BDTC

2013 中国大数据技术大会

Big Data Technology Conference 2013

Spark: High-Speed Big Data Analysis Framework

Intel

Andrew Xia

Weibo:Andrew-Xia

Agenda

- ***Intel contributions to Spark***
- Collaboration
- Real world cases
- Summary

Spark Overview

- Open source projects initiated by AMPLab in UC Berkeley

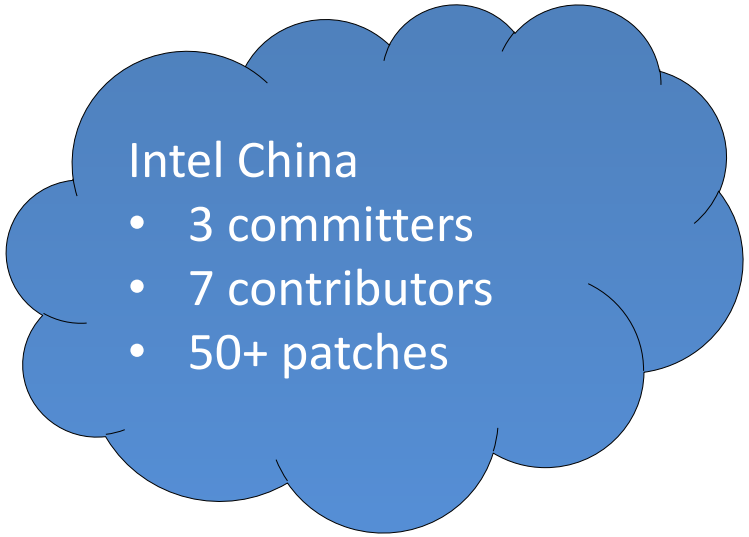
- Apache incubation since June 2013

- Intel closely collaborating with AMPLab & the community on open source development

Contributions by Intel

- Netty based shuffle for Spark
- FairScheduler for Spark
- Spark job log files
- Metrics system for Spark
- Spark shell on YARN
- Spark (standalone mode) integration with security Hadoop
- Byte code generation for Shark
- Co-partitioned join in Shark

. . .



Agenda

- Intel contributions to Spark
- ***Collaboration***
- Real world cases
- Summary

Collaboration Partners

- Intel partnering with several big websites
  - Building next-gen big data analytics using the Spark stack
  - E.g., Alibaba , Baidu iQiyi, Youku, etc.

阿里巴巴 Alibaba.com

YOUKU 优酷

iQIYI 爱奇艺
悦 享 品 质

Big Data in Partners

- Advertising
  - Operation analysis
  - Effect analysis
  - Directional optimization
- Analysis
  - Website report
  - Platform report
  - Monitor system
- Recommendation
  - Ranking list
  - Personal recommendation
  - Hot-click analysis

