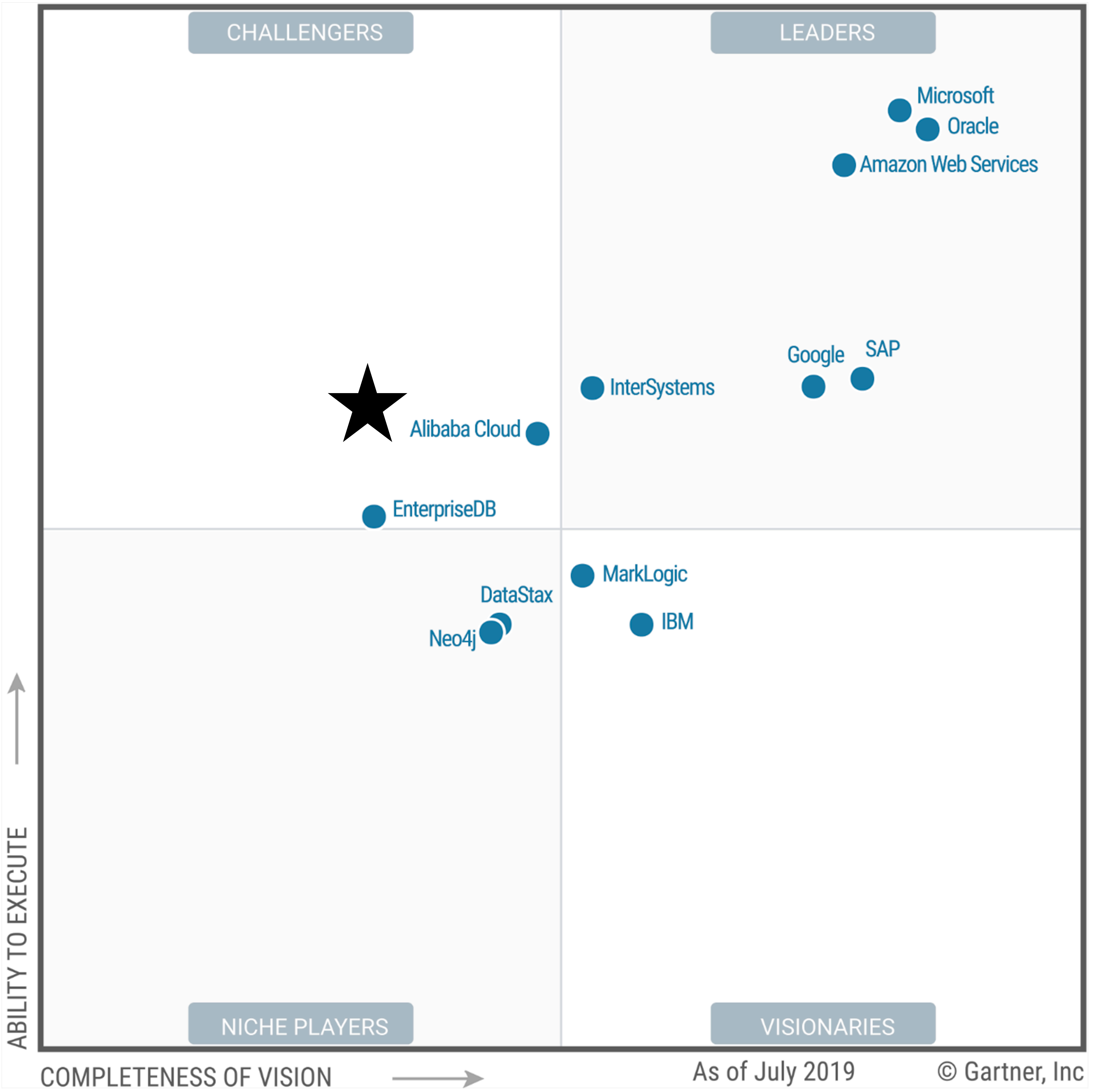
Timon: A Timestamped Event Database for Efficient Telemetry Data Processing and Analytics

Wei Cao, **Yusong Gao**, Feifei Li, Sheng Wang, Bingchen Lin, Ke Xu,
Xiaojie Feng, Yucong Wang, Zhenjun Liu, Gejin Zhang
Alibaba Cloud Database Department

2020.06

Alibaba Cloud

Database services at Alibaba Cloud



As of July 2019 ©Gantner,inc

Cloud Database Marketing:

1st, Asia Pacific

Database Products and Services:

26 Products or Services

Enterprise Users:

100 thousands

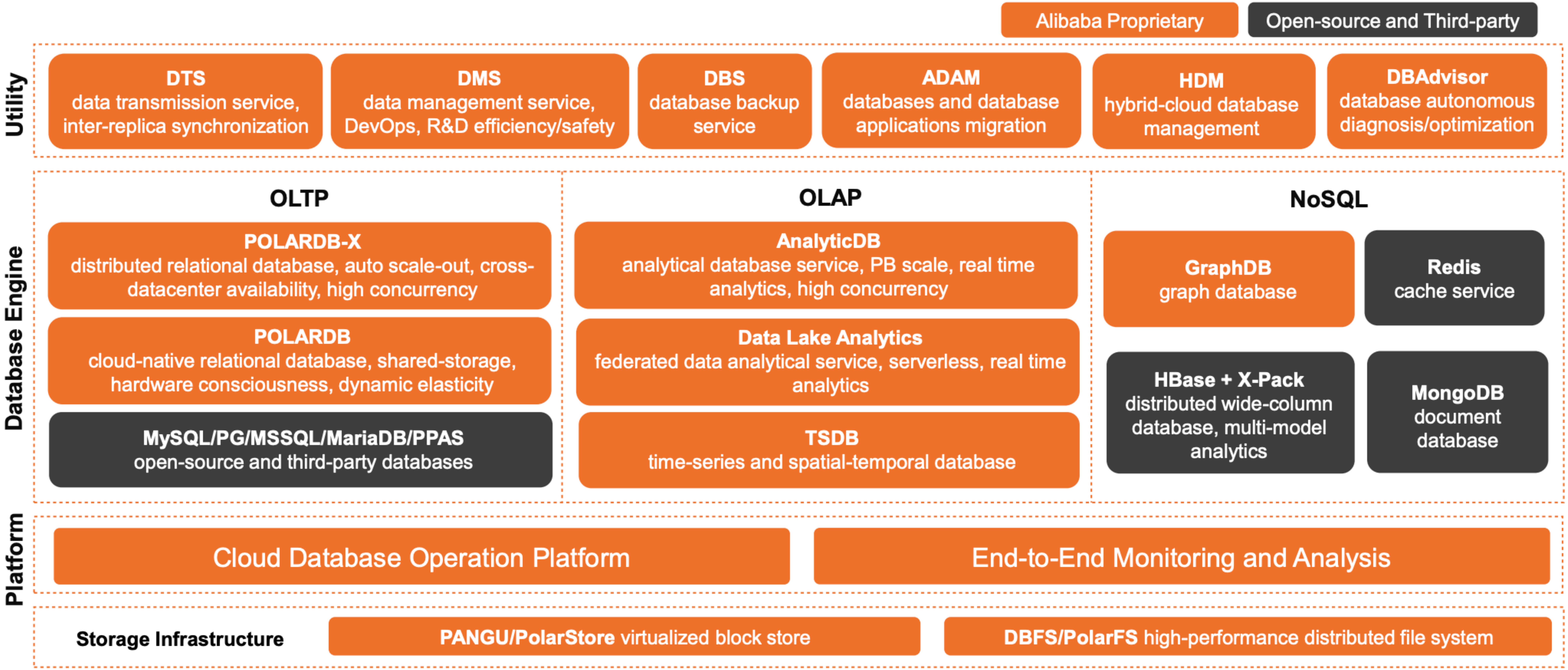
Databases Migrated:

400 thousands

Expressway for running fully managed databases on Alibaba Cloud. https://www.youtube.com/watch?v=5VkLDC_uIxM

Alibaba Cloud

Database systems and services at Alibaba and Alibaba Cloud



Alibaba Proprietary

Open-source and Third-party

Utility

DTS
data transmission service, inter-replica synchronization

DMS
data management service, DevOps, R&D efficiency/safety

DBS
database backup service

ADAM
databases and database applications migration

HDM
hybrid-cloud database management

DBAdvisor
database autonomous diagnosis/optimization

Database Engine

OLTP

POLARDB-X
distributed relational database, auto scale-out, cross-datacenter availability, high concurrency

POLARDB
cloud-native relational database, shared-storage, hardware consciousness, dynamic elasticity

MySQL/PG/MSSQL/MariaDB/PPAS
open-source and third-party databases

OLAP

AnalyticDB
analytical database service, PB scale, real time analytics, high concurrency

Data Lake Analytics
federated data analytical service, serverless, real time analytics

TSDB
time-series and spatial-temporal database

NoSQL

GraphDB
graph database

Redis
cache service

HBase + X-Pack
distributed wide-column database, multi-model analytics

MongoDB
document database

Platform

Cloud Database Operation Platform

End-to-End Monitoring and Analysis

Storage Infrastructure

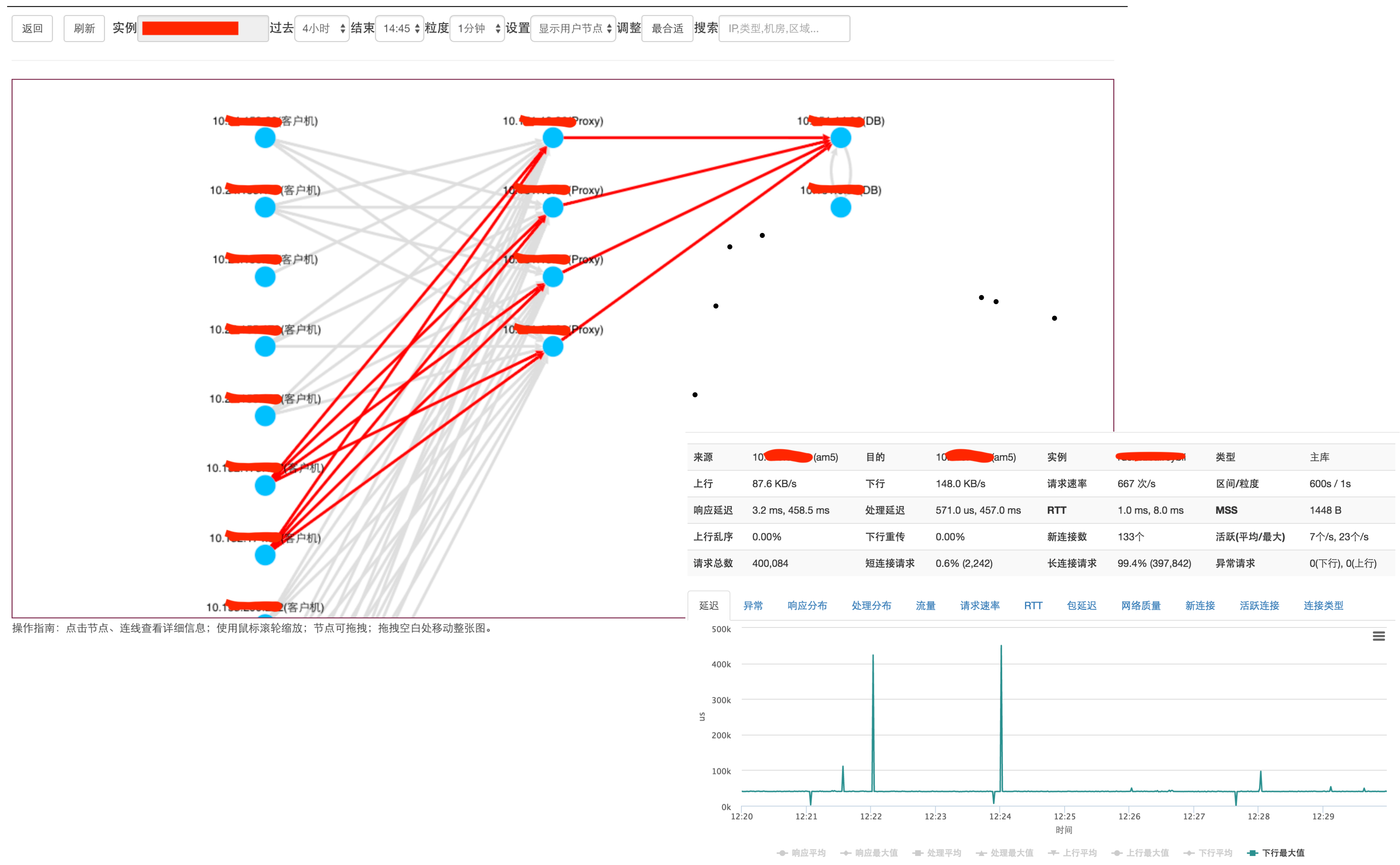
PANGU/PolarStore virtualized block store

