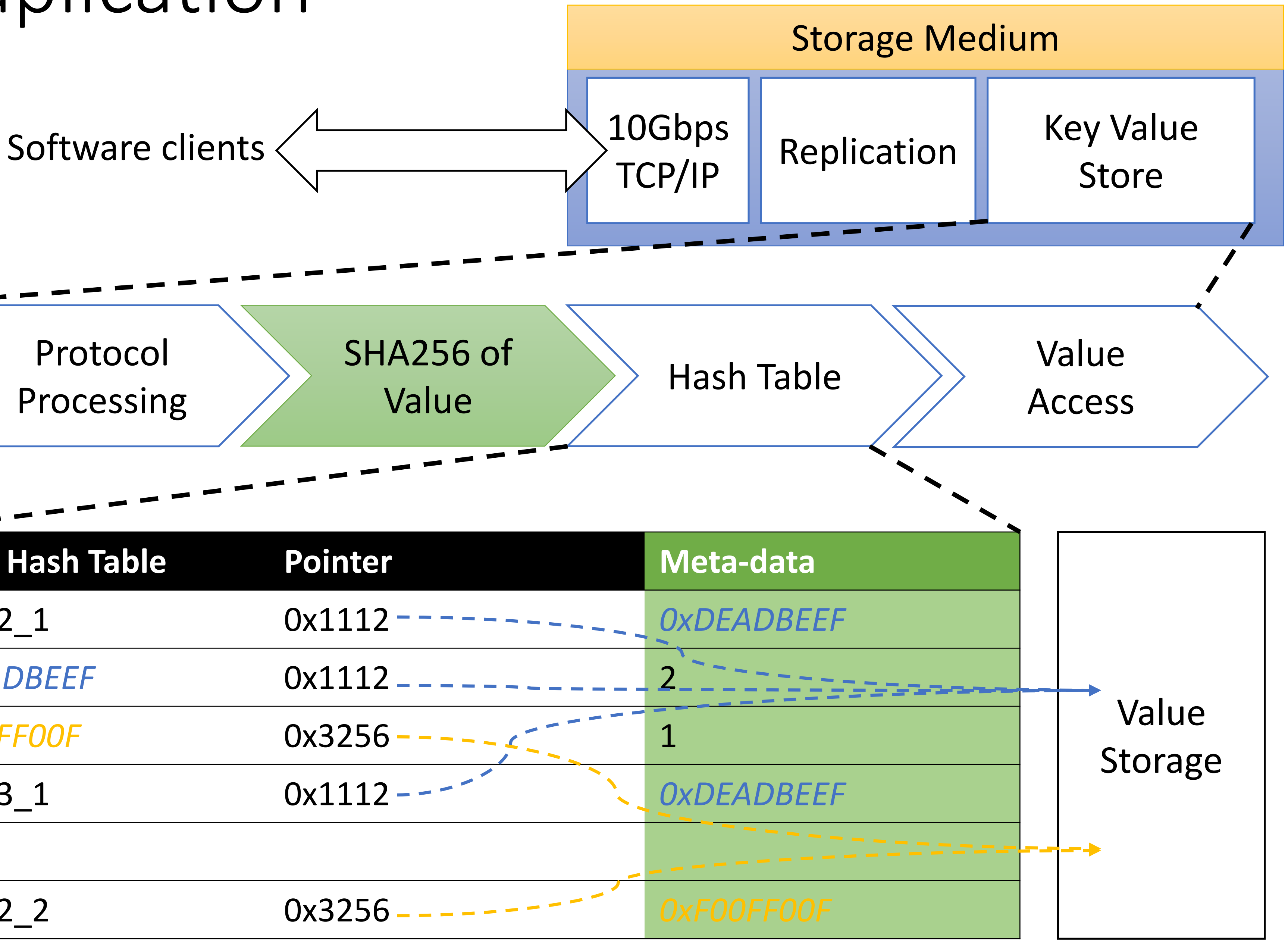
Physical KV pair

Physical KV pair

Physical KV pair

Physical KV pair

In-line deduplication



Key in Hash Table	Pointer	Meta-data
Page_2_1	0x1112	*0xDEADBEEF*
0xDEADBEEF	0x1112	2
0xF00FF00F	0x3256	1
Page_3_1	0x1112	*0xDEADBEEF*
Page_2_2	0x3256	*0xF00FF00F*

