



Introduction to Apache Spark

1TD169 - Data Engineering I

Usama Zafar

2025-02-07



UPPSALA
UNIVERSITET

Spark Lecture 1

- ❖ What is Apache Spark? What is it for?
- ❖ Why was Spark developed?
- ❖ How does Spark work? How is MapReduce Implemented?
- ❖ Demos
- ❖ *Spark Internal Mechanics → In Lecture 2*
- ❖ *New frameworks → In Lecture 2*

Recap: map(..) & reduce(..)

- map(..) and reduce(..) are built into many programming languages.
- They can be used independently of Hadoop or Apache Spark.
- Being able to write a program using map(..) and reduce(..) is one way a program can be scaled up for large datasets.
- A key skill to learning is to ‘think’ in terms of map and reduce steps – *the MapReduce programming model* – this is a transferable skill, independent of the particular framework we are using.
- The frameworks give us the ‘plumbing’ to work with such code at large scale.



Example 1 – Map and Reduce without Spark

Notebook: `Lecture1_Example1_without_spark.ipynb`

- What about a much larger file (and more complex operations)?
 - Compute could be slow.
 - Disk IO could become bottleneck.
- Why Hadoop MapReduce?
 - Performance / High Scalability
- Why Apache Spark?
 - (Better) Performance / High Scalability & **Interactivity (e.g. notebooks, shell)**

What is Apache Spark?

<https://spark.apache.org/>

Apache Spark is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters.

“Unified engine for large-scale data analytics”



What is Apache Spark?

In essence, Apache Spark is a software framework for

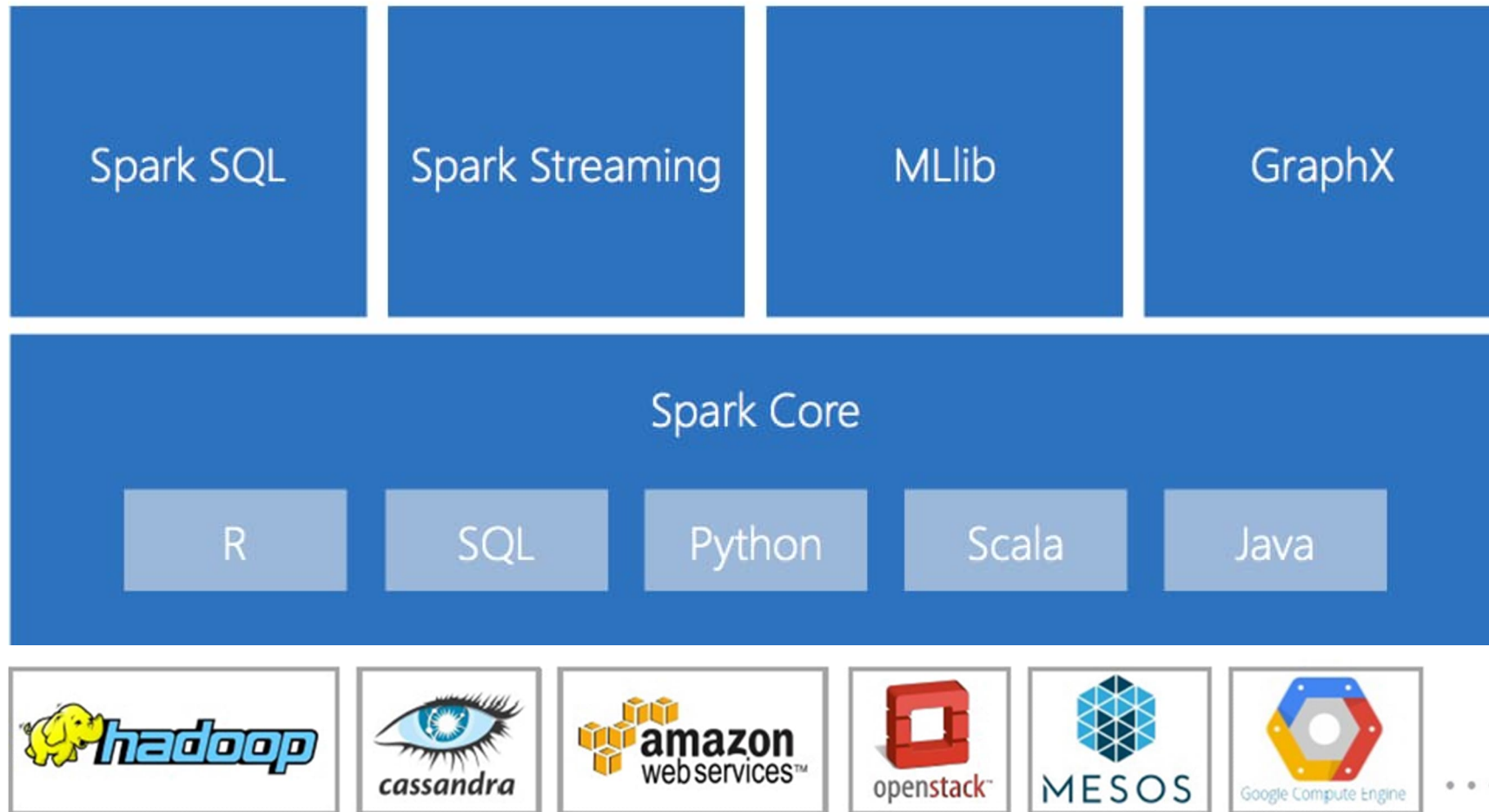
Distributed, Highly Scalable, High-Performance, Fault-Tolerant, Interactive

