CCF YOCSEF

Shanghai

Big Data Beyond Hadoop

Real-Time Analytical Processing (RTAP) Using Spark and Shark

Jason Dai

Engineering Director & Principal Engineer

Intel Software and Services Group

Agenda

Big Data beyond Hadoop

Introduction to Spark and Shark

Case study: real-time analytical processing (RTAP)

Big Data beyond Hadoop

Big Dta today

- The 🐘 is in the room

Big Data beyond Hadoop

- Real-time analytical processing (RTAP)
  - Discover and explore data iteratively and interactively for **real-time** insights
- Advanced machine leaning and data mining (MLDM)
  - **Graph-parallel** predictive analytics (non-SQL)
- Distributed in-memory analytics
  - Exploit available **main memory** in the entire cluster for >100x speedup

RTAP: Real-Time Analytical Processing

Real-Time Analytical Processing (RTAP)

- Data ingested & processed in a **streaming** fashion
- Real-time data queried and presented in an **online** fashion
- Real-time and history data combined and mined **interactively**
- Predominantly **RAM**-based processing

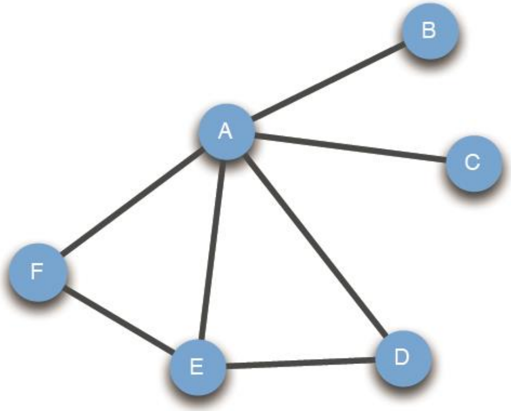
Advanced, Graph-Parallel MLDM

Advanced machine learning and data mining (MLDM)

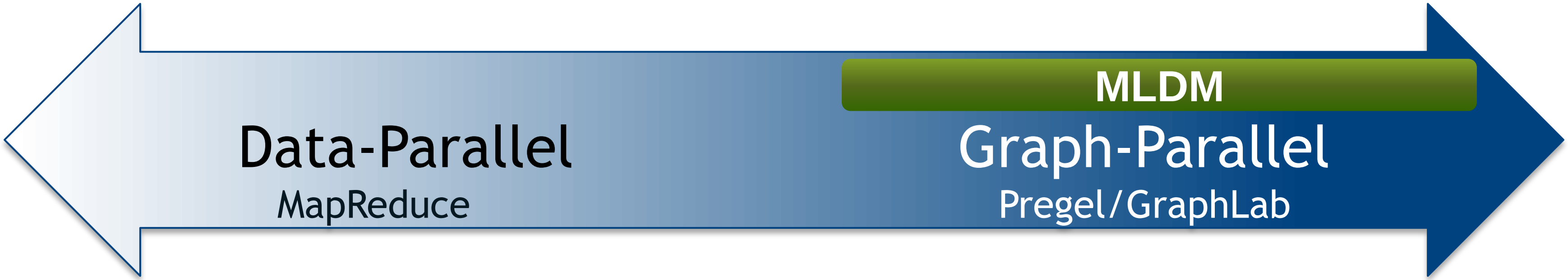
- Information retrieval (e.g., page rank)
- Recommendation engine (e.g., ALS)
- Social network analysis (e.g., clustering)
- Natural language processing (e.g., NER)
- …

Graph parallel computations

- A sparse graph *G(V, E)*
- A vertex program *P* runs on each vertex in parallel & repeatedly
- Vertices interact along edges



Advanced, Graph-Parallel MLDM



Data-Parallel

MapReduce

MLDM

Graph-Parallel

Pregel/GraphLab

- Independent data
- Single-pass
- (Bulk) synchronous

- (Sparse) data dependence
- Iterative
- Dynamically prioritized

10x~100x speedup

- Exploit **graph structure** to reduce computation & communications
- Efficient **graph partition** to balance computation/storage, and minimize network transfer

Distributed In-Memory Analytics

Memory is **king**

- 64GB/node mainstream, 192GB not uncommon, fast cheap NVRAM on the horizon

Hadoop inherently **disk**-based architecture

- Full table scan in Hive from RAM only ~40% speedup
- Read all the main-memory DB literatures ☺

Distributed in-memory analytics

- Efficient compute integrated with columnar compression
- Reliable RAM-oriented storage layer across the cluster
- Holistic allocation of memory in the cluster
  - Inputs, intermediate results, temporary data, computation state, etc.