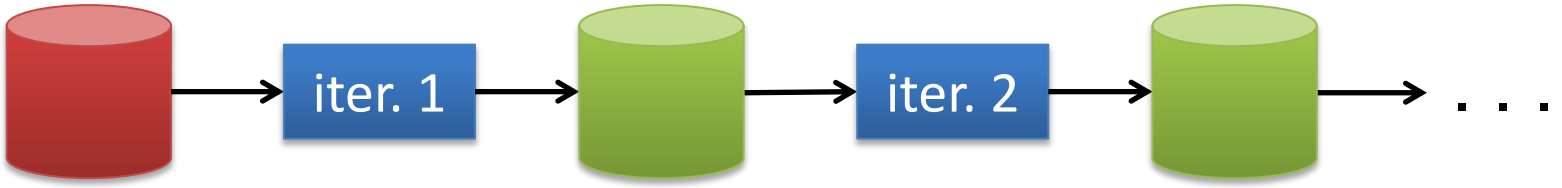
FAQs

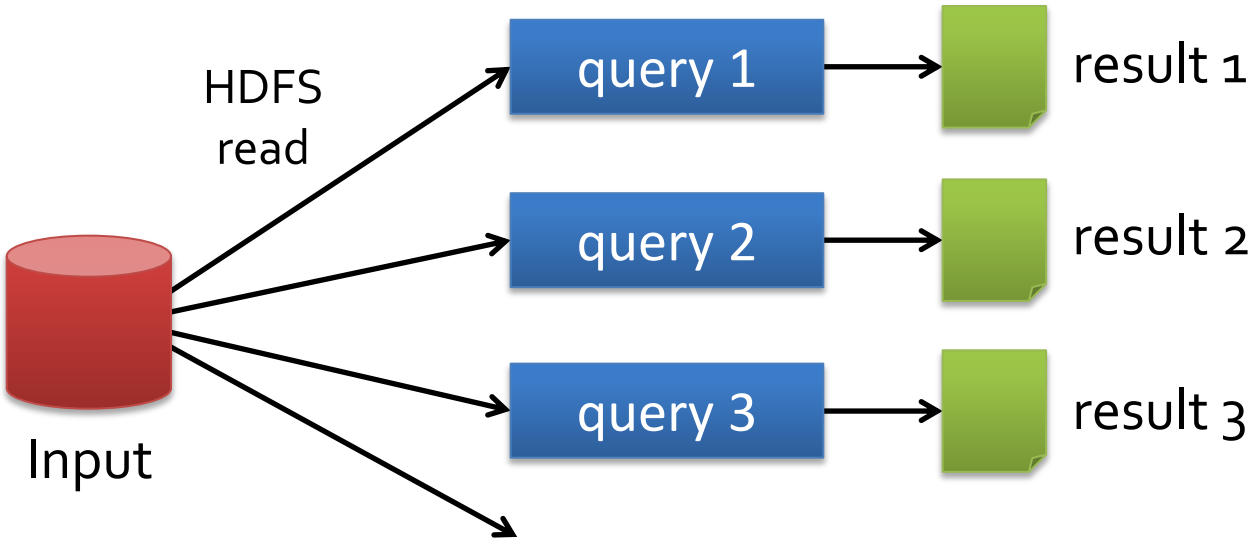
FAQ #1: Poor Performance

– machine learning and graph computation

– OLAP for Tabular data, interactive query

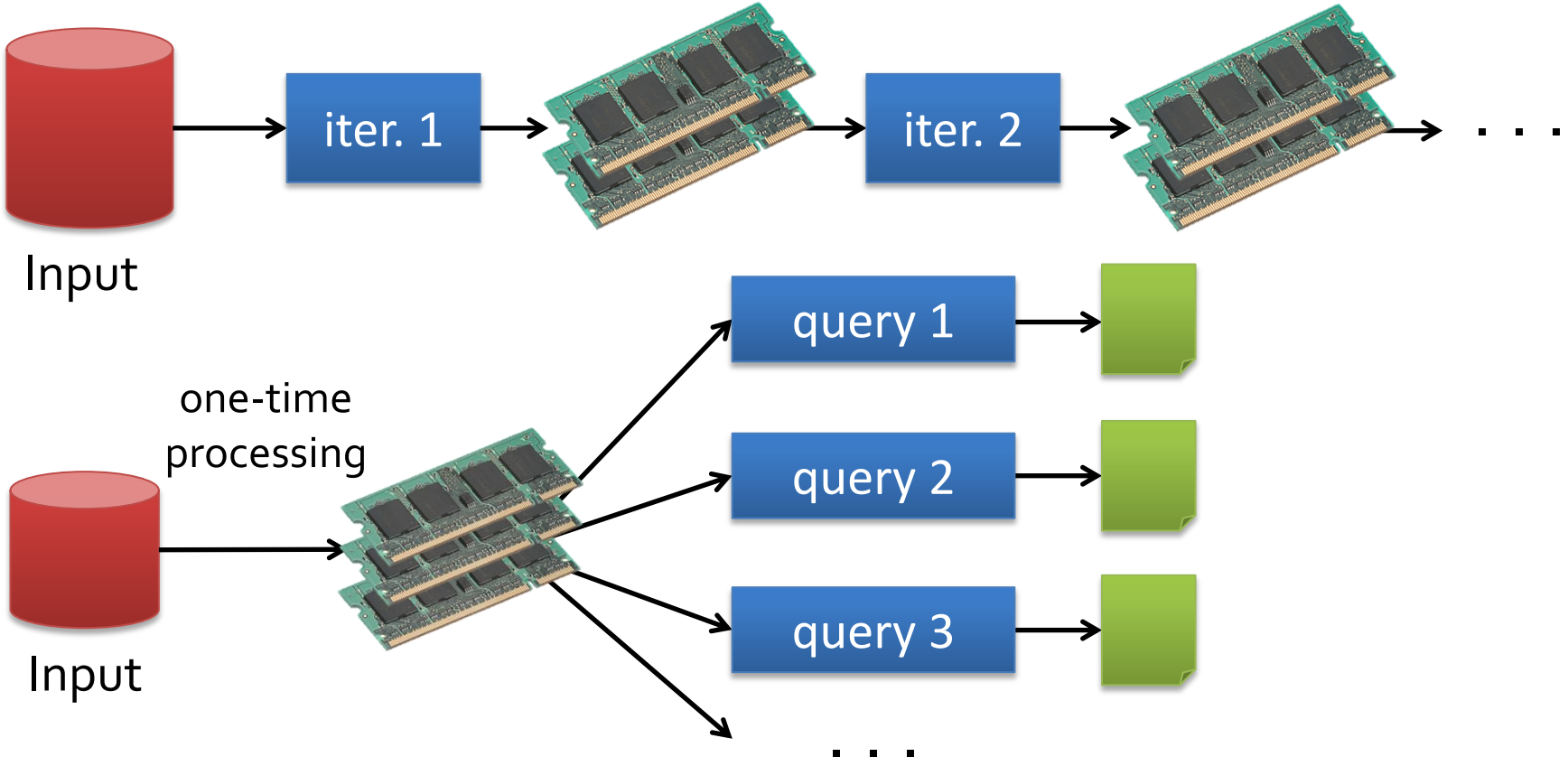