analysis of

Large Datasets using the **MapReduce** paradigm.

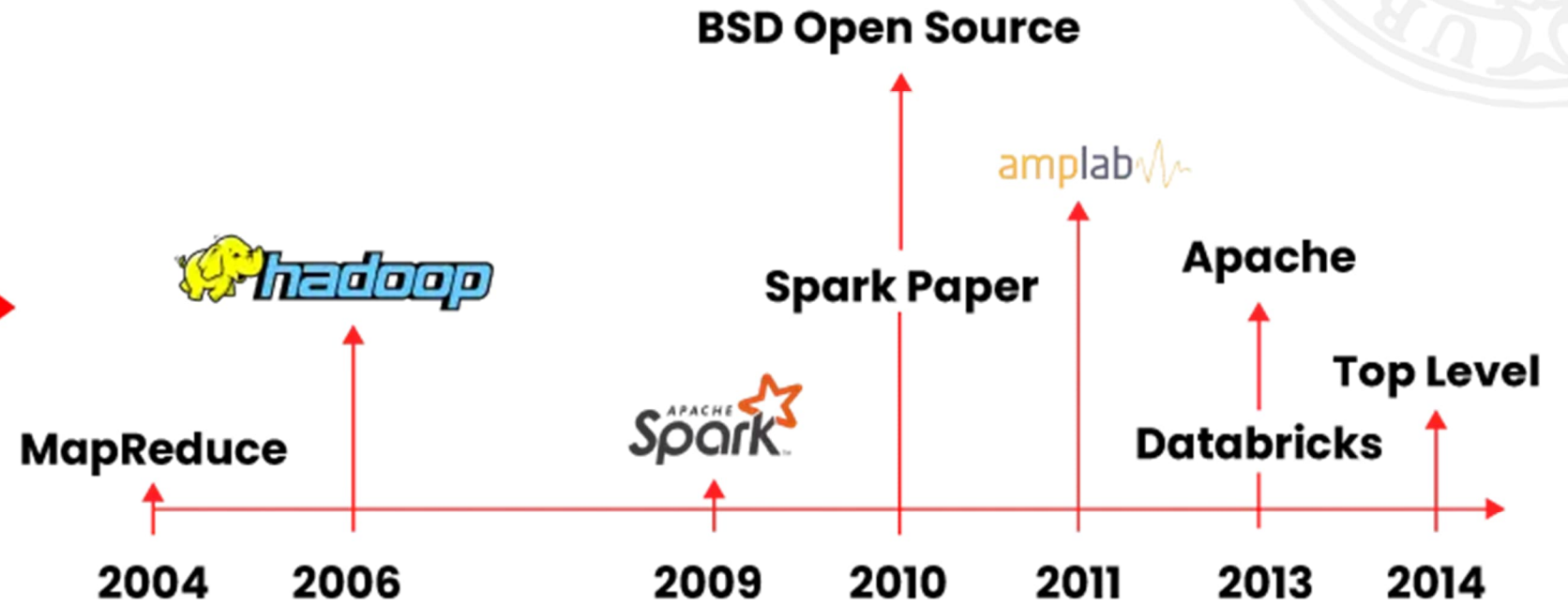


What is Apache Spark?



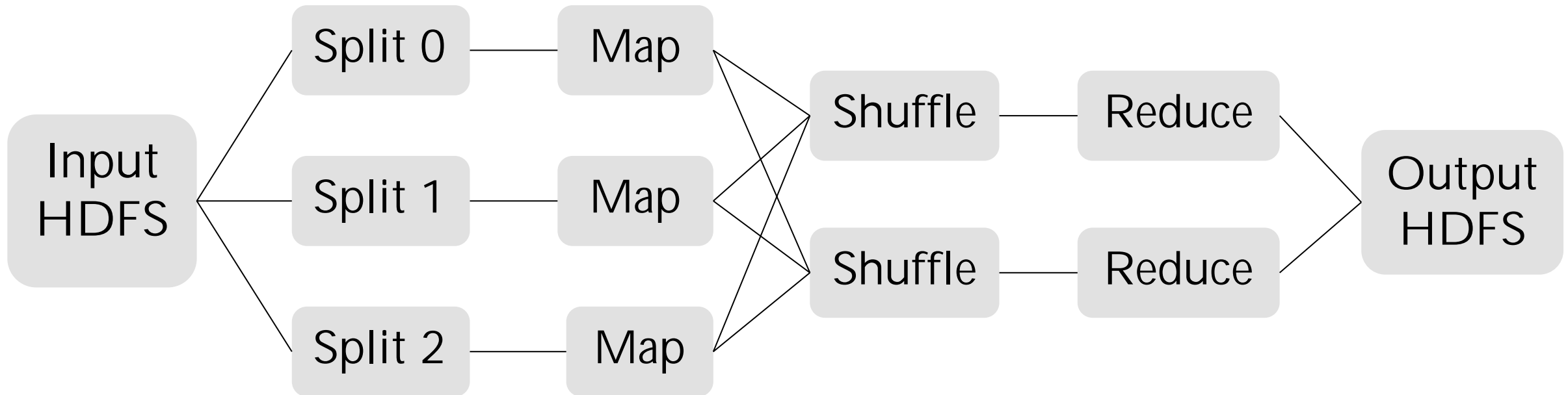
Apache Spark: A distributed computation framework

History of Spark



Why was Spark developed?

Hadoop MapReduce



Distributed computing processing framework for Big Data.

Why was Spark developed?

Only Map & Reduce, weak expressiveness.