Agenda

Big Data beyond Hadoop

Introduction to Spark and Shark

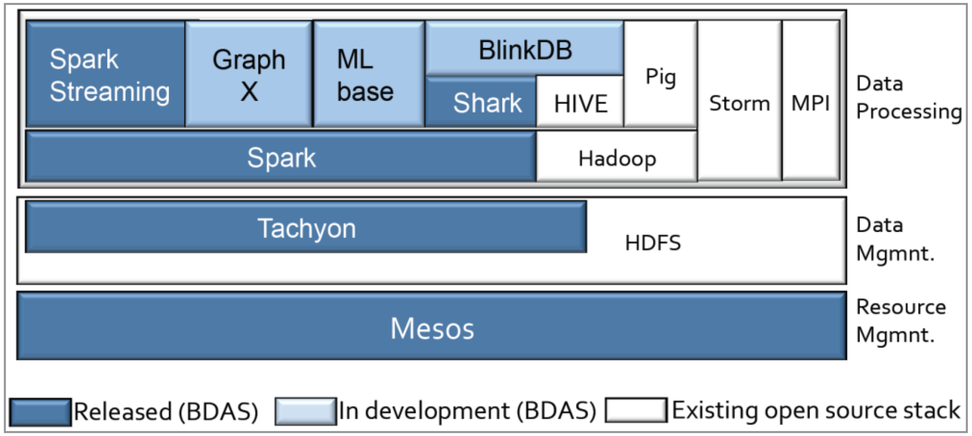
Case study: real-time analytical processing (RTAP)

Project Overview

Research & open source projects initiated by AMPLab in UC Berkeley

- Leveraging existing SW stacks (e.g., HDFS, Hive, etc.)
- Moving beyond Hadoop w/ BDAS
  - In-memory, real-time data analysis (*Spark, Shark, Tachyon, etc.)*
  - Advanced, graph-parallel machine leaning (*GraphX, MLBase*, *etc*.)
- Intel China collaborating with AMPLab on joint open source development
- Active communities and early adopters evolving
  - Spark Apache incubator proposal @ https://wiki.apache.org/incubator/SparkProposal

https://amplab.cs.berkeley.edu/
http://spark-project.org/
http://shark.cs.berkeley.edu/



Spark Streaming
Graph X
ML base
BlinkDB
Shark
HIVE
Pig
Storm
MPI
Data Processing
Spark
Hadoop
Tachyon
HDFS
Data Mgmnt.
Mesos
Resource Mgmnt.

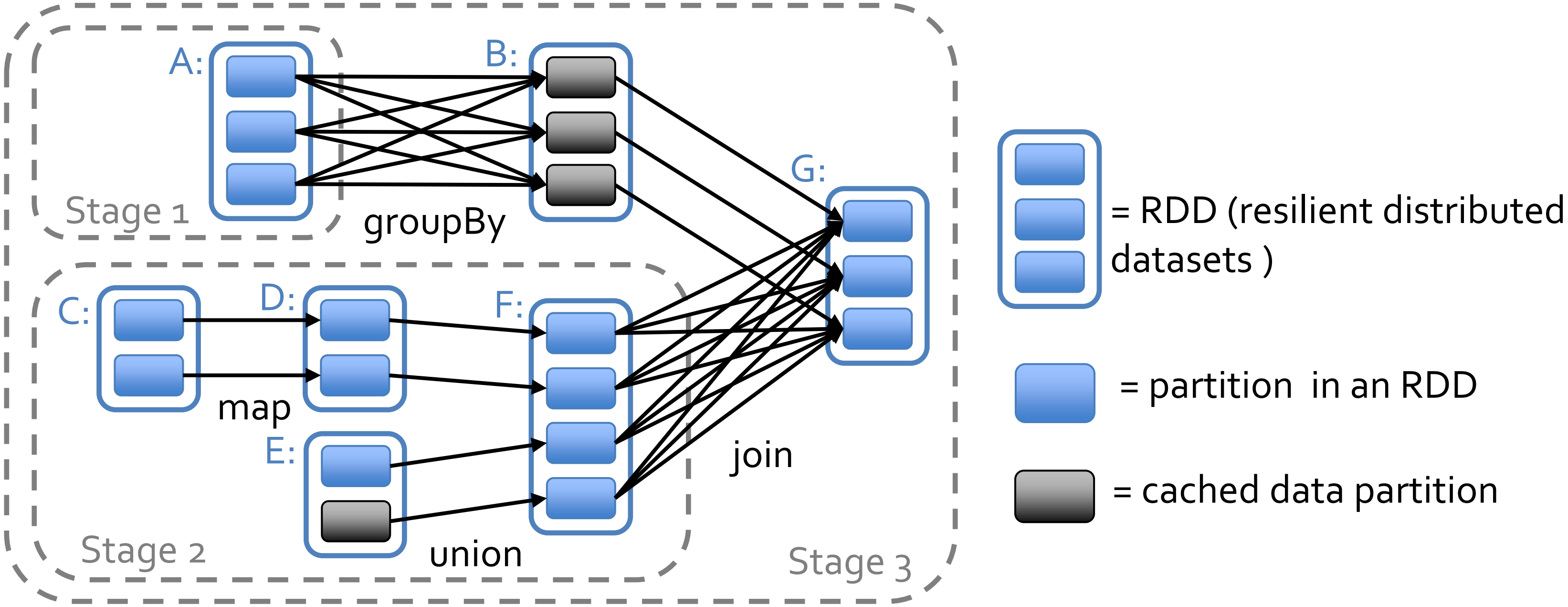
■ Released (BDAS) ▭ In development (BDAS) □ Existing open source stack

Berkeley Data Analytics Stack (BDAS)

What is Spark?

