ISTM 6212 - Week 11
Spark Introduction

Daniel Chudnov, dchud@gwu.edu

2016-11-22

Agenda

- Schedule check
- Project 02 + Reviews / Exercise 05
- Spark - Background and Setup
- Spark - Walkthrough
- Lambda Architecture
- Project 03 work session

Schedule check

Project 02 + Reviews / Exercise 05

Project 02 example

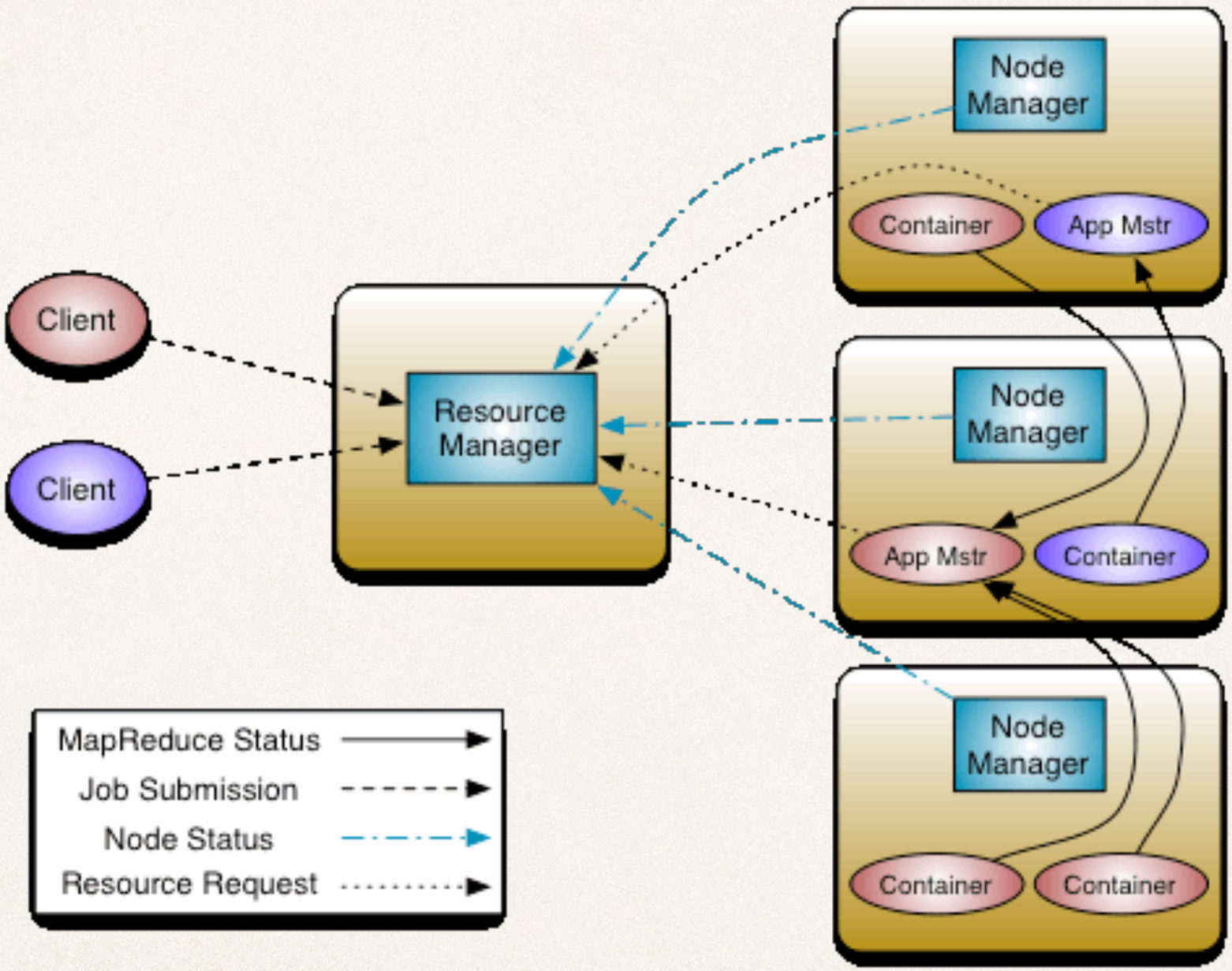
- Amit and Jon's deliberate, thorough schema construction in Problem 3
  - github.com/jdh8v/istm-6212/blob/master/Project-02/project-02_hurwitz_talapatra_final.ipynb

Spark Background

"Apache Spark is a fast and general-purpose cluster computing system."

" Spark uses Hadoop's client libraries for HDFS and YARN."

spark.apache.org/docs/latest/index.html



hadoop.apache.org/docs/stable2/hadoop-yarn/hadoop-yarn-site/YARN.html

"Map/Reduce, but faster."

–me

How much faster?