DBFS/PolarFS high-performance distributed file system

Feifei Li. Cloud-Native Database Systems at Alibaba: Opportunities and Challenges. PVLDB, 12(12): 2263 - 2272, 2019. DOI: https://doi.org/10.14778/3352063.3352141

The scenarios of using telemetry data

End-to-End Tracking System



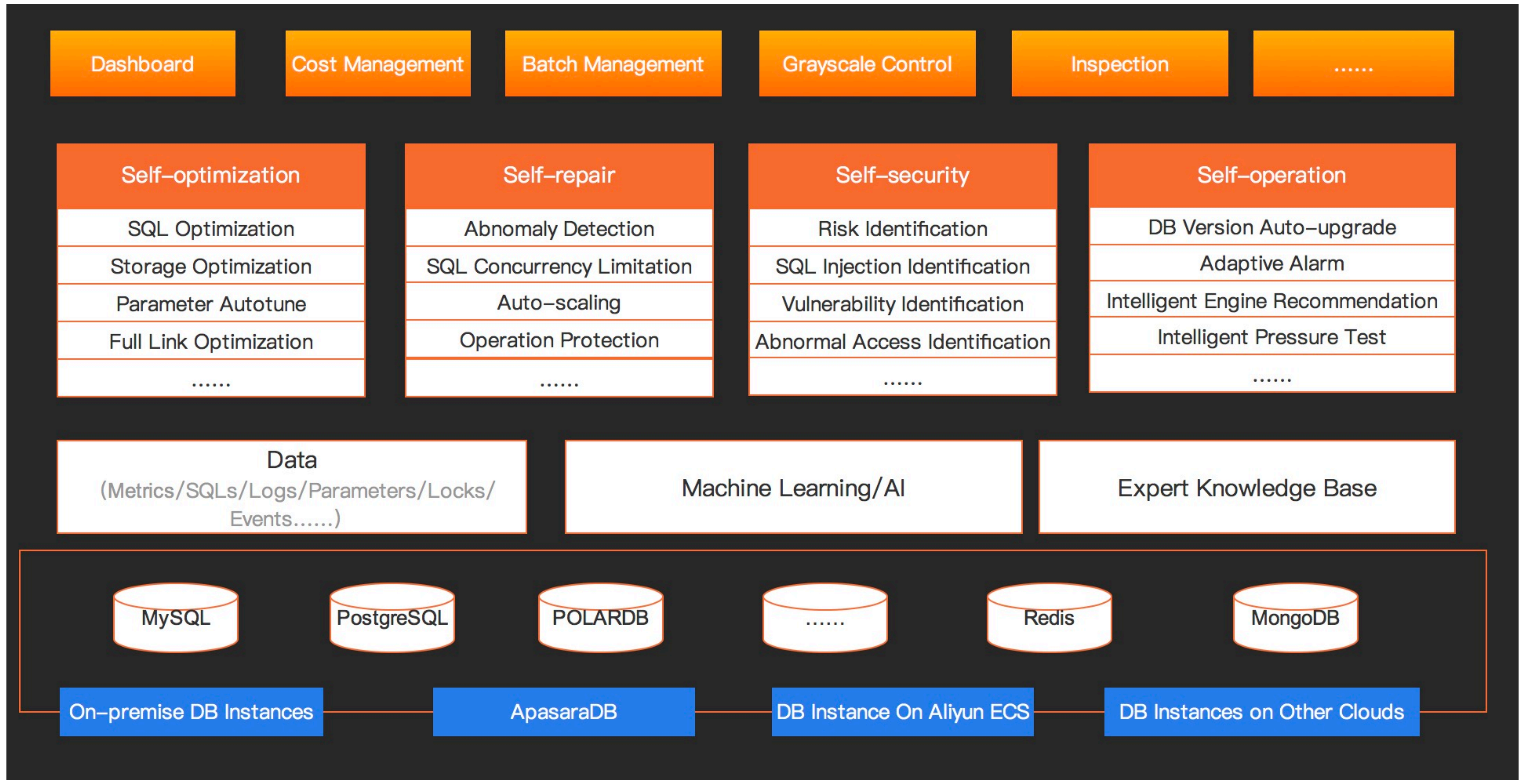
返回 刷新 实例 [redacted] 过去 4小时 结束 14:45 粒度 1分钟 设置 显示用户节点 调整 最合适 搜索 IP,类型,机房,区域...

操作指南：点击节点、连线查看详细信息；使用鼠标滚轮缩放；节点可拖拽；拖拽空白处移动整张图。

来源	10.[redacted](am5)	目的	10[redacted](am5)	实例	[redacted]	类型	主库
上行	87.6 KB/s	下行	148.0 KB/s	请求速率	667 次/s	区间/粒度	600s / 1s
响应延迟	3.2 ms, 458.5 ms	处理延迟	571.0 us, 457.0 ms	RTT	1.0 ms, 8.0 ms	MSS	1448 B
上行乱序	0.00%	下行重传	0.00%	新连接数	133个	活跃(平均/最大)	7个/s, 23个/s
请求总数	400,084	短连接请求	0.6% (2,242)	长连接请求	99.4% (397,842)	异常请求	0(下行), 0(上行)

SIGMOD'18 TcpRT: Instrument and Diagnostic Analysis System for Service Quality of Cloud Databases at Massive Scale in Real-time.

Database Autonomy Service (DAS)



Dashboard | Cost Management | Batch Management | Grayscale Control | Inspection | ......

Self–optimization
- SQL Optimization
- Storage Optimization
- Parameter Autotune
- Full Link Optimization
- ......

Self–repair
- Abnomaly Detection
- SQL Concurrency Limitation
- Auto–scaling
- Operation Protection
- ......

Self–security
- Risk Identification
- SQL Injection Identification
- Vulnerability Identification
- Abnormal Access Identification
- ......

Self–operation
- DB Version Auto–upgrade
- Adaptive Alarm
- Intelligent Engine Recommendation
- Intelligent Pressure Test
- ......

Data (Metrics/SQLs/Logs/Parameters/Locks/Events......)

Machine Learning/AI

Expert Knowledge Base

MySQL | PostgreSQL | POLARDB | ...... | Redis | MongoDB

On–premise DB Instances | ApasaraDB | DB Instance On Aliyun ECS | DB Instances on Other Clouds

https://www.alibabacloud.com/help/doc-detail/64851.htm?spm=a2c63.p38356.b99.2.61d09bb0Xe1MPU

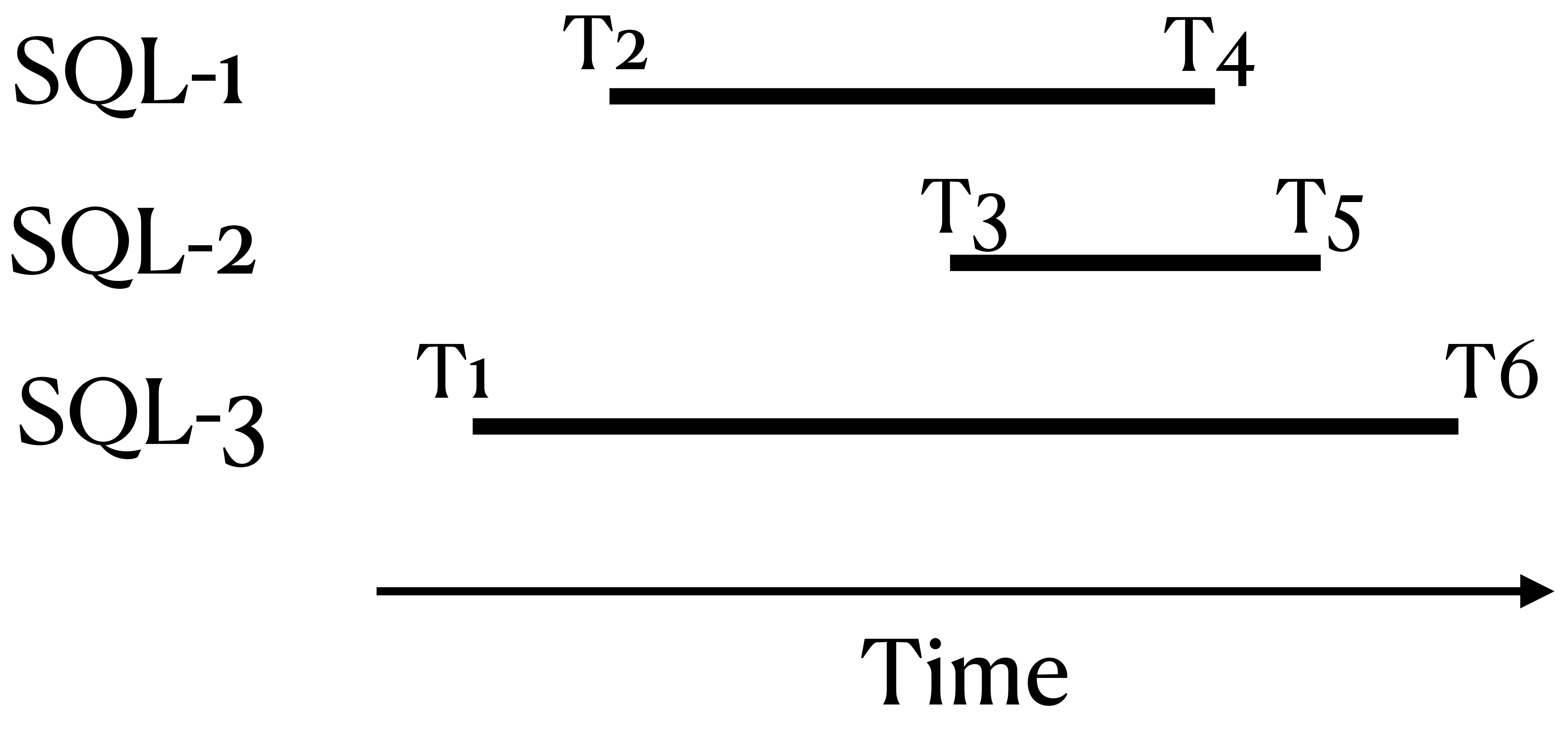
Challenge of processing telemetry data in cloud database services

- ~10 million objects
  - Cover database engine, network, operating system, and even each individual OS process.
- Support 1 second granularity and hundreds of millions data points per second
  - Find and explain peaks which last a short time.
- Support long-term & multiple granularities queries with low response latency
  - Find trend and periodicity and compare with historical data
- Strict SLA, realtime & accurate even when out-of-order events exist
  - Find and explain anomalies as soon as possible.
  - Not just for monitoring but also for autonomous optimization.