A distributed, *in-memory*, *real-time* data processing framework

- A general, efficient, Dryad-like engine
  - A superset of MapReduce, compatible with Hadoop's storage APIs, but up to 40x faster than Hadoop
  - Avoid launching multiple chained MR jobs or storing intermediate results on HDFS



A: B: G: C: D: F: E:

Stage 1 groupBy

map union join

Stage 2 Stage 3

= RDD (resilient distributed datasets)

🟦 = partition in an RDD

⬛ = cached data partition

What is Spark?

A distributed, *in-memory*, *real-time* data processing framework

- Extremely low latency
  - Optimized for tasks as short as 100s of milliseconds
  - Speed of MPP and/or in-memory databases (i.e., interactive queries), but with finer-grained fault recovery
- Efficient in-memory, real-time computing
  - Allow working set to be cached in memory, with graceful degradation under low memory
  - Efficient support for real-time and/or iterative data analysis
    - Interactive, streaming, iterative, graph-parallel, etc.

What is Shark?

A Hive-compatible data warehouse on Spark

- Compatible with existing Hive data, metastores, and queries (HiveQL, UDFs, etc.)
  - Shark/Spark specific optimizations (hash- and memory-based shuffle, data co-partitioning, etc.)
  - Up to 40x faster than Hive, and support interactive queries
- Allow table to be cached in memory for online & iterative mining
- Integration with Spark to combine SQL and machine learning algorithms

Use Cases

Ad-hoc & interactive queries

- Allow close-to sub-second latency
  - E.g., similar to Dremel & Implala (but with fine-grained fault-tolerance)

In-memory, real-time analysis

- Load data (reliably) in distributed memory for online analysis
  - E.g., similar to PowerDrill

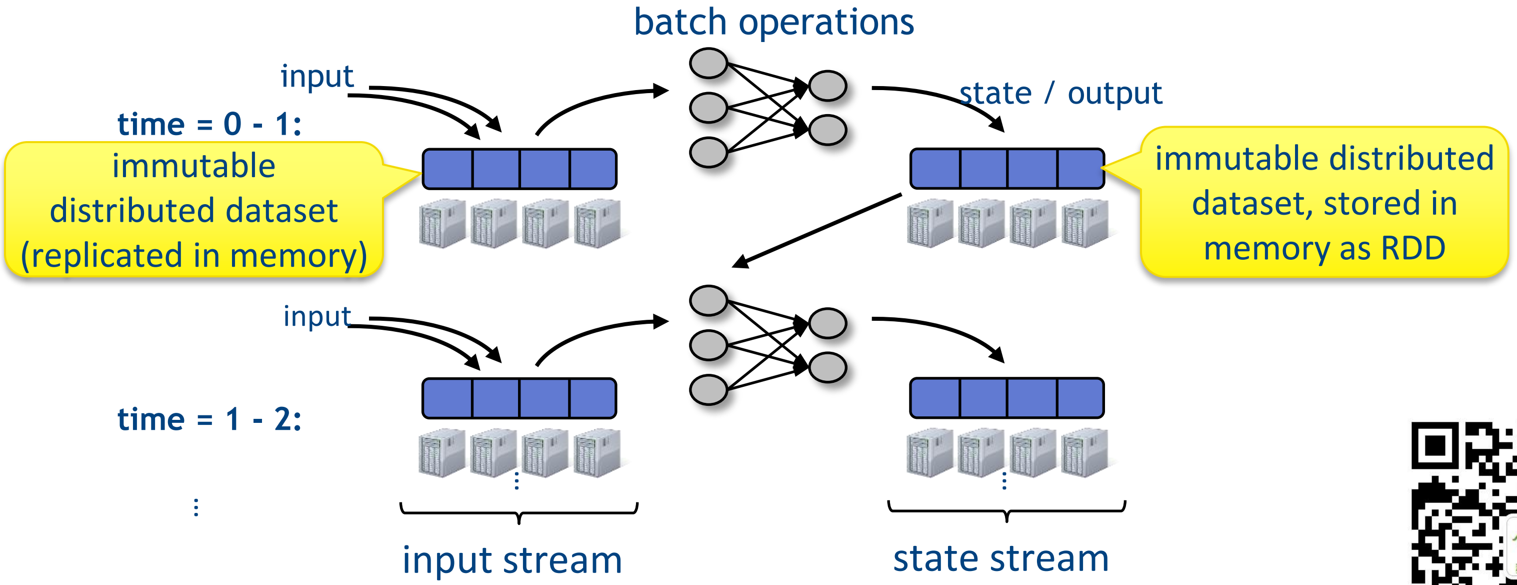
Iterative, graph-parallel analysis (esp. machine learning)