Hadoop Data Sharing





Slow due to replication, serialization, and disk IO

Spark Data Sharing

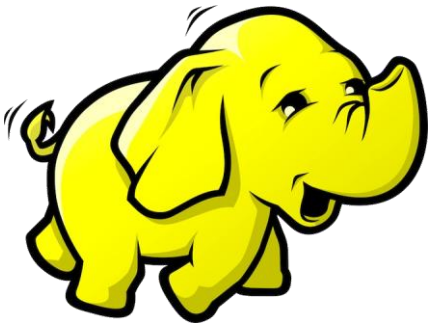


10-100× faster than network and disk

FAQs

FAQ #2: Too many big data systems

Big Data Systems Today

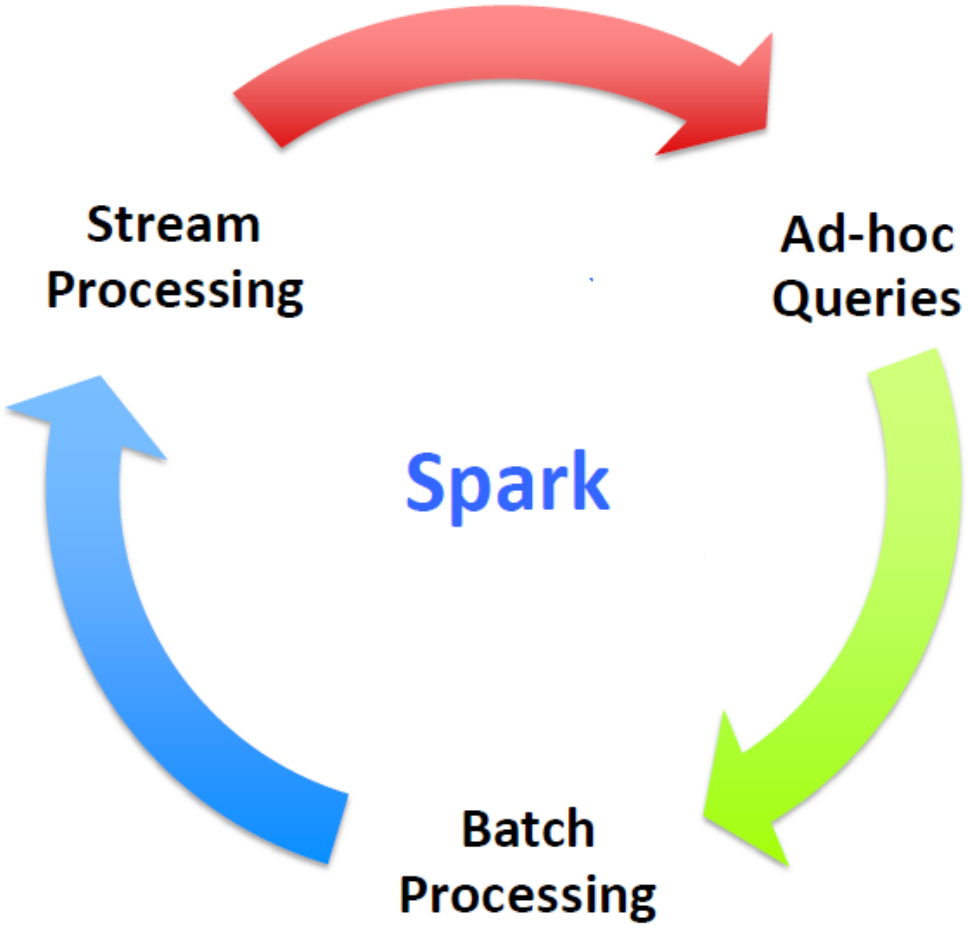


MapReduce

General batch processing

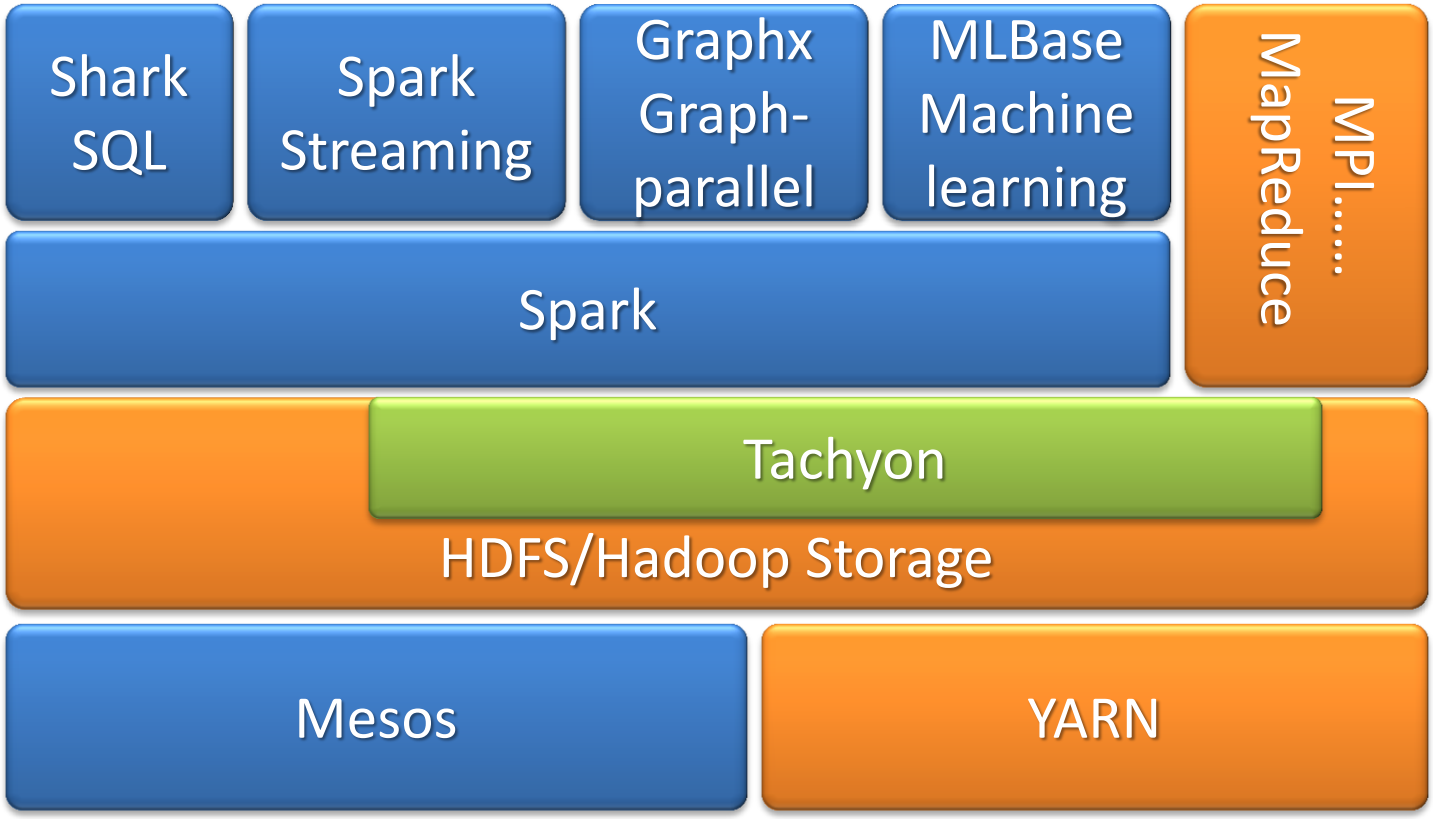
Specialized systems (iterative, interactive and streaming apps)

Vision of Spark Ecosystem



One stack to rule them all!