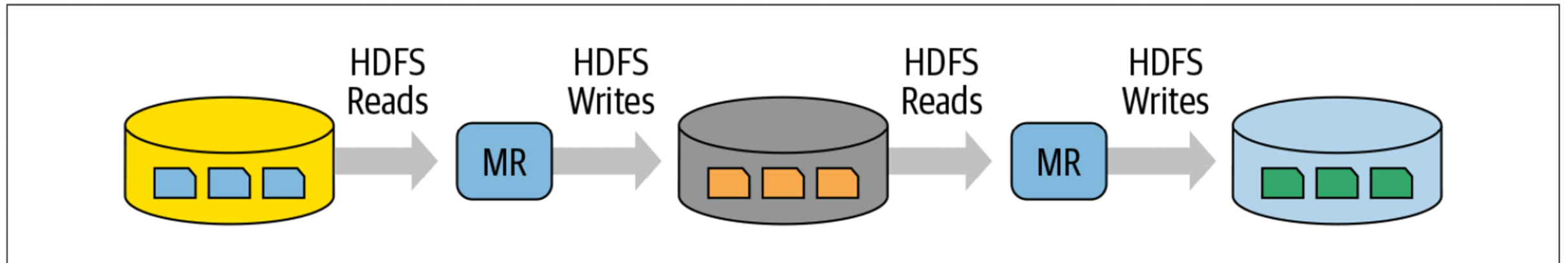
Input / Output: files on disk, jobs run sequentially. (**latency**)



Why was Spark developed?

Only Map & Reduce, weak expressiveness.

Input / Output: files on disk, jobs run sequentially. (**latency**)



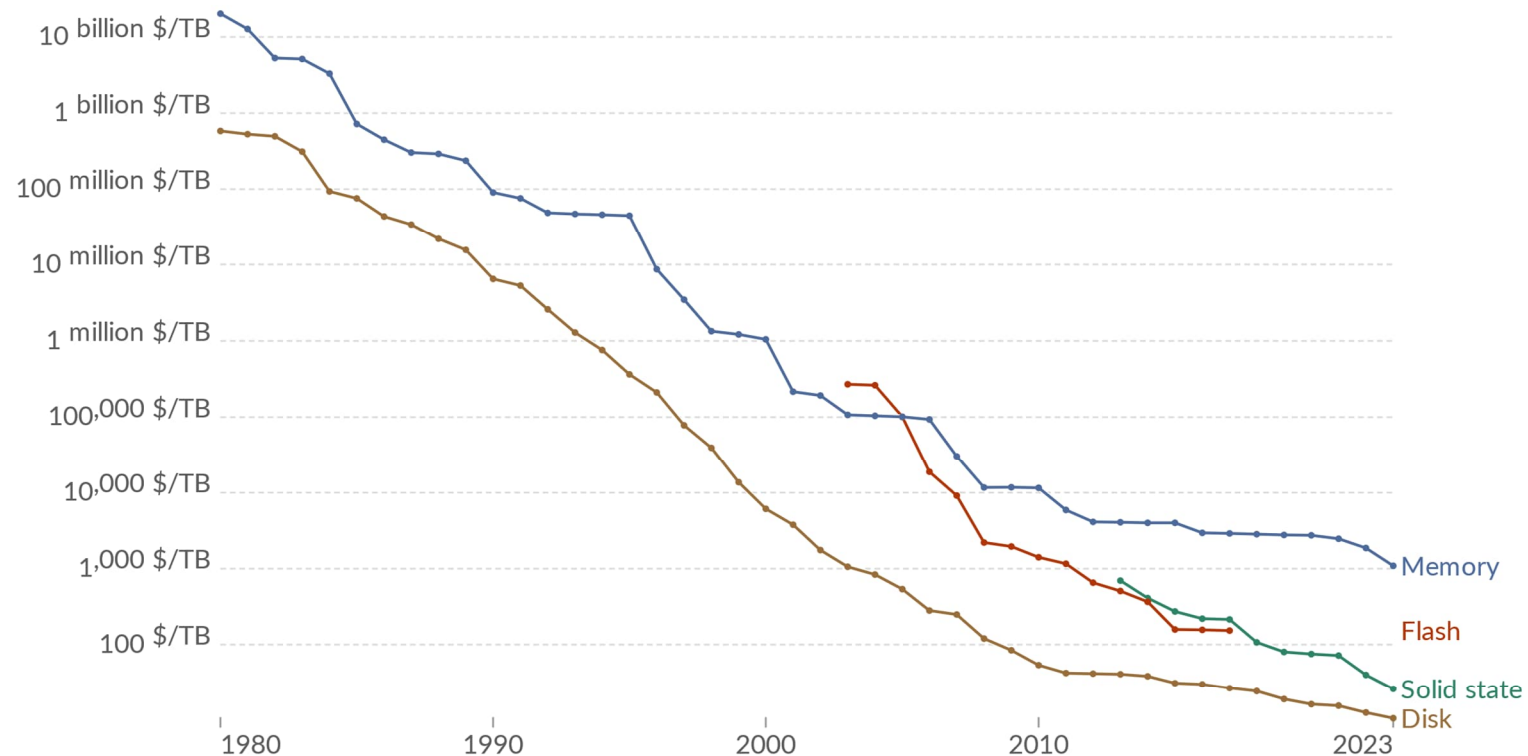
Intermittent iterations of reads and writes between map and reduce computations