- Cache intermediate results in memory for iterative machine learning
- Graph-parallel computing (e.g., Pregrel and GraphLab models) on Spark

Use Cases

Stream processing

- Spark streaming
  - Run streaming computation as a series of very small, deterministic batch jobs
    - As frequent as ~1/2 second
  - Better fault tolerance, straggler handling & state consistency
  - Potentially combine batch, interactive & streaming workloads


batch operations
input
time = 0 - 1:
immutable distributed dataset (replicated in memory)
state / output
immutable distributed dataset, stored in memory as RDD
input
time = 1 - 2:
⋮
input stream
state stream

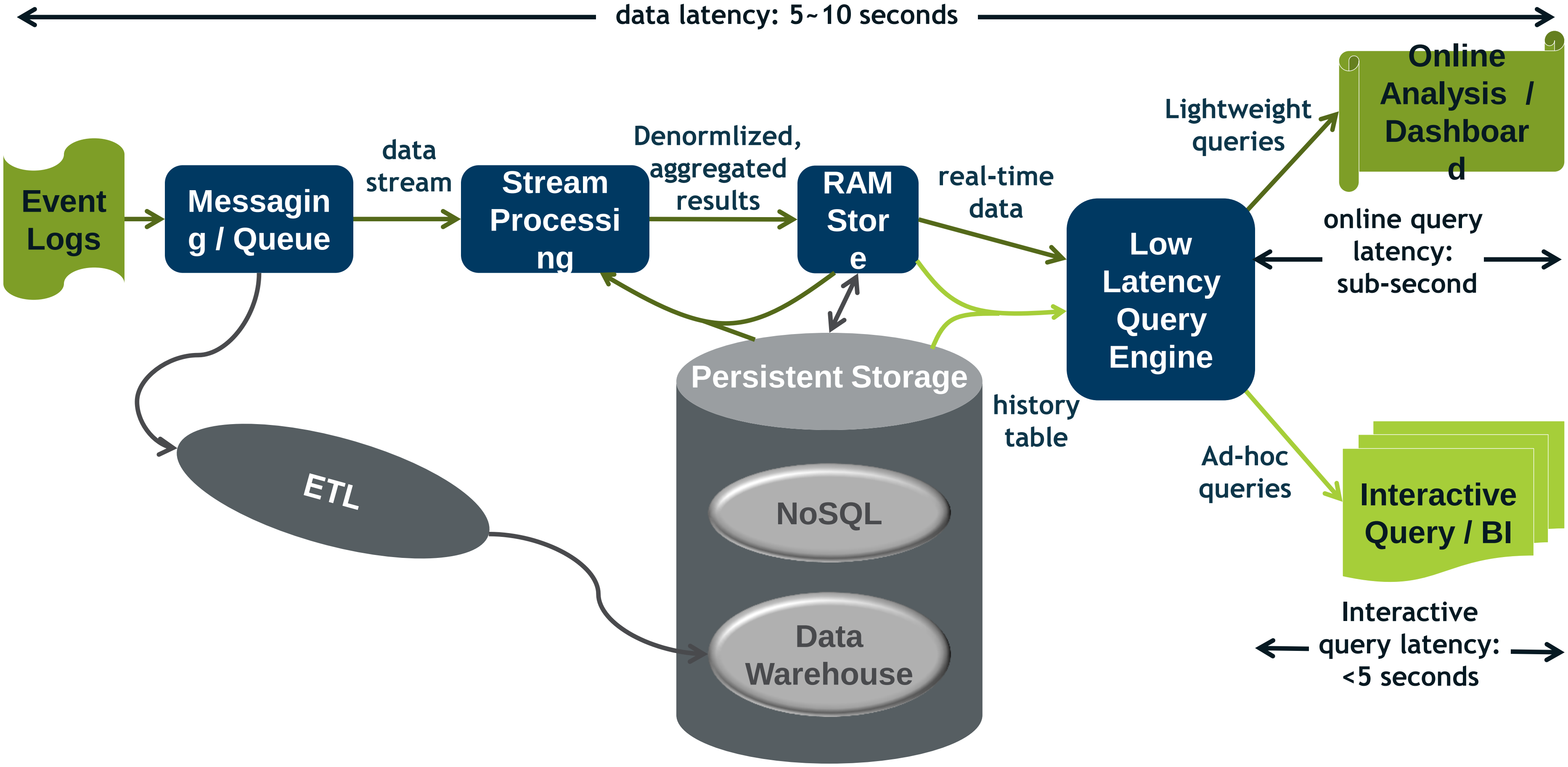
Agenda

Big Data beyond Hadoop

Introduction to Spark and Shark

Case study: real-time analytical processing (RTAP)