- "Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk."
- "It has been used to sort 100 TB of data 3X faster than Hadoop MapReduce on 1/10th of the machines"

spark.apache.org

Map / Reduce

- Dean and Ghemawat, 2004
- Simplified programming model
- Flexible data model
- Optimized computing model

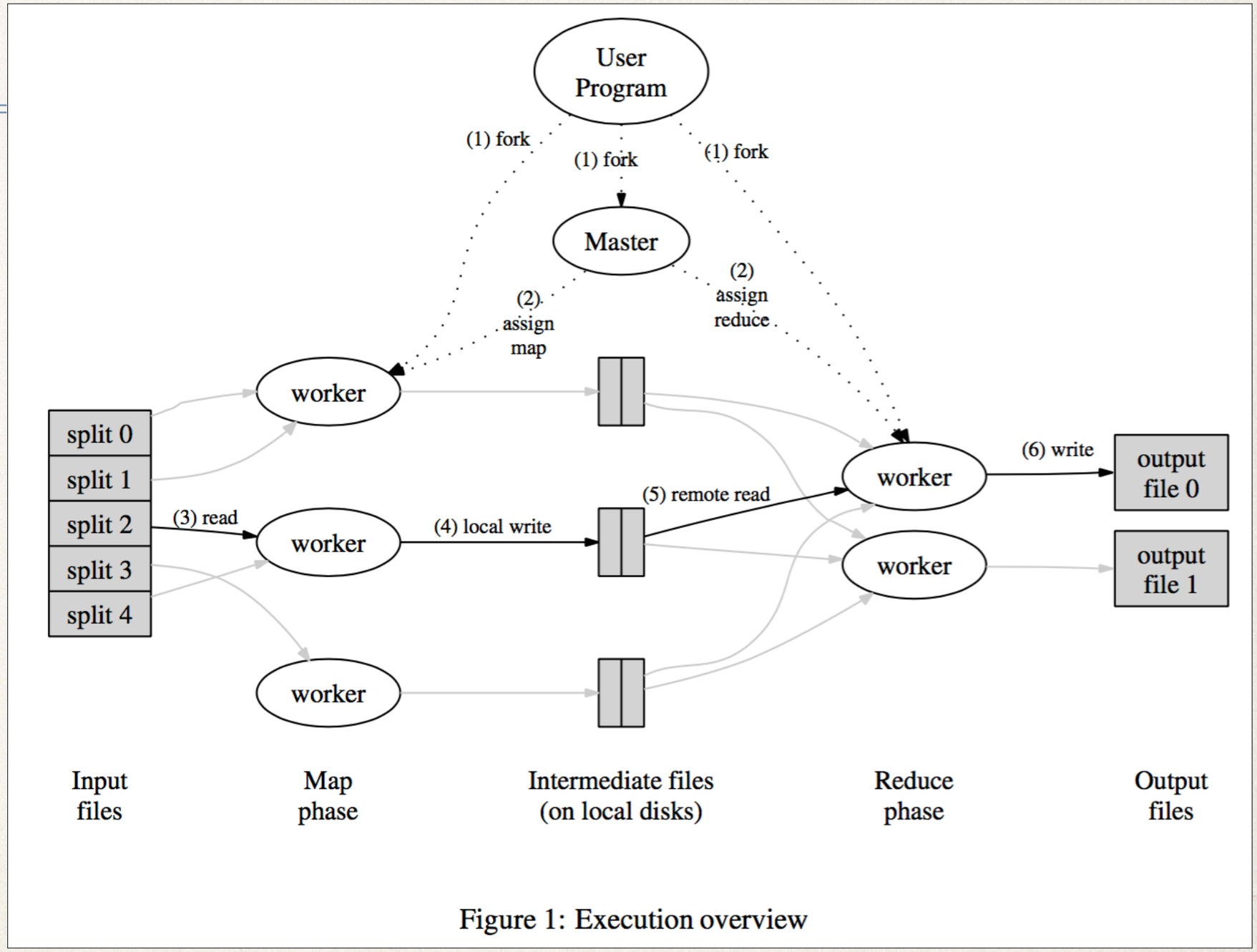


Figure 1: Execution overview

Hadoop + HDFS = Disk read/write

- Results from each intermediate step move to disk
- Disk storage is redundant, so extra copies, extra tracking overhead, extra time
- In the meantime, memory grew cheaper

Summary Metrics for 1878 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	8 ms	11 ms	14 ms	20 ms	0.4 s
GC Time	0 ms	0 ms	0 ms	0 ms	25 ms
Input Size / Records	51.3 KB / 1758	71.1 KB / 5255	77.9 KB / 6395	86.3 KB / 7197	2.3 MB / 14228
Shuffle Write Size / Records	42.0 B / 1	42.0 B / 1	42.0 B / 1	42.0 B / 1	42.0 B / 1

Why so fast?