Sources of out-of-order events

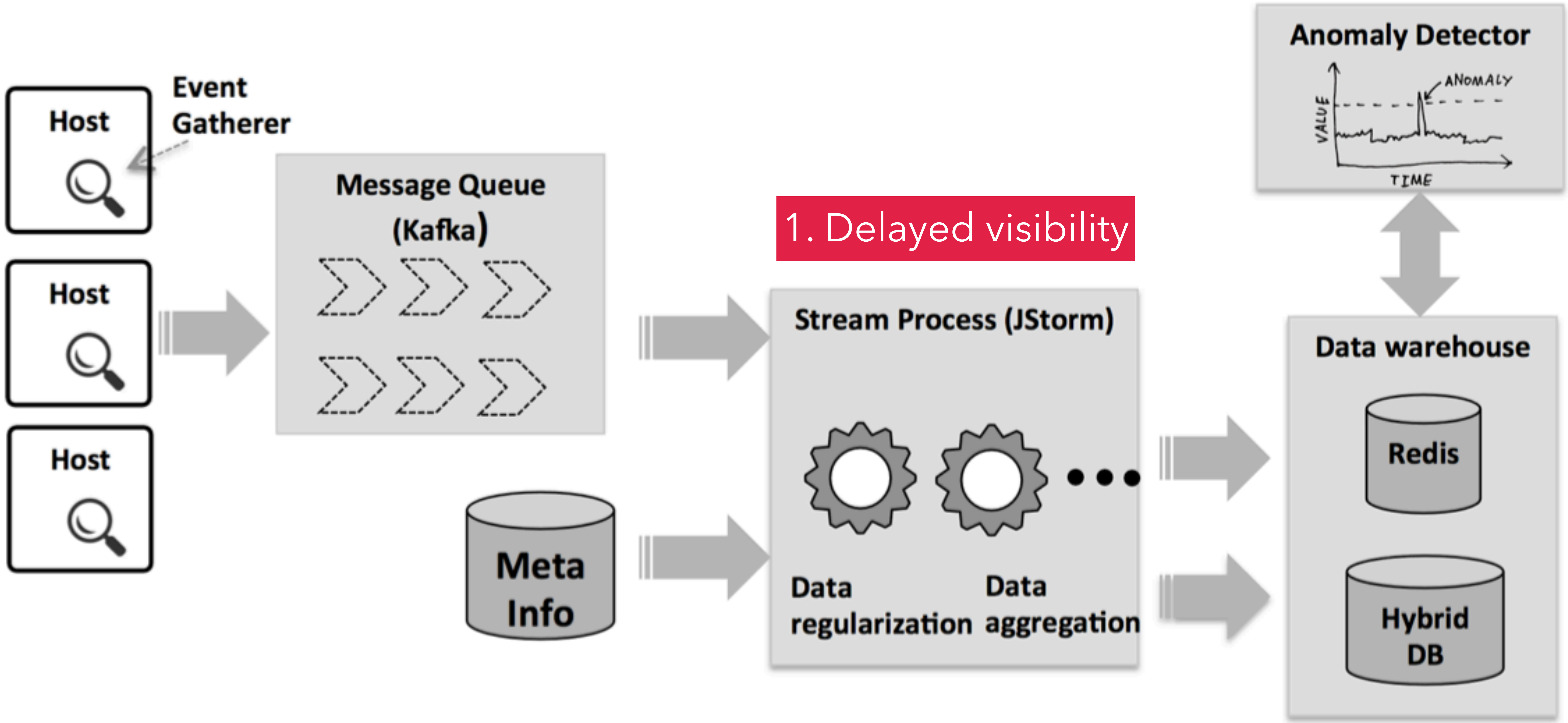
- Distributed Computing Environment
  - Machine failures
  - Network failures
  - Clock skew
  - etc…

- Application Natures
  - For instance: TcpRT events can’t be collected until requests complete. (53.90% out-of-order events)



The order of events arrival will be T2, T3, T1

Our first generation data processing system

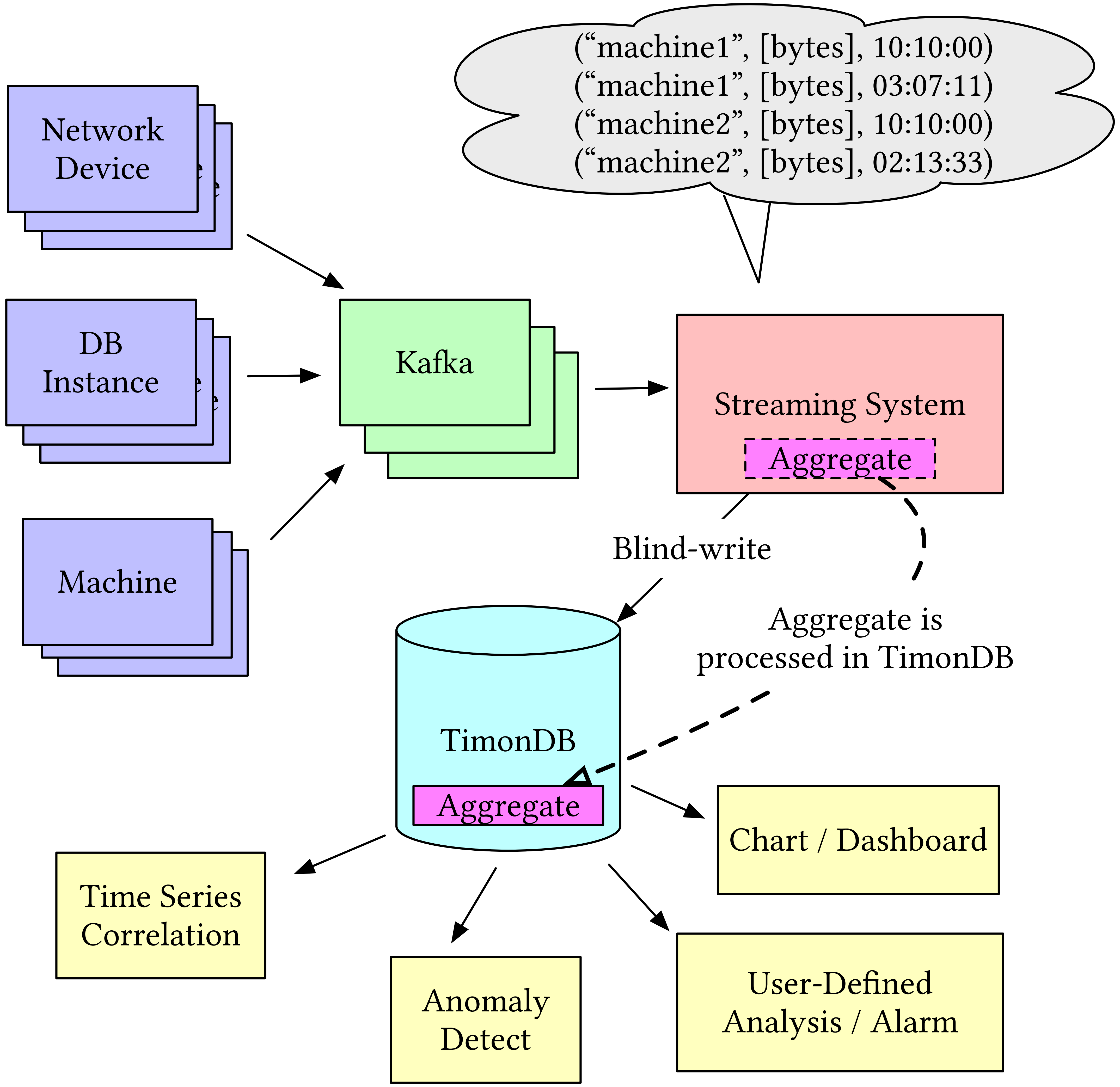


1. Delayed visibility

3. It's not good for analysis on massive and long-term events

2. When late data arrives, a Read-Modify-Write operation is required to read and update the previously aggregated value on external storage.

Solution: using blind-write and moving aggregate to storage

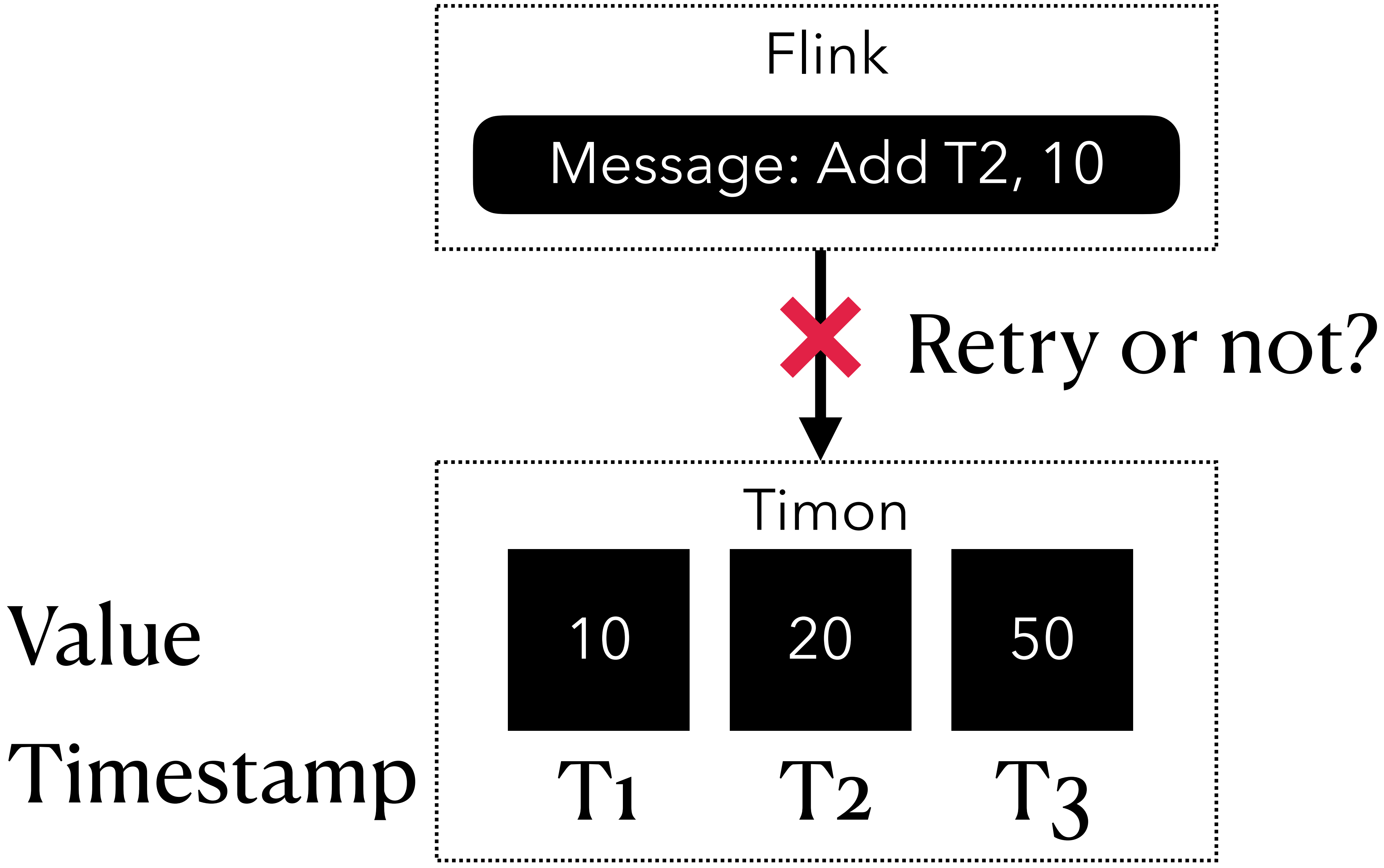

("machine1", [bytes], 10:10:00)
("machine1", [bytes], 03:07:11)
("machine2", [bytes], 10:10:00)
("machine2", [bytes], 02:13:33)
Network Device
DB Instance
Machine
Kafka
Streaming System
Aggregate
Blind-write
Aggregate is processed in TimonDB
TimonDB
Aggregate
Chart / Dashboard
Time Series Correlation
Anomaly Detect
User-Defined Analysis / Alarm