RTAP Architecture

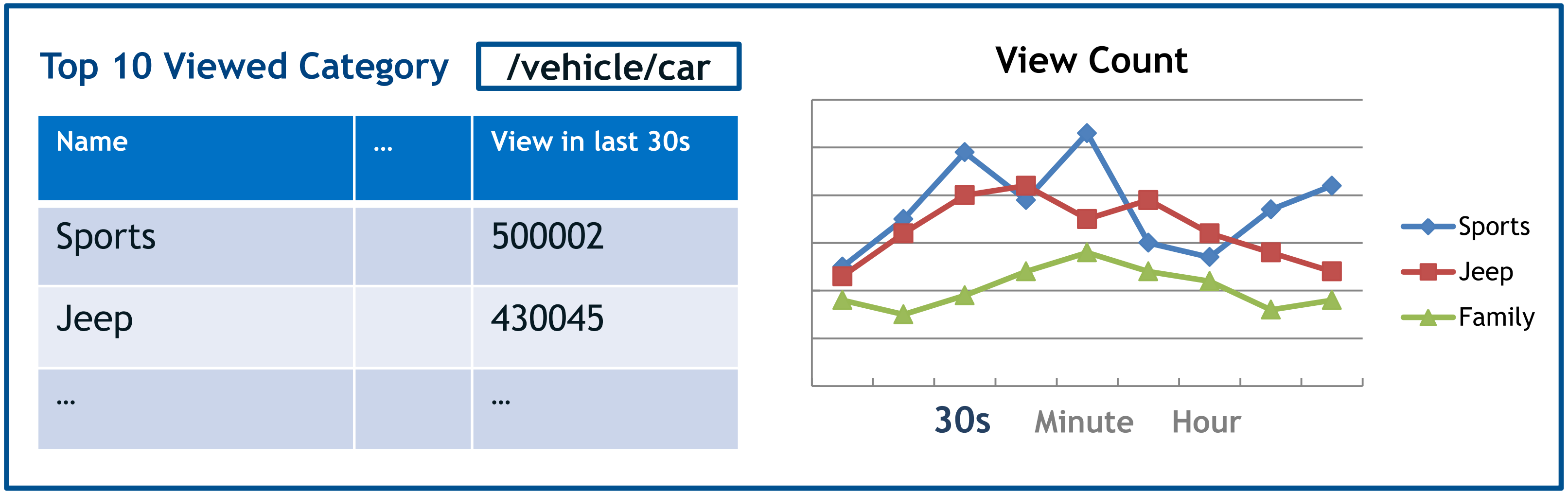


We are partnering with several web sites on building the *RTAP* framework using Spark & Shark

RTAP Use Cases

Online dashboard

- Pages/Ads/Videos/Items – time base aggregations – break-down by categories/demography