- Improved data flow between tasks

- More use of memory; less use of disk

- "Resilient distributed datasets" (RDDs) recover from failure automatically

spark.apache.org/research.html

Why so popular?

- Speed
- Ease of use - full support in Scala, Java, Python, R
- Easy to set up and test, then scale on cluster
- AWS EMR, Google Cloud Dataproc, Azure HDInsight

Spark setup

- Download from spark.apache.org/downloads.html
  - (latest release, "Pre-built for" latest Hadoop)
  - With Python and Jupyter already installed in your Ubuntu VM:

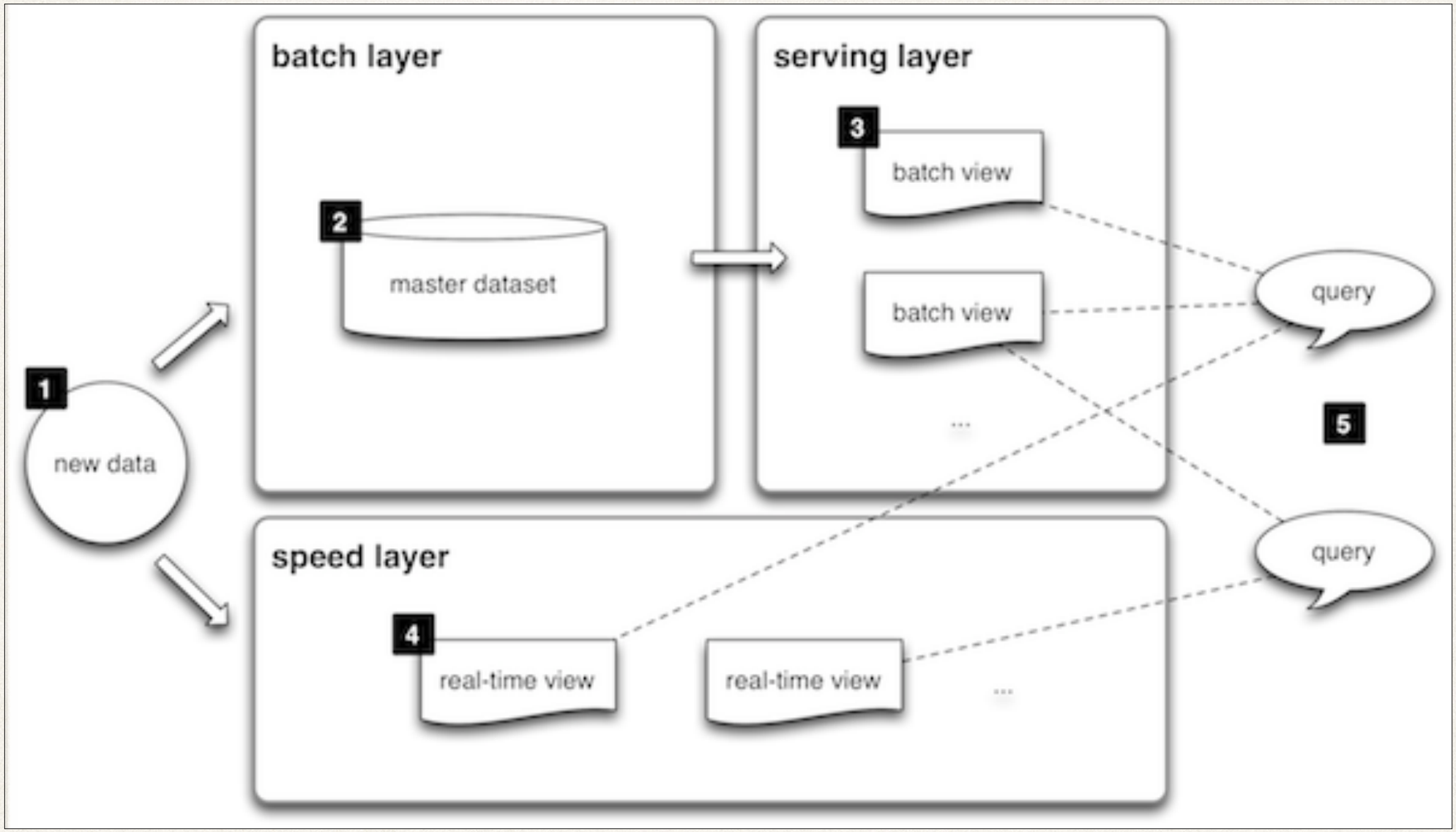
```
% sudo apt-get install openjdk-9-jre-headless
% tar xvzf spark-2.0.2-bin-hadoop2.7.tgz
% export PATH=`pwd`/spark-2.0.2-bin-hadoop2.7/bin:$PATH
% export PYSPARK_DRIVER_PYTHON=jupyter
% export PYSPARK_DRIVER_PYTHON_OPTS='notebook' pyspark
% pyspark
```