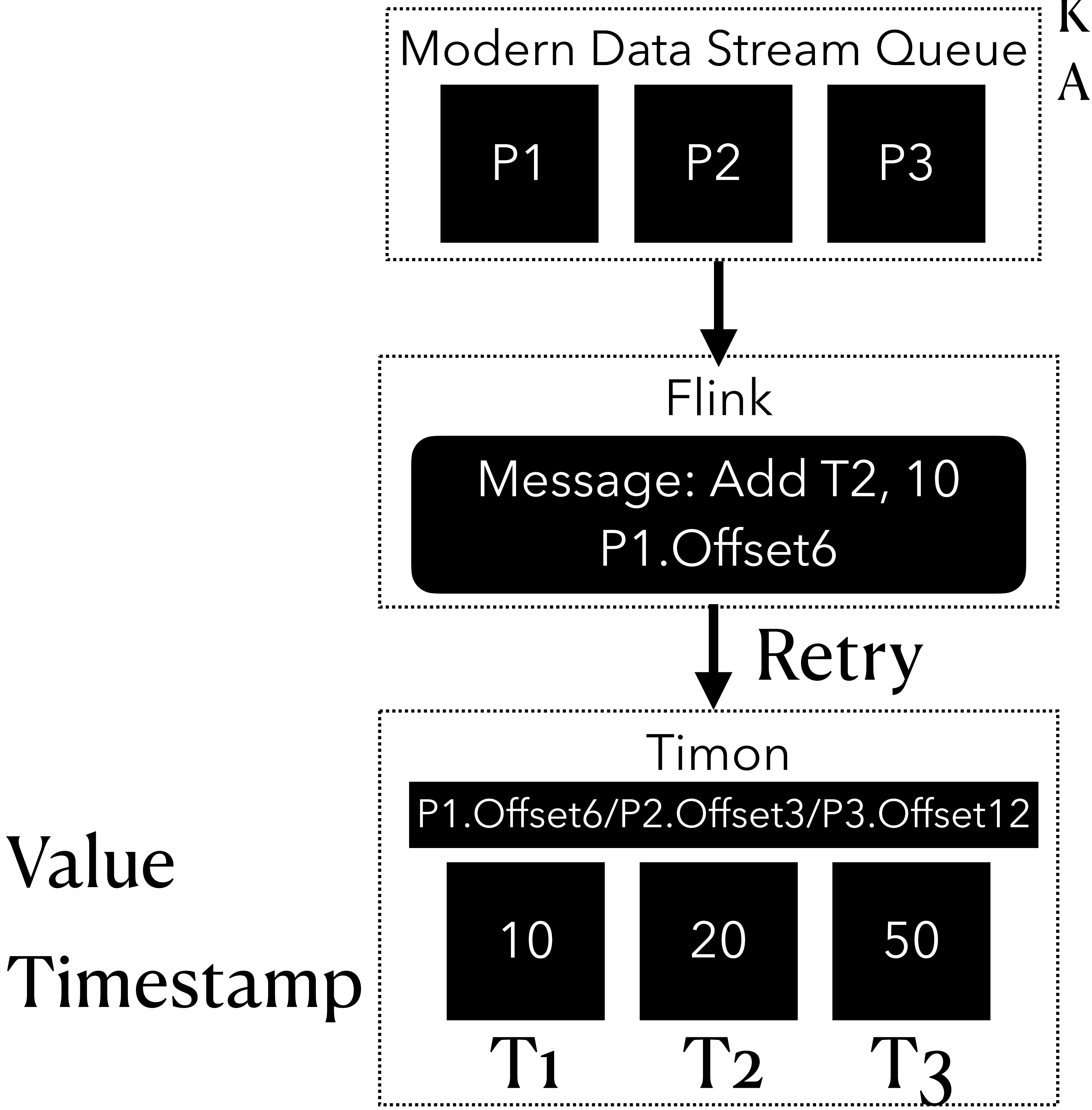
- The most common operators are associative and commutative.
  - **sum**, **max**, **min**, **avg**, **stddev**
  - **quantile**: histogram, t-digest*
  - **distinct**: HyperLogLog
- So it's a good idea to support incremental processing after a data point is written.

* t-digest: https://github.com/tdunning/t-digest

Incremental Processing Issue: Idempotence

- If failovers occur while the data points are being written, we have to remove the points that have been successfully written.





Kafka/AWS Kinesis/
Aliyun Loghub etc..

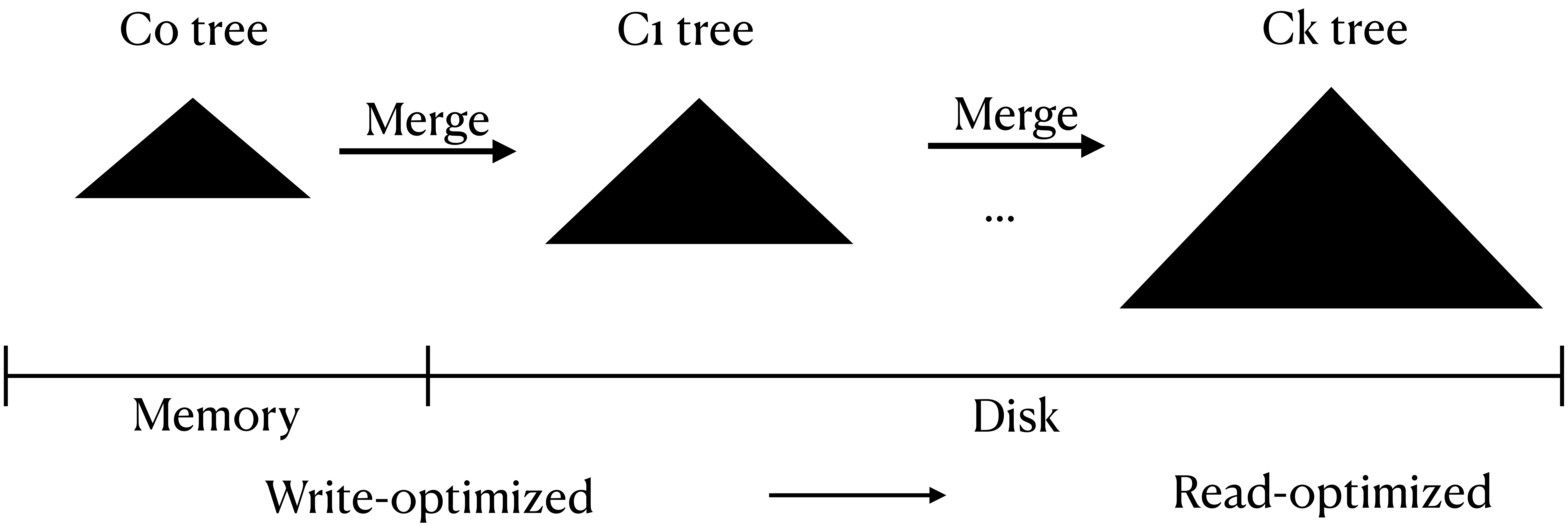
Each partition is an ordered sequence of records.

Each record has an offset.

Timon keeps the maximum offset it encountered for each partition.

How to support blind-write and incremental processing efficiently

- LSM-Tree by P O’Neil - 1996



C0 tree

Merge

C1 tree

Merge

...

Ck tree

Memory

Disk

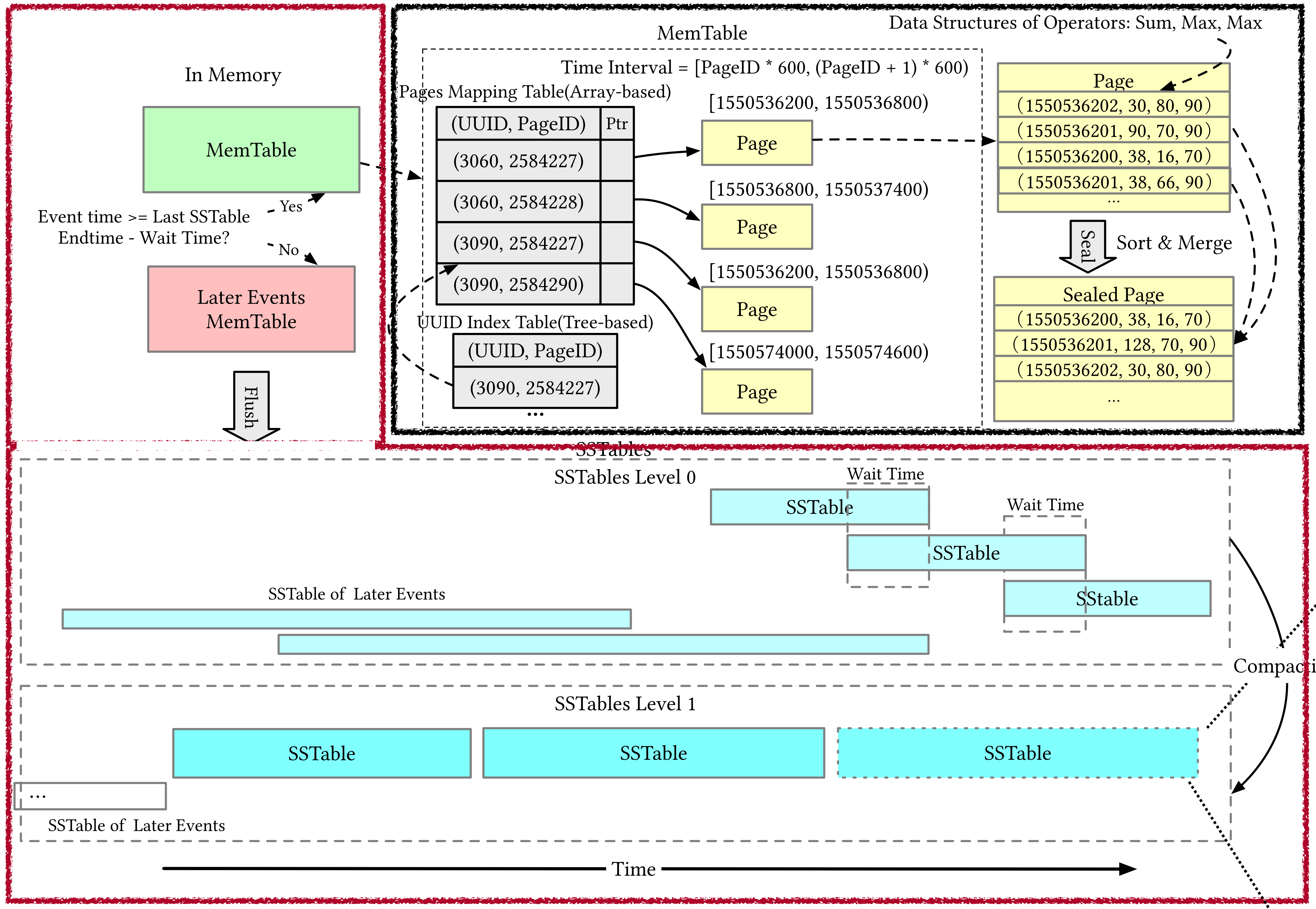
Write-optimized → Read-optimized

Append & merge to tolerate out-of-order events.

Build time-partitioning tree index for long-term queries.