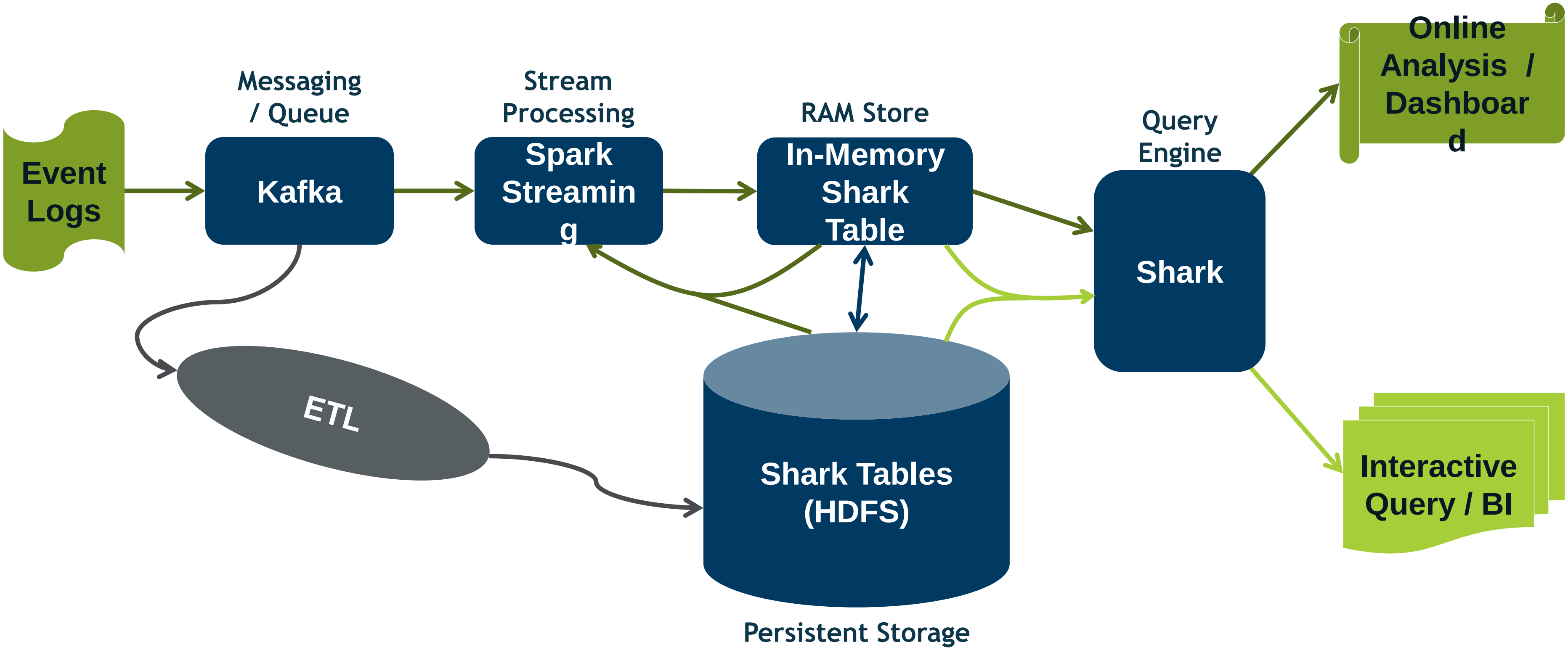
Top 10 Viewed Category /vehicle/car

Name	…	View in last 30s
Sports		500002
Jeep		430045
…		…

Interactive BI

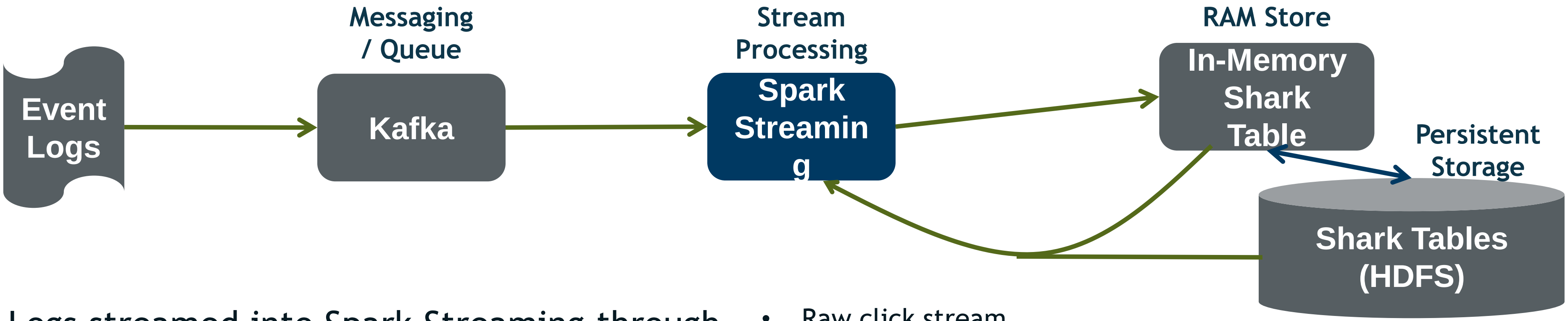
- Combined with history & dimension data when necessary
  - E.g., top 100 viewed videos under each category in the last month