Why was Spark developed?

Historical price of computer memory and storage

Our World
in Data

This data is expressed in US dollars per terabyte (TB), adjusted for inflation. "Memory" refers to random access memory (RAM), "disk" to magnetic storage, "flash" to special memory used for rapid data access and rewriting, and "solid state" to solid-state drives (SSDs).



Data source: John C. McCallum (2023); U.S. Bureau of Labor Statistics (2024)

OurWorldinData.org/technological-change | CC BY

Note: For each year, the time series shows the cheapest historical price recorded until that year. This data is expressed in constant 2020 US\$.

Why was Spark developed?

- Spark provide **memory**-based computing, higher iteration efficiency. (**RDD**)
- Distributed parallel computing with **DAG** (Directed Acyclic Graph), instead of shuffle.
- **Operations** include Map, Filter, FlatMap, Sample, GroupByKey, ReduceByKey, Union, Join, MapValues, Sort, PartionBy...
- General, uniform framework for different type of data.
- Interactivity: spark-shell, pyspark, sparkR, etc.

Why was Spark developed?

	Hadoop MR Record	Spark Record (2014)	Spark-based System 3x faster with 1/10 # of nodes
Data Size	102.5 TB	100 TB	
Elapsed Time	72 mins	23 mins	
# Nodes	2100	206	
# Cores	50400 physical	6592 virtualized	
Cluster disk throughput	3150 GB/s (est.)	618 GB/s	
Network	dedicated data center, 10Gbps	virtualized (EC2) 10Gbps network	
Sort rate	1.42 TB/min	4.27 TB/min	
Sort rate/node	0.67 GB/min	20.7 GB/min	

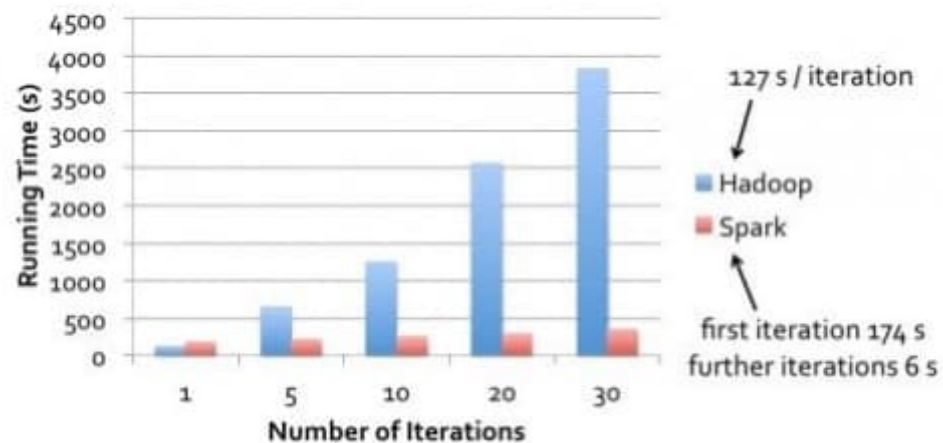
Sort benchmark, Daytona Gray: sort of 100 TB of data (1 trillion records)

<http://databricks.com/blog/2014/11/05/spark-officially-sets-a-new-record-in-large-scale-sorting.html>