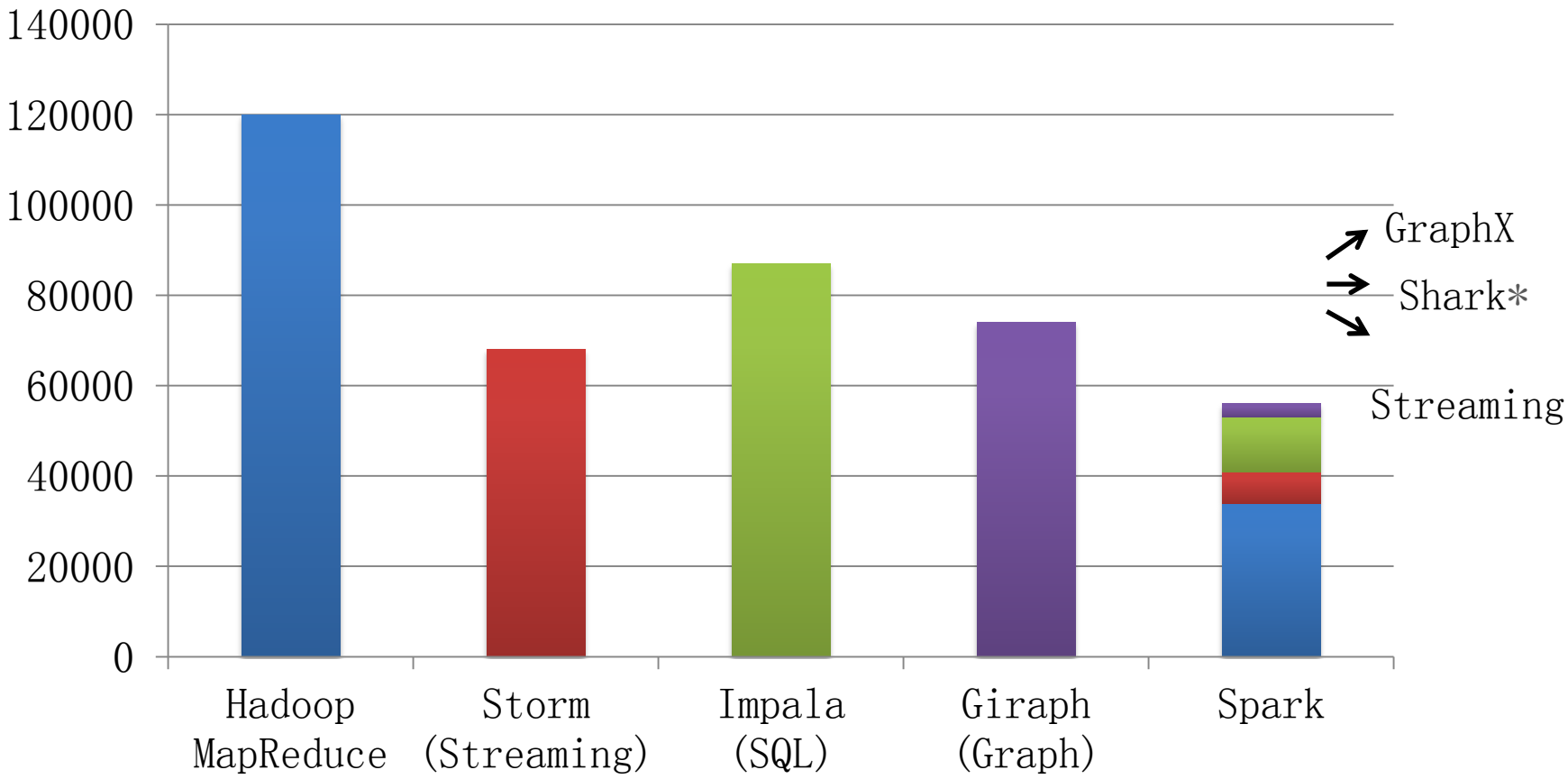
Spark Ecosystem



FAQs

FAQ #3: Study cost

Code Size



non-test, non-example source lines

* also calls into Hiv

FAQs

FAQ #4: Is Spark Stable?