Spark - Walkthrough

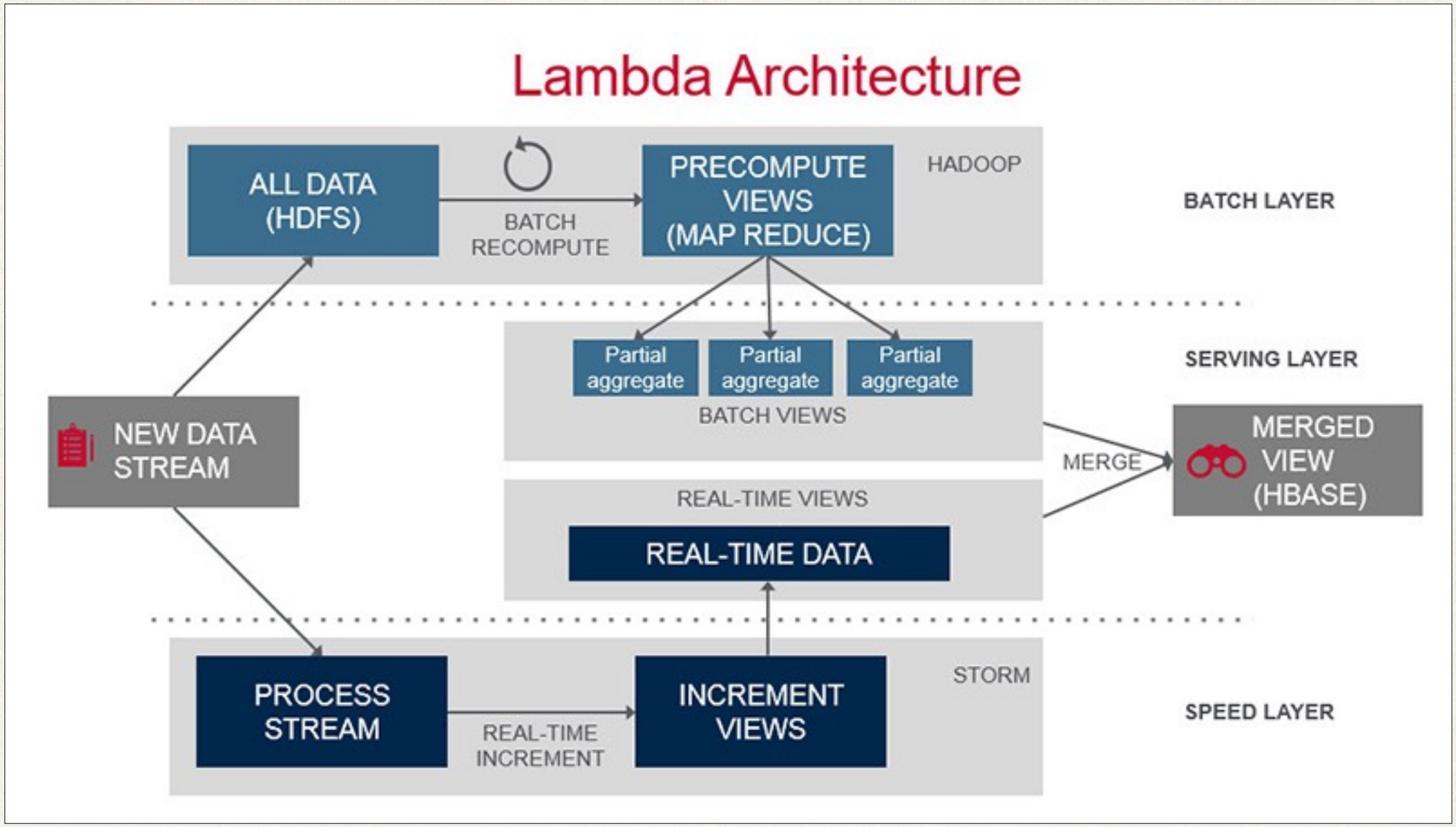
Lambda Architecture

Lambda Architecture - Overview

- Framework for scaling distributed data processing for heavy real-time needs
- Fault-tolerance, horizontal scaling
- New data moves through two layers: batch and speed



lambda-architecture.net



www.mapr.com/developercentral/lambda-architecture

Lambda advantages

- Clearly defined “master” dataset or “source of truth” in batch layer
- Speed layer handles real-time data responsiveness
- User experience supported by merged data from both batch and speed layers
- Good, rapidly improving tools for all these pieces

Lambda disadvantages

- Redundancy - two paradigms/platforms for modeling, query, development, maintenance
- Complexity - requires careful design to enable debugging and other troubleshooting
- Rapid improvements mean constant versioning, testing, deployment overhead

Project 03 work session