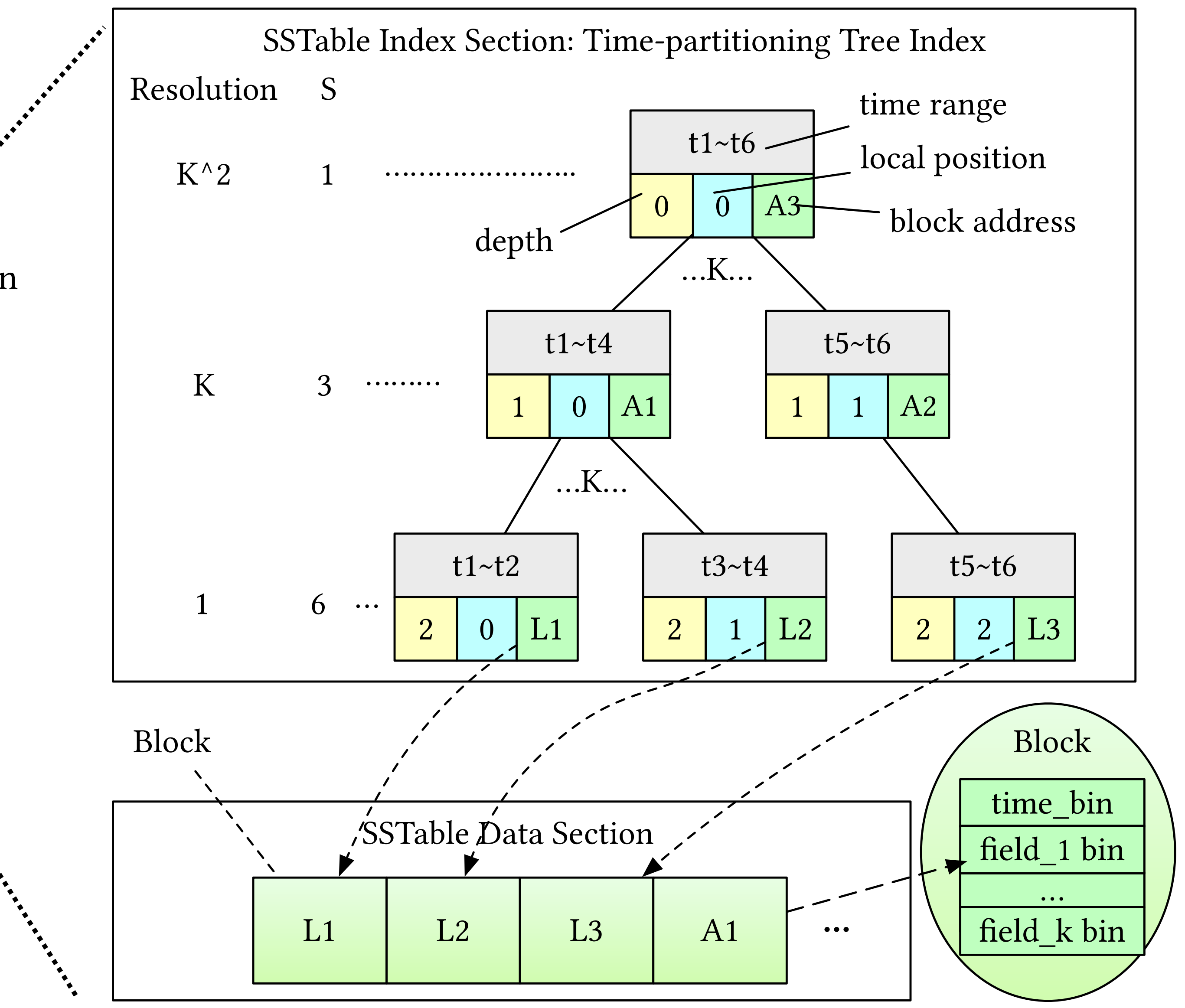
Time-Segment Log-Structured Merge-Tree

Lazy Merge



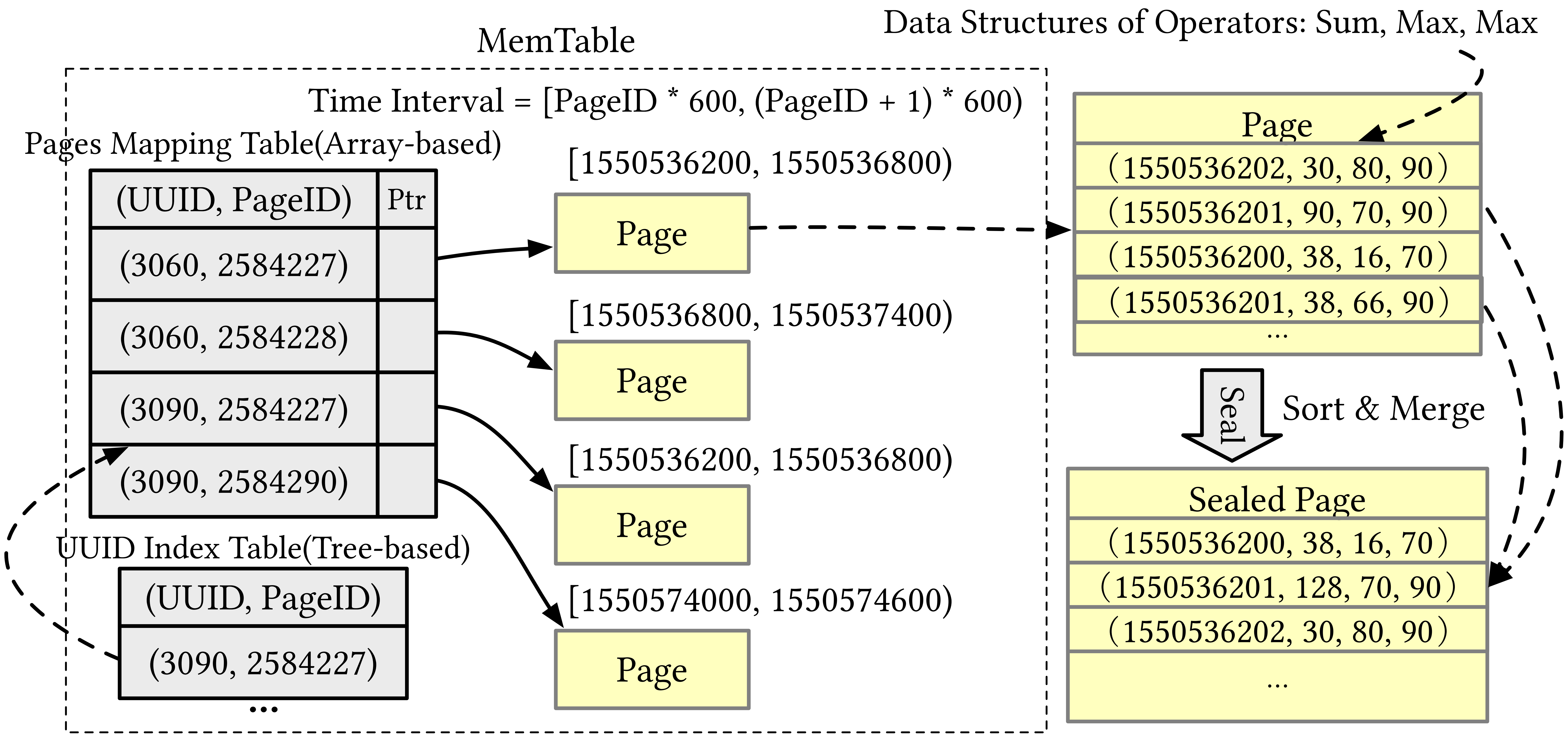
MemTable is optimized for time series data.

Build time-partitioning tree index when compacting.



Optimized MemTable for Time Series

- Hybrid Tree-based Structure and Array-based Structure
- Pages are used to process sparse events.



MemTable

Data Structures of Operators: Sum, Max, Max

Time Interval = [PageID * 600, (PageID + 1) * 600)

Pages Mapping Table(Array-based)

(UUID, PageID)	Ptr
(3060, 2584227)	
(3060, 2584228)	
(3090, 2584227)	
(3090, 2584290)	

UUID Index Table(Tree-based)

(UUID, PageID)
(3090, 2584227)
...

[1550536200, 1550536800)

Page

[1550536800, 1550537400)

Page

[1550536200, 1550536800)

Page

[1550574000, 1550574600)

Page

Page
(1550536202, 30, 80, 90)
(1550536201, 90, 70, 90)
(1550536200, 38, 16, 70)
(1550536201, 38, 66, 90)
...