Spark Status

- Spark 0.8 has been released
- Spark 0.9 will be release in Jan 2013

FAQs

FAQ #5: Not enough memory to cache

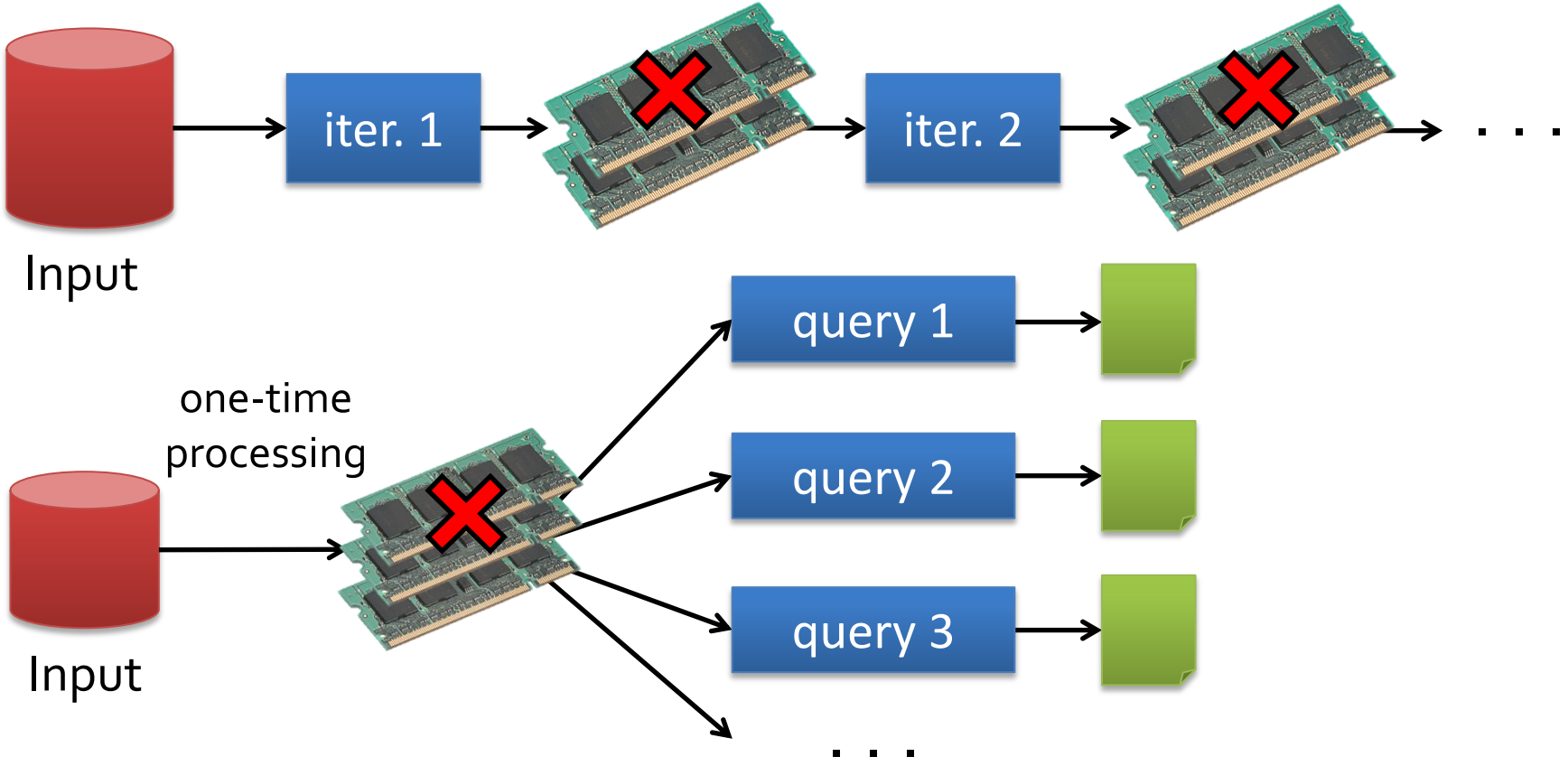
Not Enough Memory

- Graceful degradation
- Scheduler takes care of this
- Other options
  - MEMORY_ONLY
  - MEMORY_ONLY_SER
  - MEMORY_AND_DISK
  - DISK_ONLY

FAQs

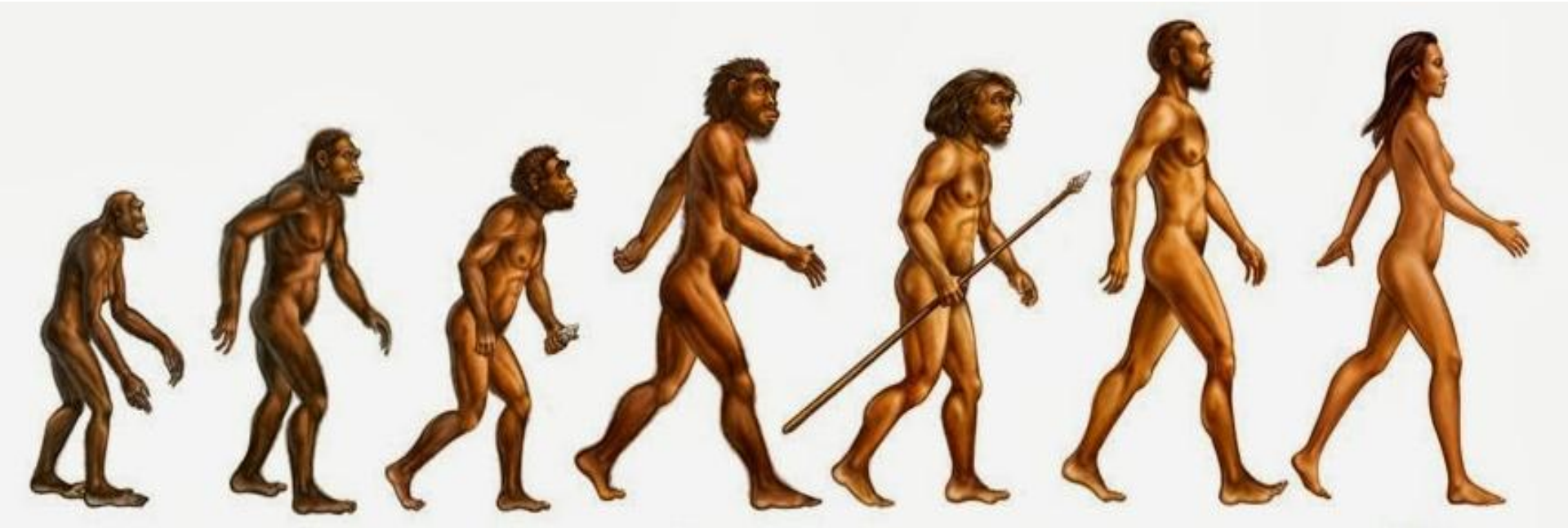
FAQ #6: How to recover when failure?