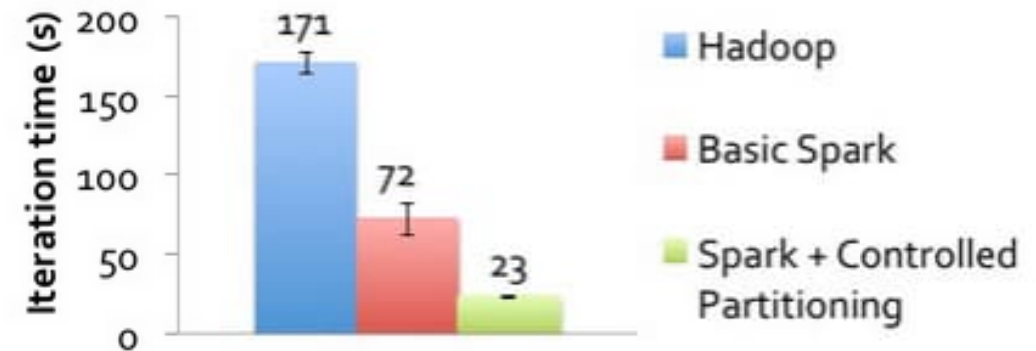


Why was Spark developed?

Logistic Regression Performance



PageRank Performance



Source: [Spark vs Hadoop: Which is the Best Big Data Framework?](#)

Example 2 – Map and Reduce with Spark

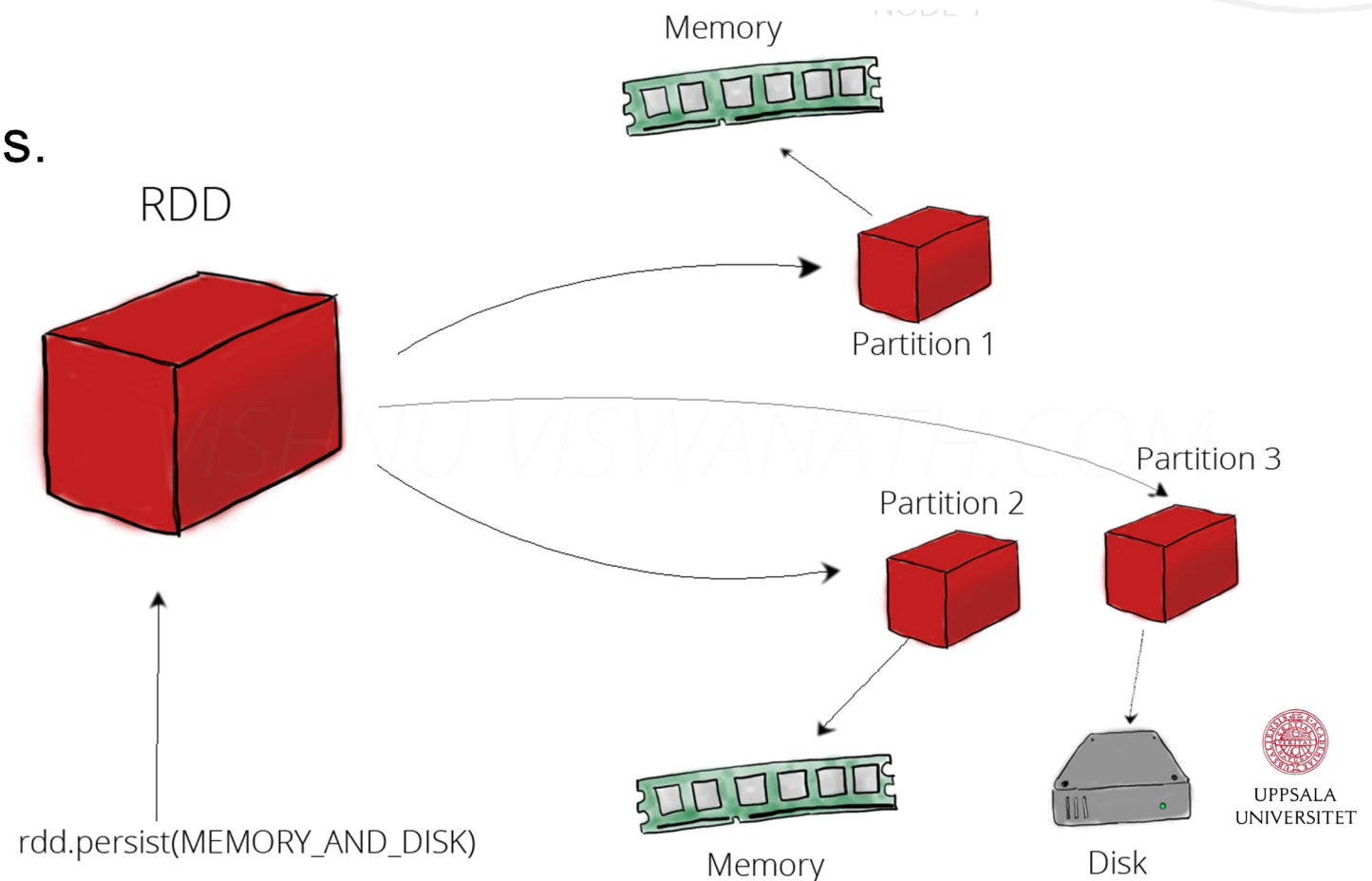
Notebook: Lecture1_Example2_with_spark.ipynb