Data Access from Apps

- The library exposes experimental bindings to C and Python
- Access data easily by column
- Allow processing pushdown in the future
- Python example:

```
h = pq.connect(‘11.1.212.209:2880’)
md = pq.open_metadata(h, ‘p001’, schema=0)

airline = pq.get_string_column(h, md, 1)
weight = pq.get_int_column(h, md, 10)

df = pd.DataFrame(data={'a':airline, 'w':weight})
df.sort_values(by=['w'], inplace=True)
print(df.head(5))
```

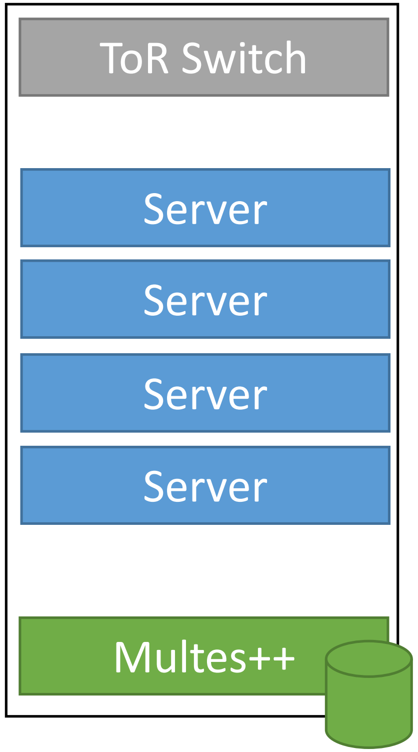
Access storage node and get Parquet schema