How to Failover?



How to Failover?

- Lineage*:* track the graph of transformations that built RDD
- Checkpoint: lineage graphs get large



FAQs

FAQ #7: Is Spark compatible with Hadoop ecosystem?

FAQs

FAQ #8:Need port to Spark?

FAQs

FAQ #9: Any cons about Spark?

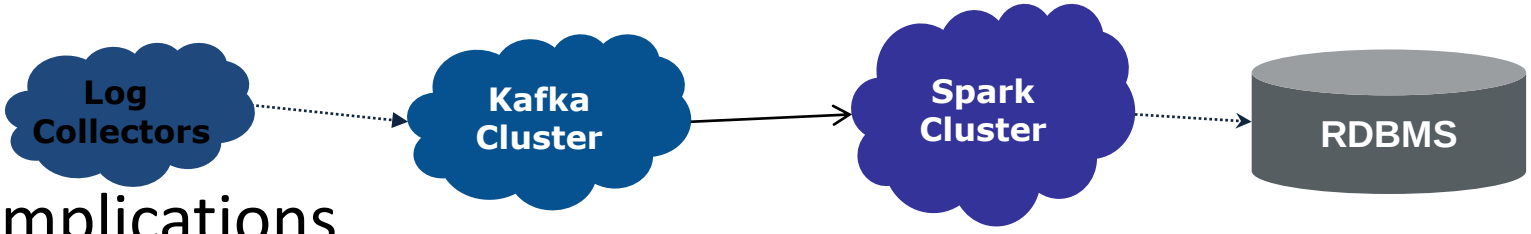
Agenda

- Intel contributions to Spark
- Collaboration
- ***Real world cases***
- Summary

Case1#:Real-Time Log Aggregation

- Logs continuously collected & streamed in
  - Through queuing/messaging systems
- Incoming logs processed in a (semi) streaming fashion
  - Aggregations for different time periods, demographics, etc.
  - Join logs and history tables when necessary
- Aggregation results then consumed in a (semi) streaming fashion
  - Monitoring, alerting, etc.

Real-Time Log Aggregation: Spark Streaming



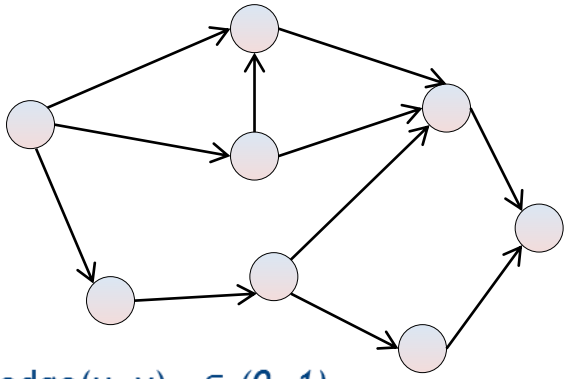
- Implications
  - Better streaming framework support
    - Complex (e.g., statful) analysis, fault-tolerance, etc.
  - Kafka & Spark not collocated
    - DStream retrieves logs in background (over network) and caches blocks in memory
  - Memory tuning to reduce GC is critical
    - `spark.cleaner.ttl` (throughput * spark.cleaner.ttl < spark mem free size)
    - Storage level (`MEMORY_ONLY_SER2`)
  - Lower latency (several seconds)
    - No startup overhead (reusing `SparkContext`)

Case #2:Machine Learning & Graph Analysis

- Algorithm: complex match operations
  - Mostly matrix based
    - Multiplication, factorization, etc.
  - Sometime graph-based
    - E.g., sparse matrix
- Iterative computations
  - Matrix (graph) cached in memory across iterations