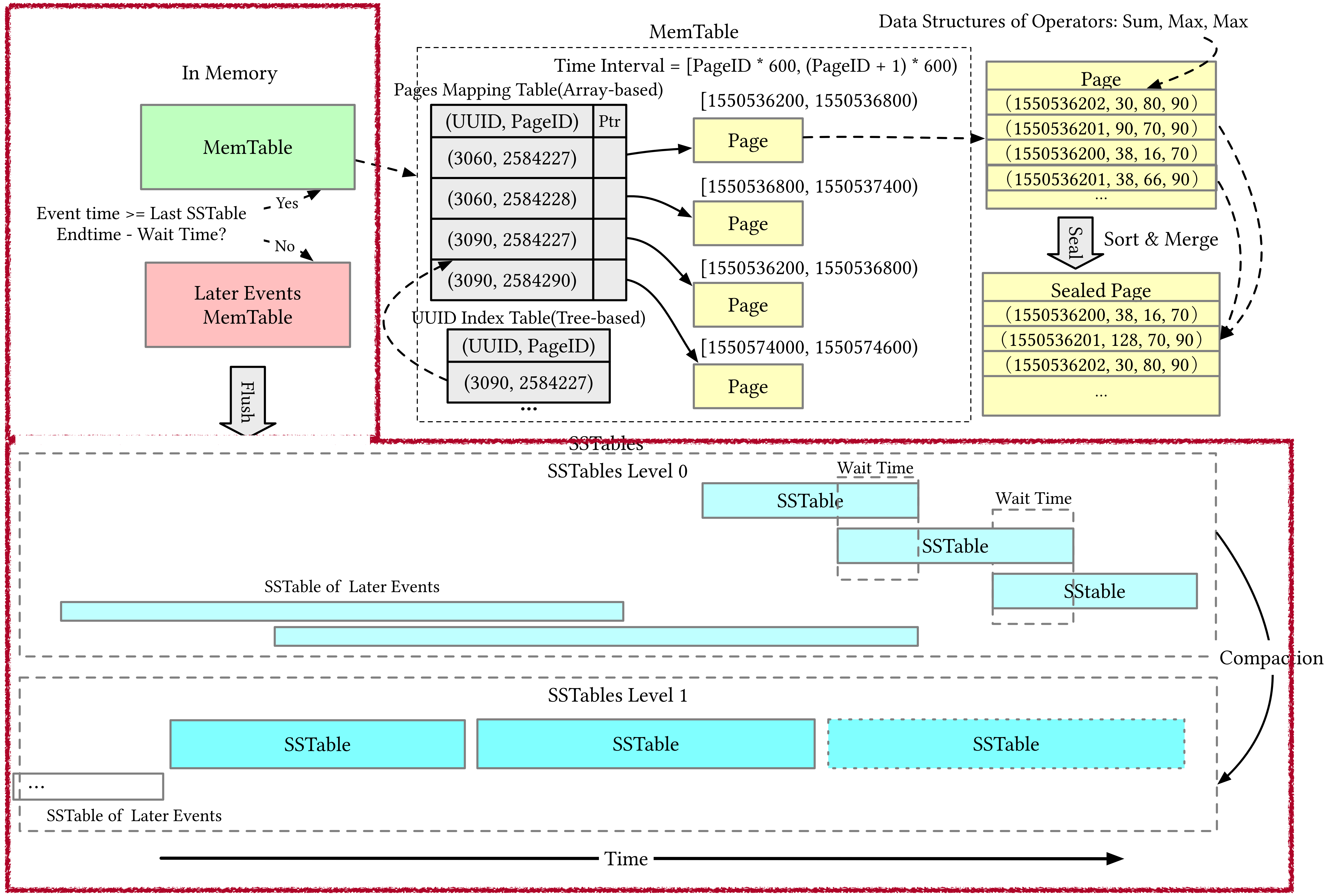
Seal

Sort & Merge

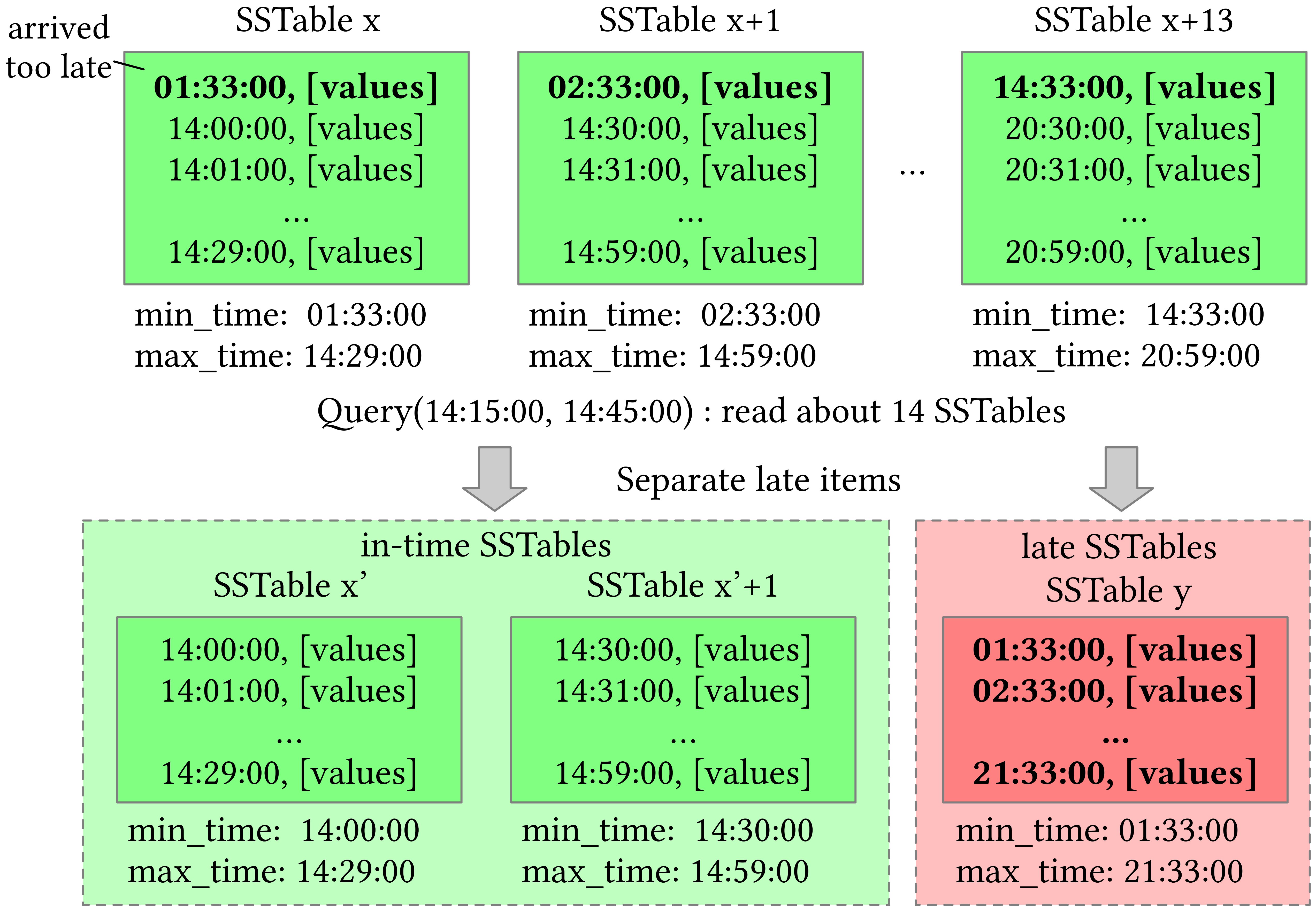
Sealed Page
(1550536200, 38, 16, 70)
(1550536201, 128, 70, 90)
(1550536202, 30, 80, 90)
...

Tolerating out-of-order events (1)

- Most of out-of-order events arrive with a delay of less than 5 minutes.
- Some latecomers are small in proportion but wide in time range.



Tolerating out-of-order events (2)



arrived too late

SSTable x

01:33:00, [values]
14:00:00, [values]
14:01:00, [values]
…
14:29:00, [values]

min_time: 01:33:00
max_time: 14:29:00

SSTable x+1

02:33:00, [values]
14:30:00, [values]
14:31:00, [values]
…
14:59:00, [values]

min_time: 02:33:00
max_time: 14:59:00

…

SSTable x+13

14:33:00, [values]
20:30:00, [values]
20:31:00, [values]
…
20:59:00, [values]

min_time: 14:33:00
max_time: 20:59:00

Query(14:15:00, 14:45:00) : read about 14 SSTables

Separate late items

in-time SSTables

SSTable x'

14:00:00, [values]
14:01:00, [values]
…
14:29:00, [values]

min_time: 14:00:00
max_time: 14:29:00

SSTable x'+1

14:30:00, [values]
14:31:00, [values]
…
14:59:00, [values]

min_time: 14:30:00
max_time: 14:59:00

late SSTables

SSTable y

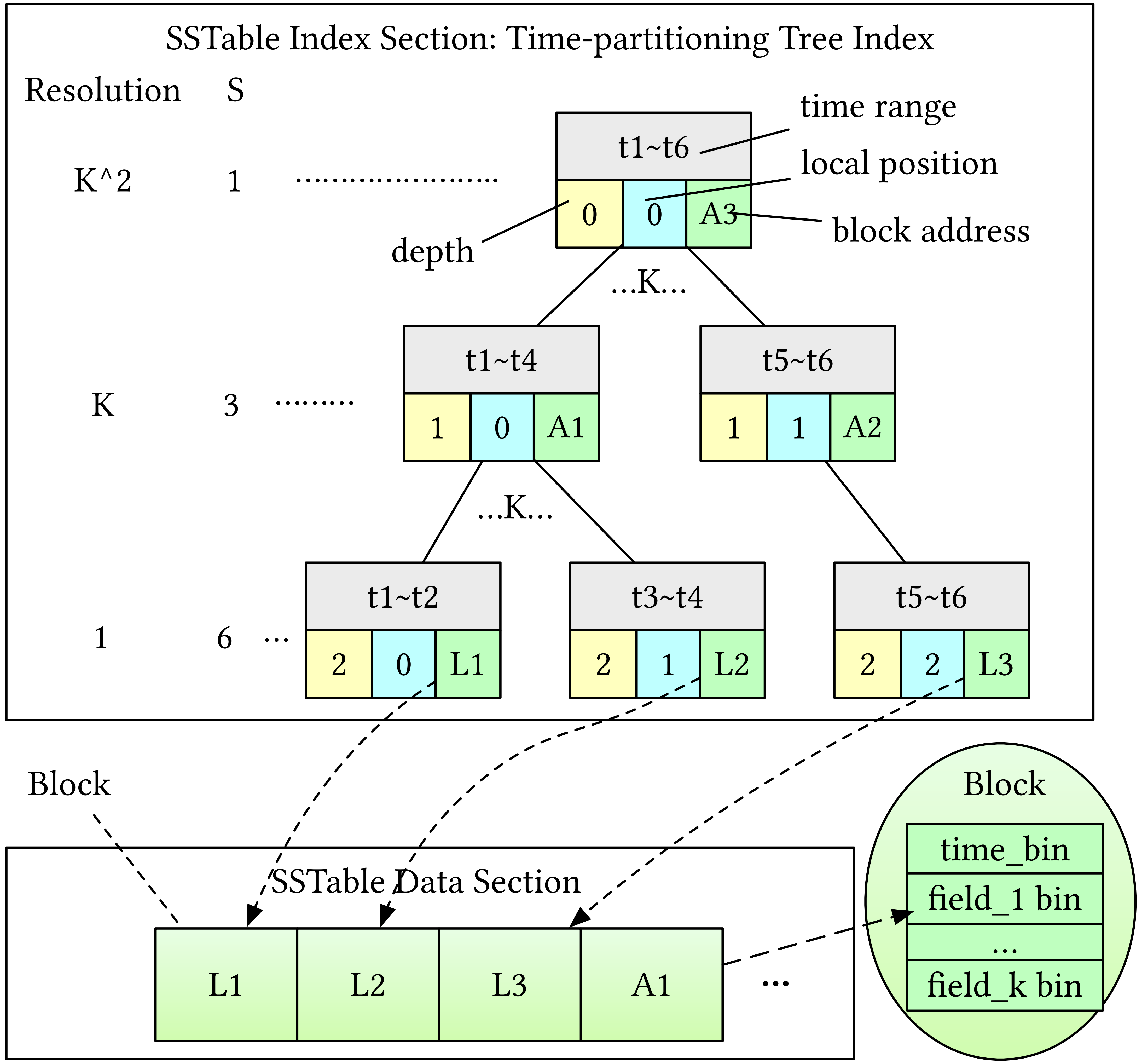
01:33:00, [values]
02:33:00, [values]
…
21:33:00, [values]

min_time: 01:33:00
max_time: 21:33:00

Query(14:15:00, 14:45:00) : read about 3 SSTables
(2 in-time SSTables and 1 late SSTables)

How to process aggregated queries efficiently

- Build time-partitioning tree index by compaction for fast exploration of long-term time-series.
- An aggregated query will scan recent data from MemTable and Lo SSTable, and historical data from time-partitioning tree.