Read columns of interest

Perform computation of interest

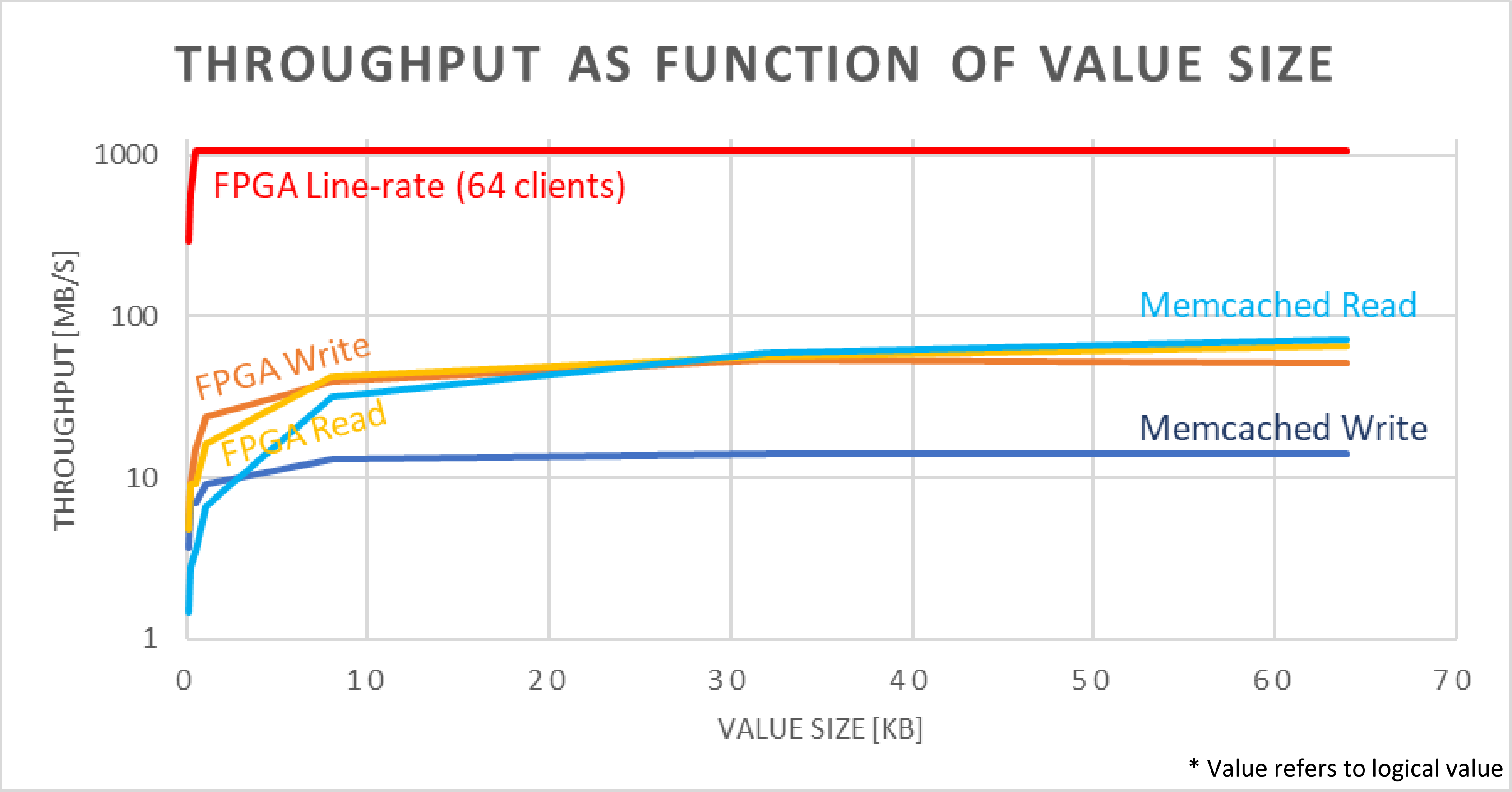
Evaluation

- KVS: Xilinx VC709 (8GB DRAM)
    - In the meantime: ported to VCU1525 (64GB RAM), experimented with Optane NVDIMM timings
- 4x servers: Intel Xeon Silver 4114 CPU and 10Gbps TCP/IP networking
- Datasets from https://datasf.org/opendata/
- Focus on writes and deduplication
    - Read throughput is not impacted



Throughput with deduplication

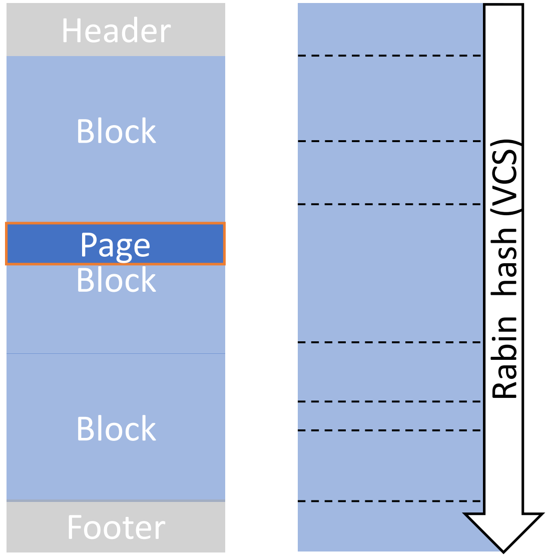
- As expected from Hardware: line-rate operation
  - Even single-threaded clients have good performance