Graph Analysis: N-Degree Association

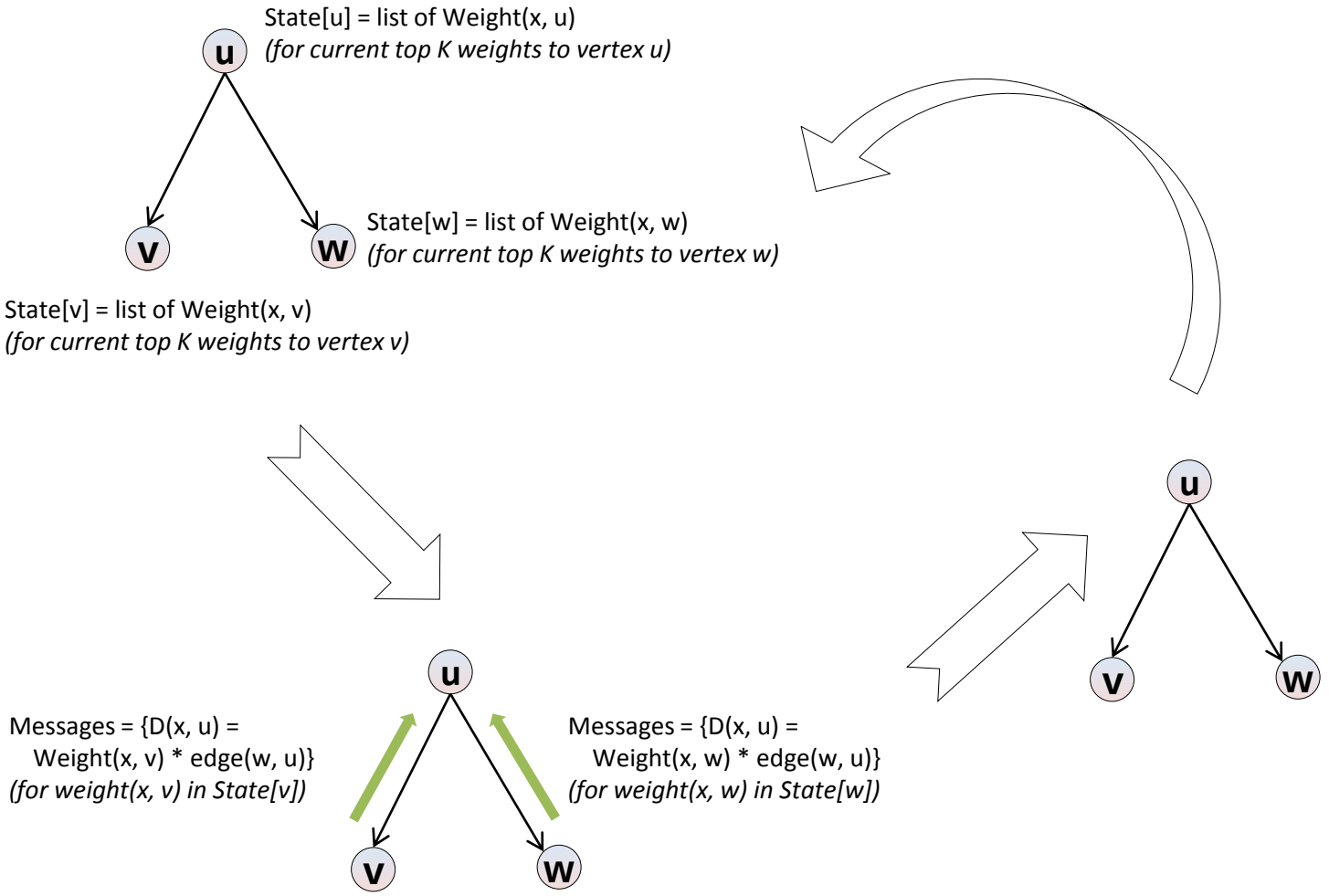
- N-degree association in the graph
  - Computing associations between two vertices that are *n-hop* away
  - E.g., friends of friend



$$Weight_1(u, v) = edge(u, v) \quad \in (0, 1)$$
$$Weight_n(u, v) = \sum_{x \to v} Weight_{n-1}(u, x) * Weight_1(x, v)$$

- Graph-parallel implementation
  - Bagel (Pregel on Spark) and GraphX
    - Memory optimizations for efficient graph caching critical
  - Speedup from 20+ minutes to <2 minutes

Graph Analysis: N-Degree Association

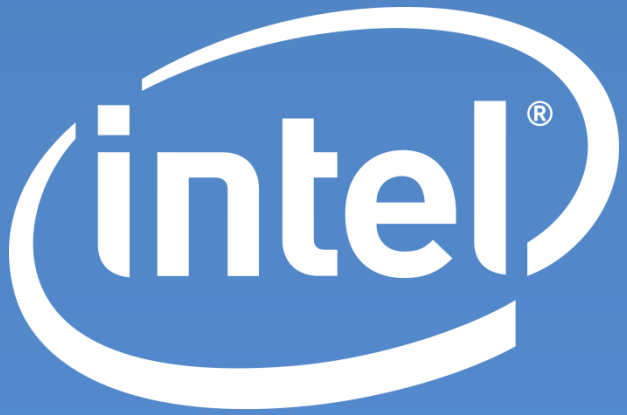


Agenda

- Intel contributions to Spark
- Collaboration
- Real world cases
- ***Summary***

Summary

- ***Memory is King!***
- ***One stack to rule them all!***
- ***Contribute to community!***


intel