RTAP Framework using Spark & Shark



A work in progress

Real-Time Data Stream Processing



Logs streamed into Spark Streaming through Kafka in real-time

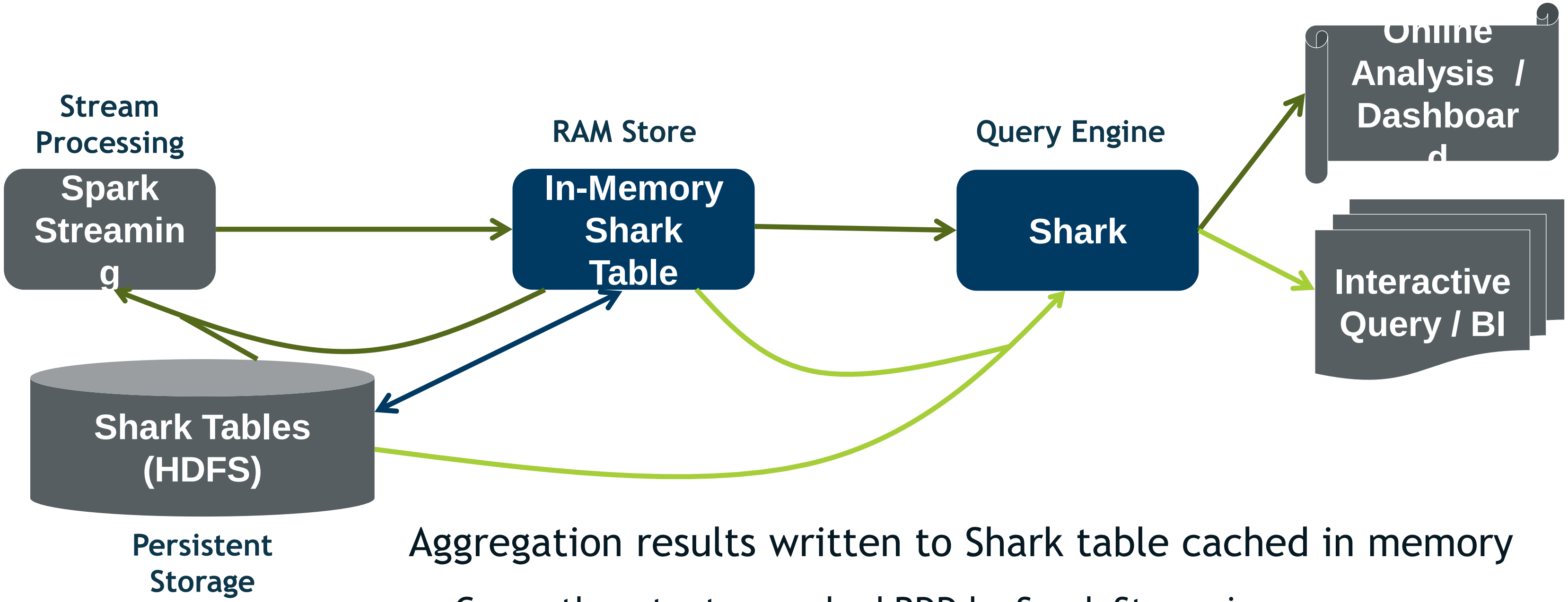
Incoming logs processed by Spark Streaming in small batches (e.g., 5 seconds)

- Compute multiple aggregations over logs received in the last window
- Join logs and history tables when necessary

Plan to add the Streaming support directly in Shark

- Raw click stream
  - *0.6.38.68 - - BAF42487E0C7076CE576FAAB0E1852EC [14/Dec/2012 8:21:16 -0] "GET ?video=8745 HTTP/1.1" 101 1345 http://www.foo.com/bar/?ivideo=8745 "Mozilla/4.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)“*
- Compute page view in the last minute
  - E.g., *www.foo.com/bar/?video=8745, www.foo.com/bar, www.foo.com,* etc.
- Compute category view count in the last minute
  - E.g., join logs and the video table (assuming *video 8745* belongs to */vehicle/car/sports*) for */vehicle, /vehicle/car, /vehicle/car/sports*, etc.

Real-Time Data Store and Query Engine



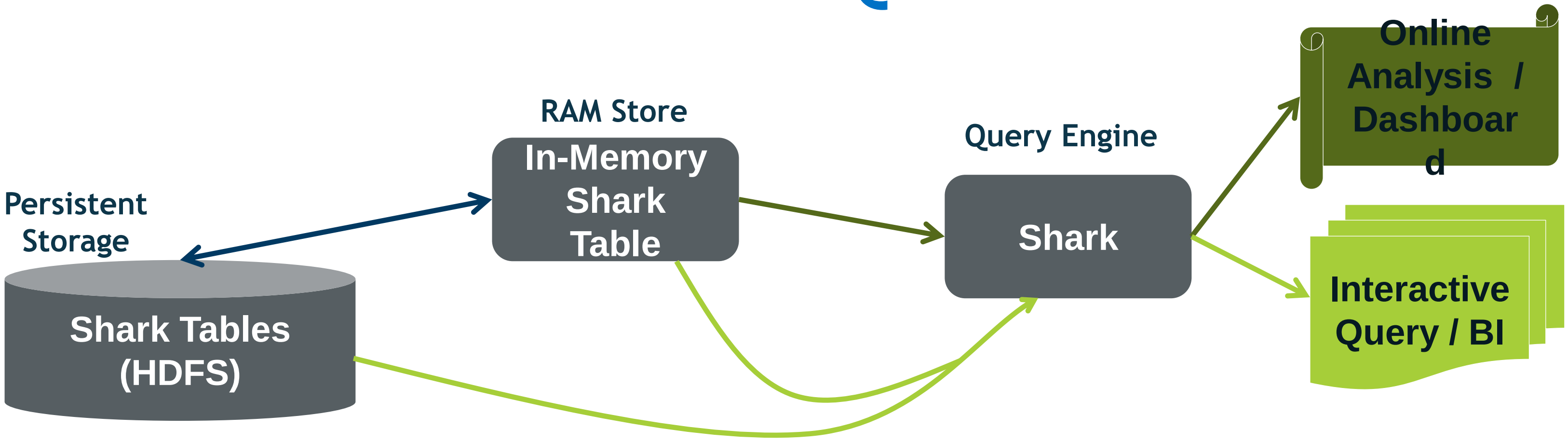
Aggregation results written to Shark table cached in memory