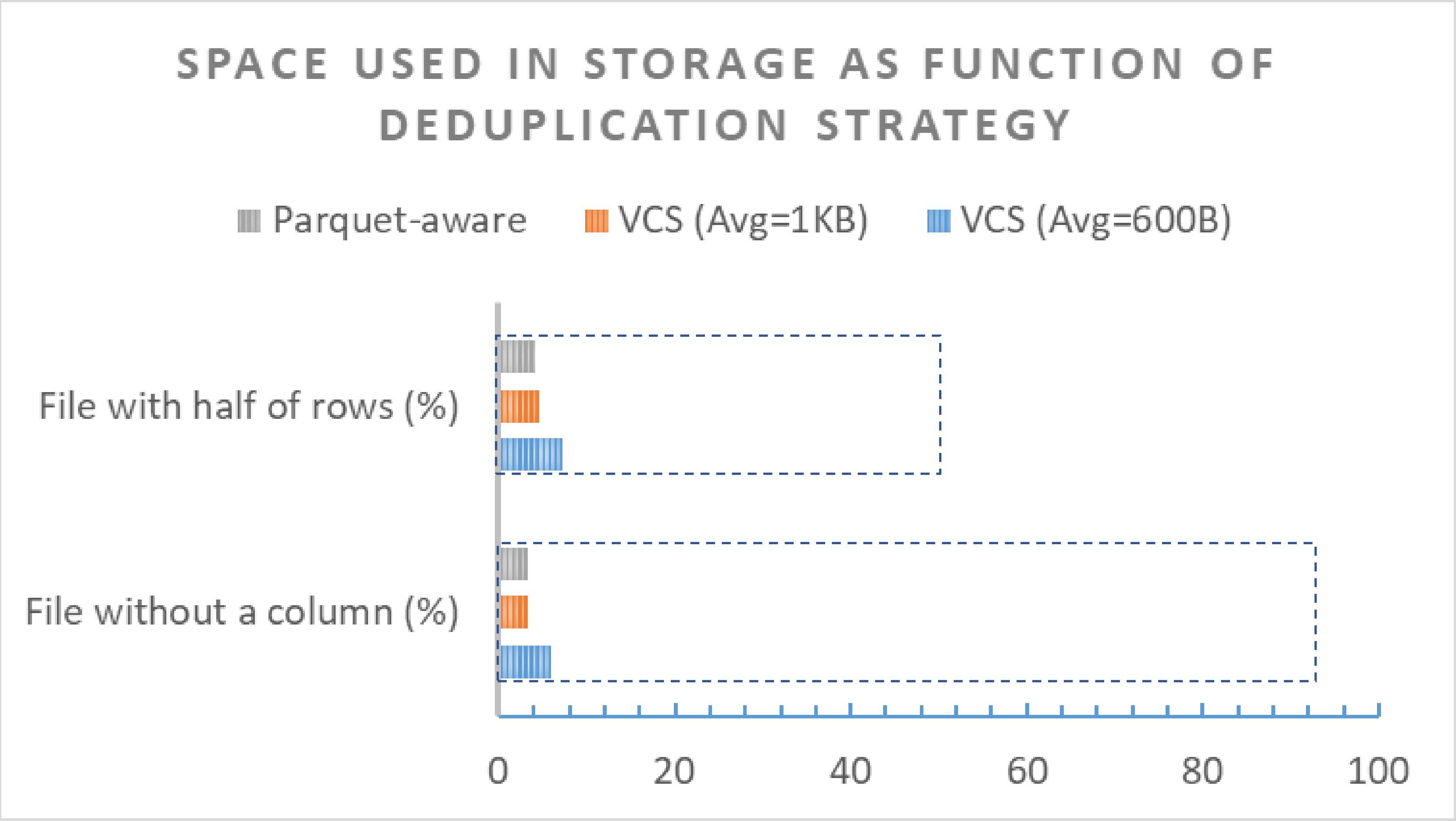


* Value refers to logical value

Deduplication Effectiveness

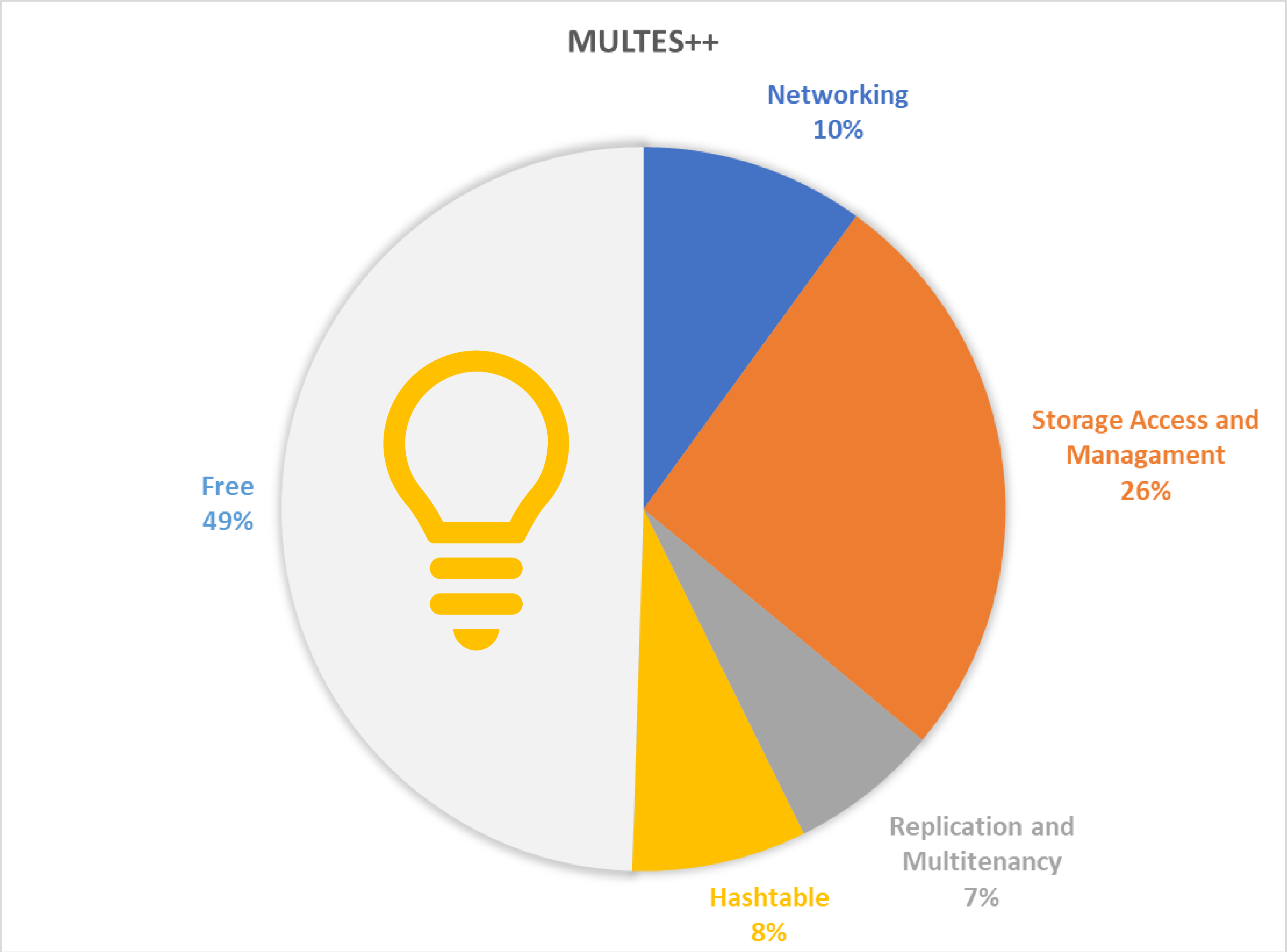
- As good as VCS and requires no additional computation





Used Police dataset: https://data.sfgov.org/Public-Safety/Police-Department-Incident-Reports-Historical-2003/tmnf-yvry

Resource Consumption



Closing Thoughts

We should use more specialized hardware in the cloud but design with service-centric view

- Parquet-aware KVS as an example
  - Deduplication, data management logic is common for tenants → HW
  - Each tenant can have different file format/library → SW
  - Benefits of specialized solution, flexibility of software

➢ In-storage processing of column chunks – no complex parsing is needed

What areas outside of ML/Analytics would benefit from Smart Storage?

What types of applications/services have a common element like KVSs?

How do we systematically split applications across devices?