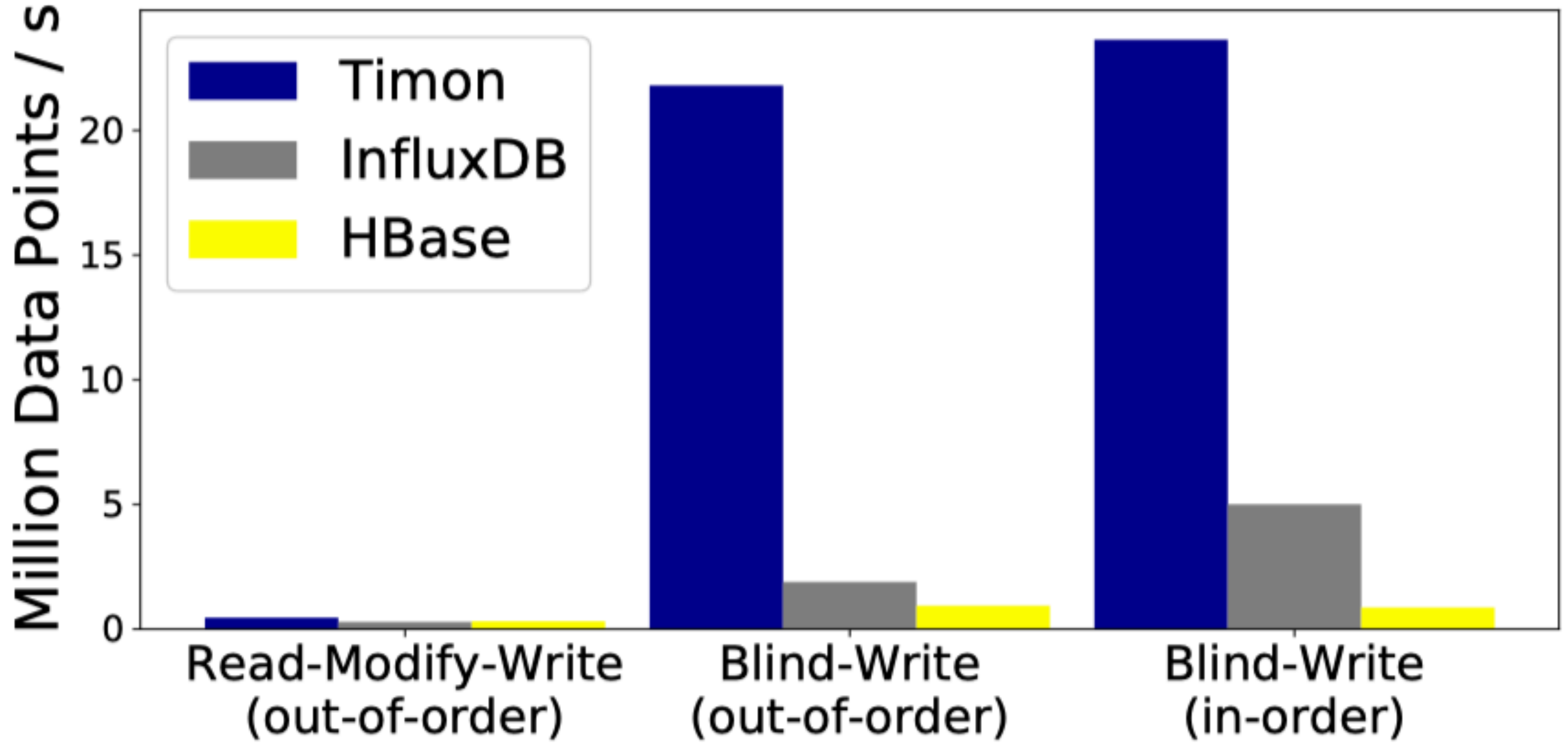


Timon with user-friendly tools and facilities

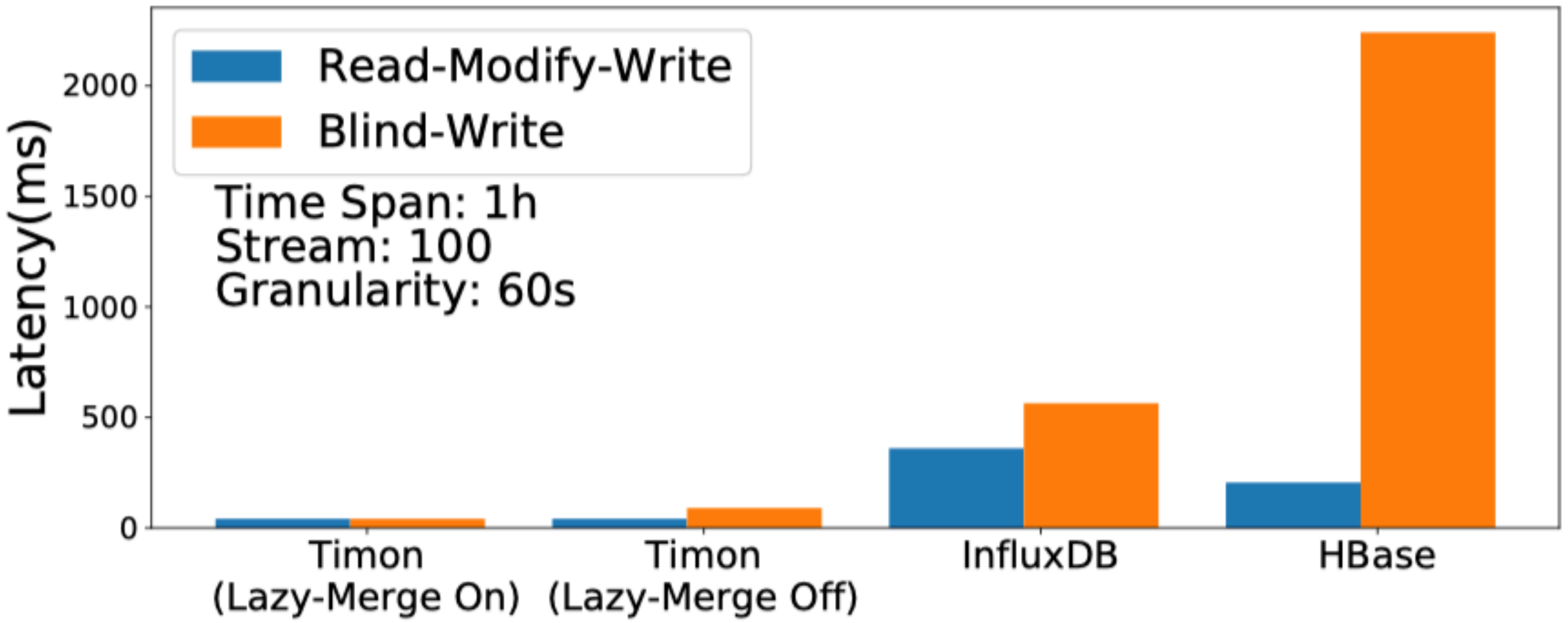
- We enhance Timon with user-friendly tools and facilities, such as metric set, materialized view and TQL.
  - Metric set
    - For record which contains dozens of metric values.
  - Materialized view
    - Aggregating data on a higher abstract level, e.g., the region level.
  - TQL
    - SQL-like query languages which allows users to retrieve and analyze the underlying timestamped event data with rich semantics.

Alibaba Cloud

Benchmark



Throughput with different write mode

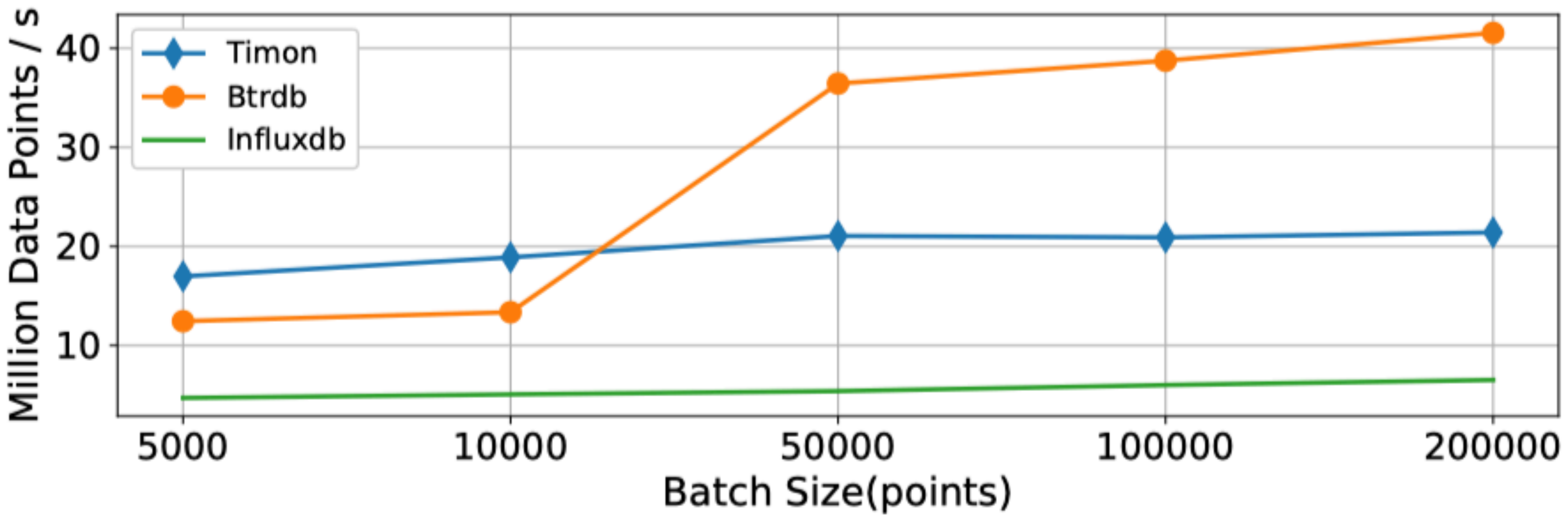


Query latency with different write mode.

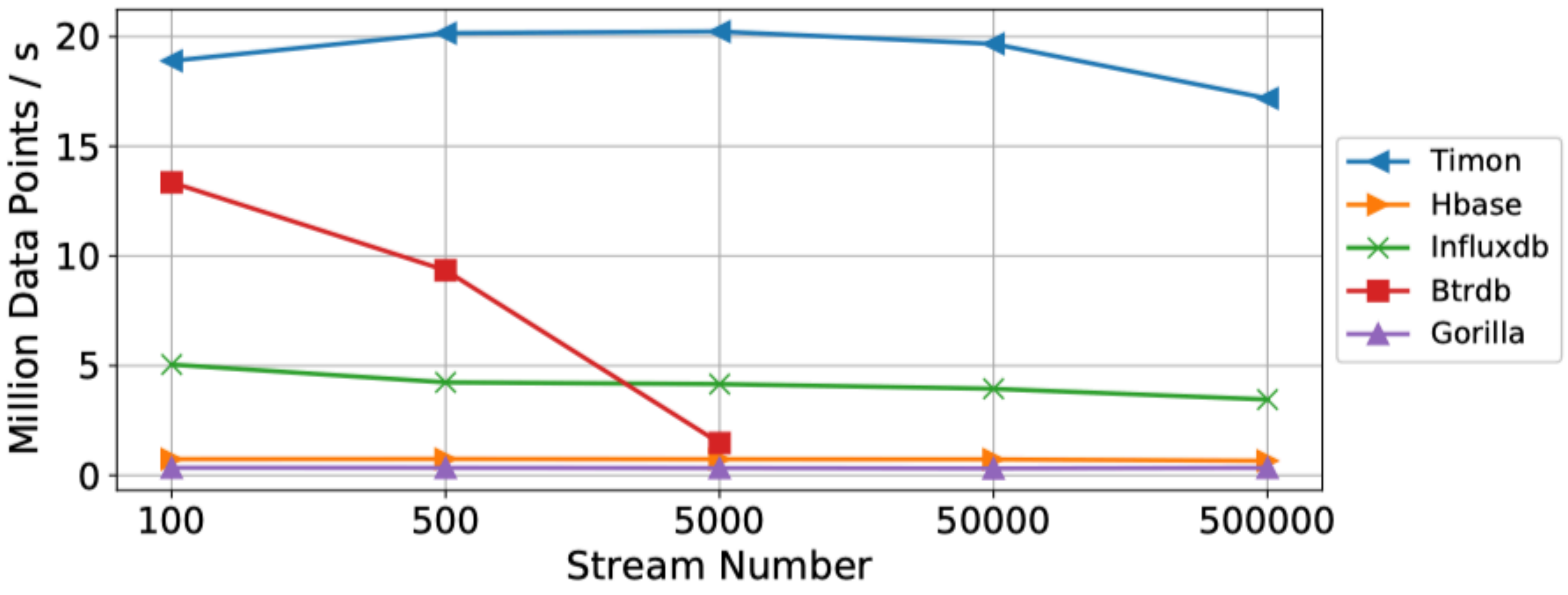
Alibaba Cloud

Benchmark: Write Throughput as batch size grow or streams grow

- BtrDB (M. P. Andersen at FAST 2016) is a state-of-the-art TSDB with very high performance and long-term time-series exploration support.
- BtrDB is much better for ultra-high frequency data points (i.e., sub-microsecond precision timestamps).
- Timon maintains high write performance stably when the number of streams increases.
- The main difference is that Timon builds Time-partitioning Tree Indexes by a batch and async procedure, but BtrDB inserts records directly into its tree indexes.