- Currently output as cached RDD by Spark Streaming
  - Require Spark Streaming embedded in the Shark server JVM
- Plan to move to Tachyon for better sharing and fault tolerance

Both real-time aggregations and history data queried through Shark

- History data loaded into memory for iterative mining
- Working on query optimizations & standard SQl-92 support

Online and Interactive Queries



Online analysis

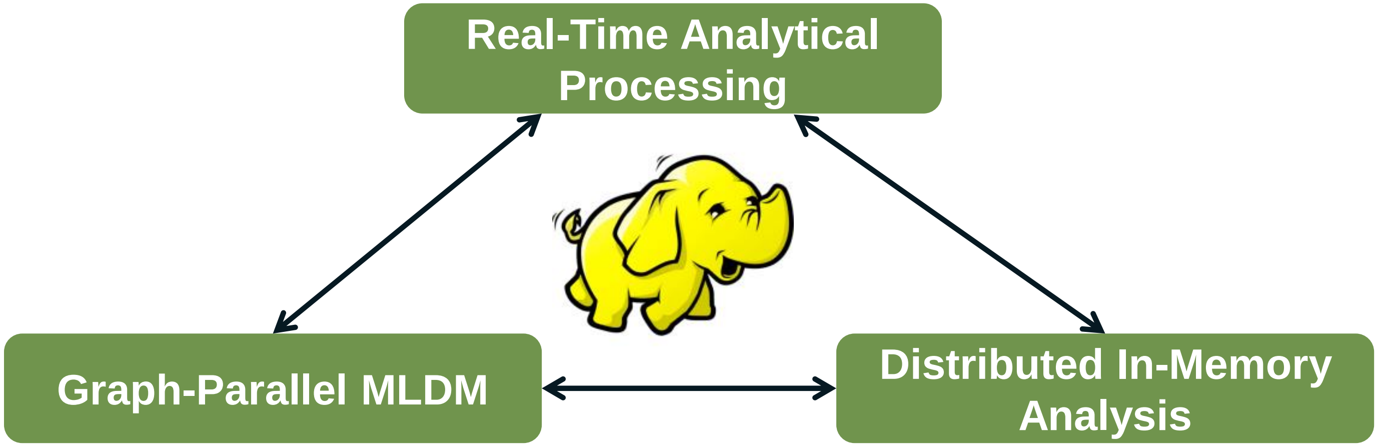
- A lightweight UI frontending Shark for online dashboard
- Mostly time-based lightweight queries (filtering, ordering, TopN, aggregations, etc.) with sub-second latency

Interactive query / BI

- Ad-hoc, (more) complex SQL queries (with <5 seconds latency)
- Heavily denormalized to eliminate join as much as possible

Summary

1 Big Data **beyond** Hadoop



Real-Time Analytical Processing

Graph-Parallel MLDM

Distributed In-Memory Analysis

2 BDAS: one stack to **rule** them all!

Intel China collaborating with UC Berkeley & web sites on production deployment

Active communities and early adopters evolving (e.g., Spark Apache incubator proposal)

3 Call to action

Work with us on next-gen Big Data beyond Hadoop using Spark/Shark

2013英特尔® 软件学院课程概览

2013英特尔® 软件学院课程图

英特尔® 平台并行程序设计

- 高级 — 基于英特尔® 集成众核架构的编程和优化
- 高级 – 针对多核微架构的优化
- 中级 — 使用工具进行并行程序优化设计
- 初级 — 并行编程基础

移动互联设备与嵌入式系统

- 基于超极本™ 和平板的Windows* 8 应用开发
- 基于英特尔® 平台的Android* 应用开发
- HTML5 移动应用开发
- 基于英特尔® 凌动™ 平台的嵌入式开发应用

数据中心与云计算

- 英特尔® Apache Hadoop* 软件发行版的安装，运营和管理
- 基于英特尔® Apache Hadoop* 软件发行版的大数据应用开发
- 基于英特尔® 平台的企业云计算架构设计
- 基于英特尔® 平台的分布式存储架构设计与调优
- 高性能计算–集群搭建和应用调试

英特尔® 平台技术

- 英特尔® 可视化计算应用开发和调优
- 针对英特尔® 核芯显卡优化3D游戏客户端性能
- 英特尔® 功耗优化策略和工具
- 基于英特尔® 平台的感知计算应用开发

管理与软技能

- 软件质量控制
- 软件产品测试
- 软件项目管理基础
- 建立战略合作伙伴
- 销售基础
- 演讲与沟通技巧
- 问题解决技巧

英特尔计划于9月举办大数据师资研讨活动，有兴趣参与的老师请联系：hai.shen@intel.com intel®