(a) Write Throughput as batch size grow



(b) Write Throughput as streams grow

Benchmark: Query Latency

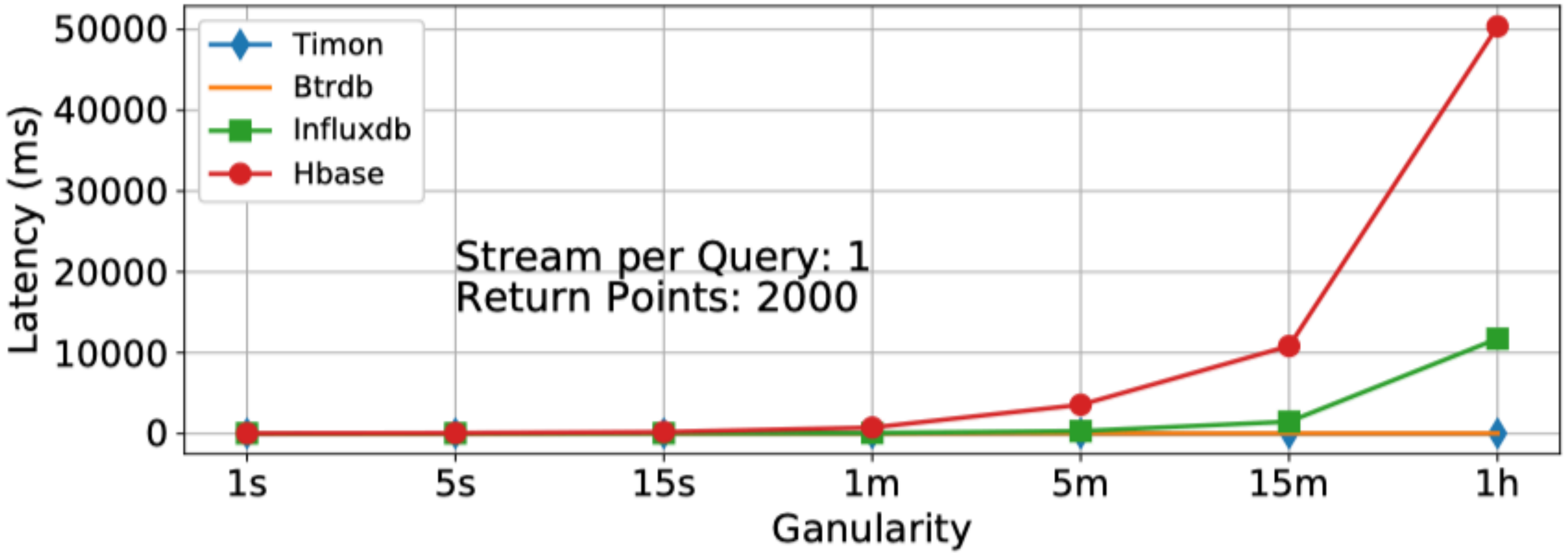
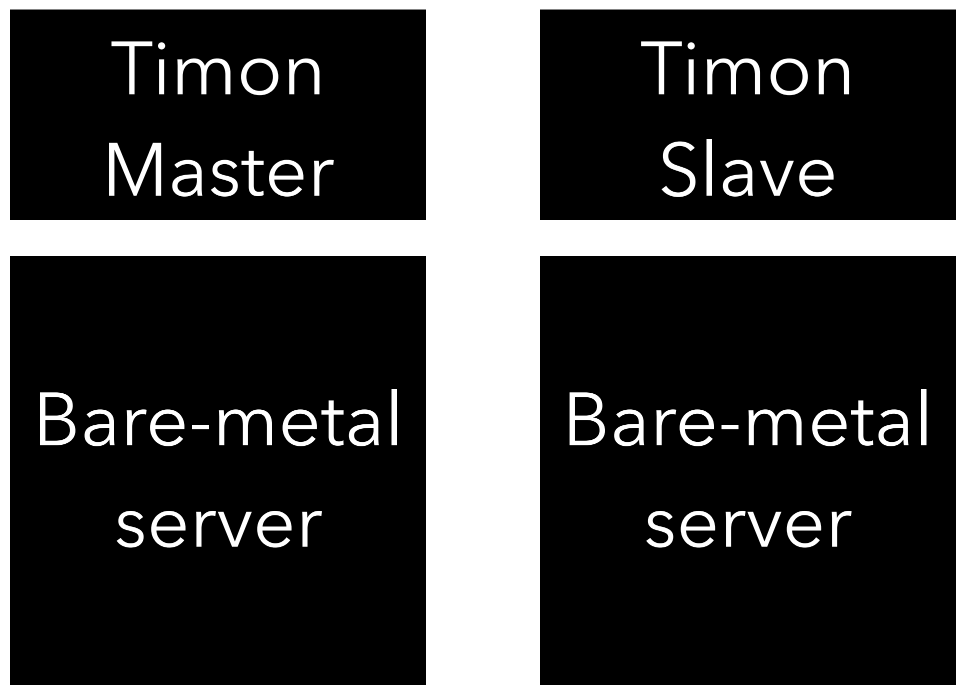


Figure 17: Query latencies between databases as time granularity changes.

Deployment

- Deployed in data centers distributed in 21 regions around the world.
- The biggest application cluster: 97 Timon nodes support about 500 million data points writing per second from the system, and the busiest node serves about 18 million data points per second.



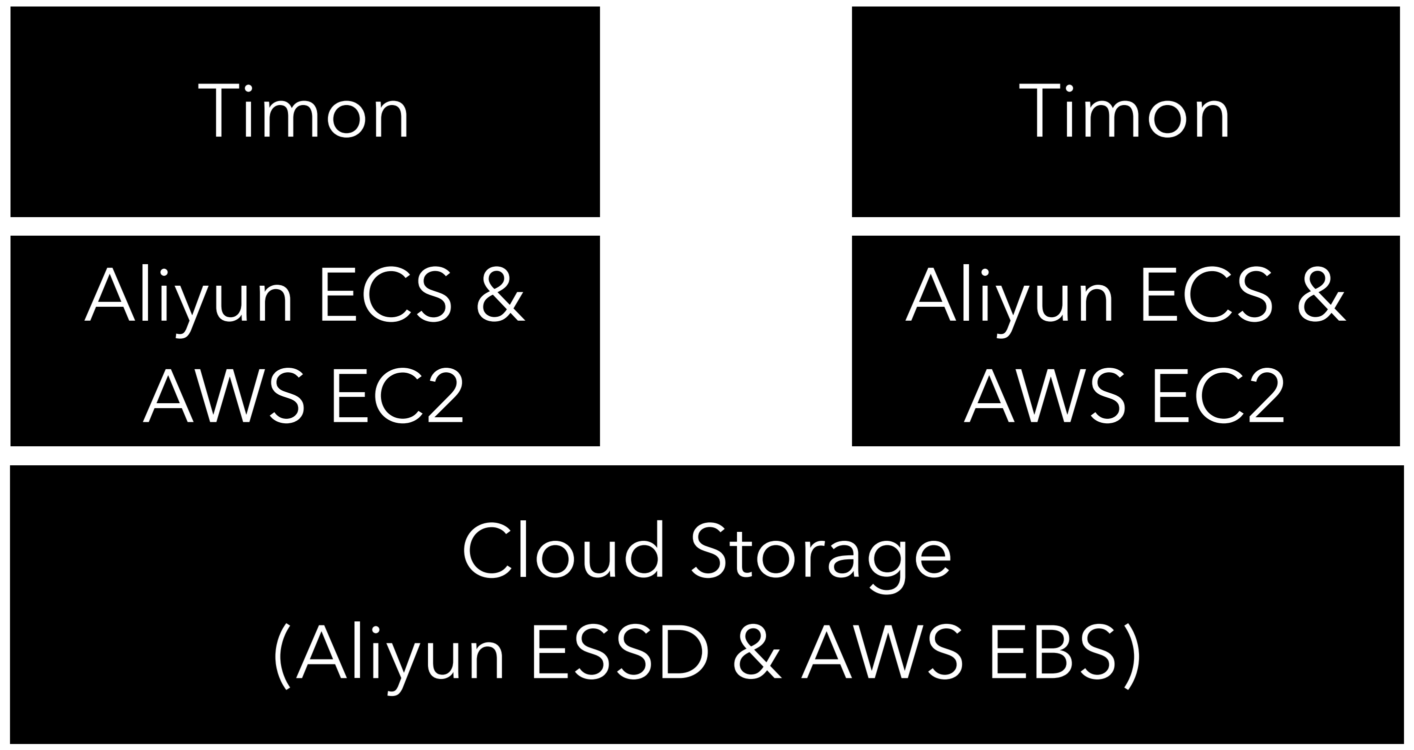
Timon Master

Timon Slave

Bare-metal server

Bare-metal server

On Bare-metal Server



Timon

Timon

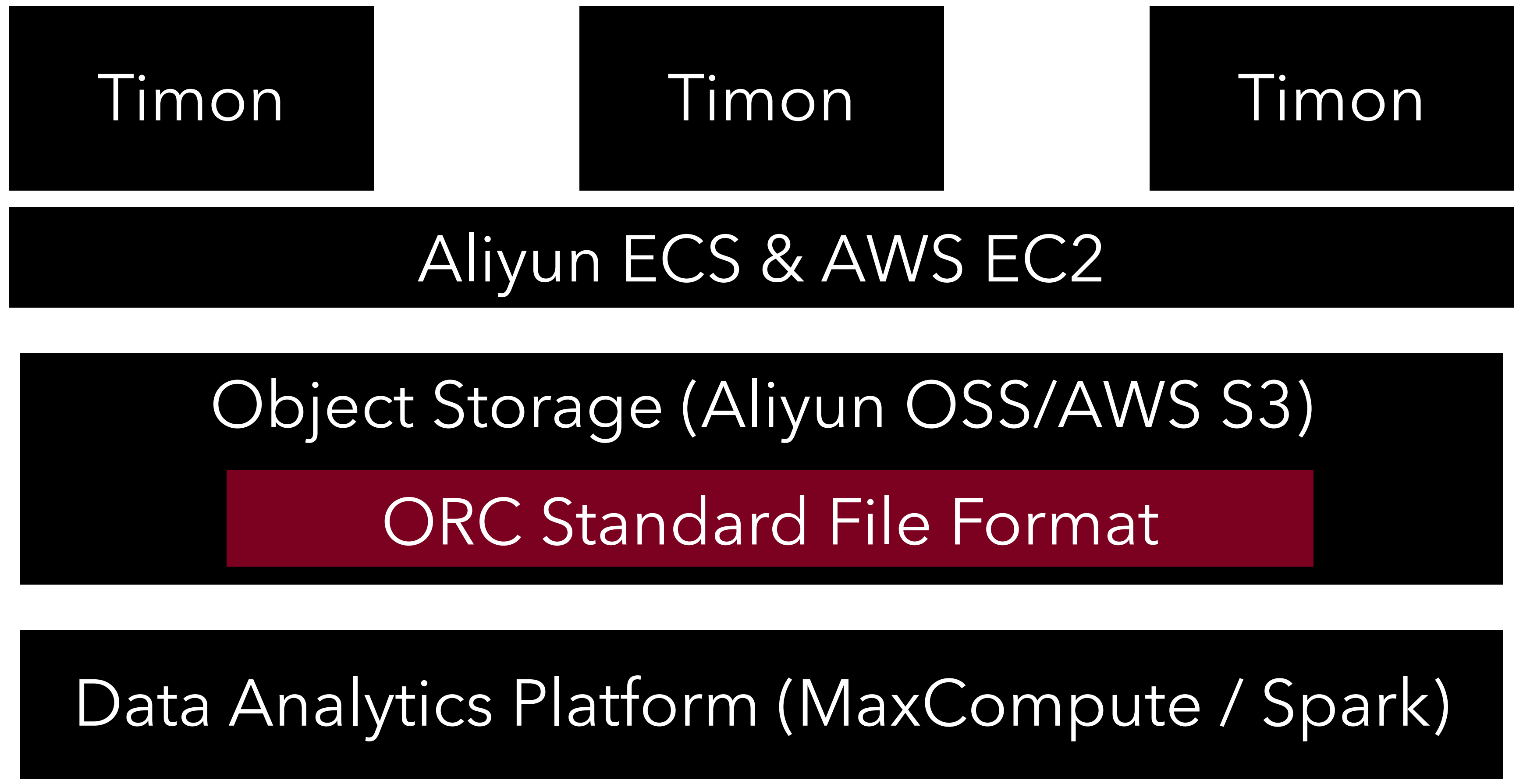
Aliyun ECS & AWS EC2

Aliyun ECS & AWS EC2

Cloud Storage (Aliyun ESSD & AWS EBS)

On Elastic Compute

(Each Timon node sees its own data)



Timon

Timon

Timon

Aliyun ECS & AWS EC2

Object Storage (Aliyun OSS/AWS S3)

ORC Standard File Format

Data Analytics Platform (MaxCompute / Spark)

On Elastic Compute

And Using Object Storage

Thanks

Yusong Gao

jianchuan.gys@alibaba-inc.com