- HDFS Web UI (file explorer): [Hadoop Web UI](#)
- Apache Spark Cluster Web UI: [Spark Web UI](#)
- Spark Application Link: [Spark Applications](#)

How does Spark work?

Resilient Distributed Dataset (RDD): fundamental data structure of spark.

- Read-only multiset of data items.
- Abstraction of dataset.
- Stored in memory / disk.
- Hash / ranged partitioned.




RDD Operations

- **Transformations** to build RDDs through deterministic operations on other RDDs
 - transformations include *map*, *filter*, *join*, ...
 - lazy operation
- **Actions** to return value or export data
 - actions include *count*, *collect*, *save*, ...
 - triggers execution



How does Spark work?

 = easy

 = medium

Essential Core & Intermediate Spark Operations

TRANSFORMATIONS

General

- map
- filter
- flatMap
- mapPartitions
- mapPartitionsWithIndex
- groupBy
- sortBy

Math / Statistical

- sample
- randomSplit

Set Theory / Relational

- union
- intersection
- subtract
- distinct
- cartesian
- zip

Data Structure / I/O

- keyBy
- zipWithIndex
- zipWithUniqueId
- zipPartitions
- coalesce
- repartition
- repartitionAndSortWithinPartitions
- pipe

ACTIONS

- reduce
- collect
- aggregate
- fold
- first
- take
- foreach
- top
- treeAggregate
- treeReduce
- foreachPartition
- collectAsMap


- count
- takeSample
- max
- min
- sum
- histogram
- mean
- variance
- stdev
- sampleVariance
- countApprox
- countApproxDistinct

- takeOrdered

- saveAsTextFile
- saveAsSequenceFile
- saveAsObjectFile
- saveAsHadoopDataset
- saveAsHadoopFile
- saveAsNewAPIHadoopDataset
- saveAsNewAPIHadoopFile

How does Spark work?

 = easy

 = medium

Essential Core & Intermediate PairRDD Operations


TRANSFORMATIONS

General

- flatMapValues
- groupByKey
- reduceByKey
- reduceByKeyLocally
- foldByKey
- aggregateByKey
- sortByKey
- combineByKey

Math / Statistical

- sampleByKey

Set Theory / Relational

- cogroup (=groupWith)
- join
- subtractByKey
- fullOuterJoin
- leftOuterJoin
- rightOuterJoin

Data Structure

- partitionBy

ACTIONS



- keys
- values

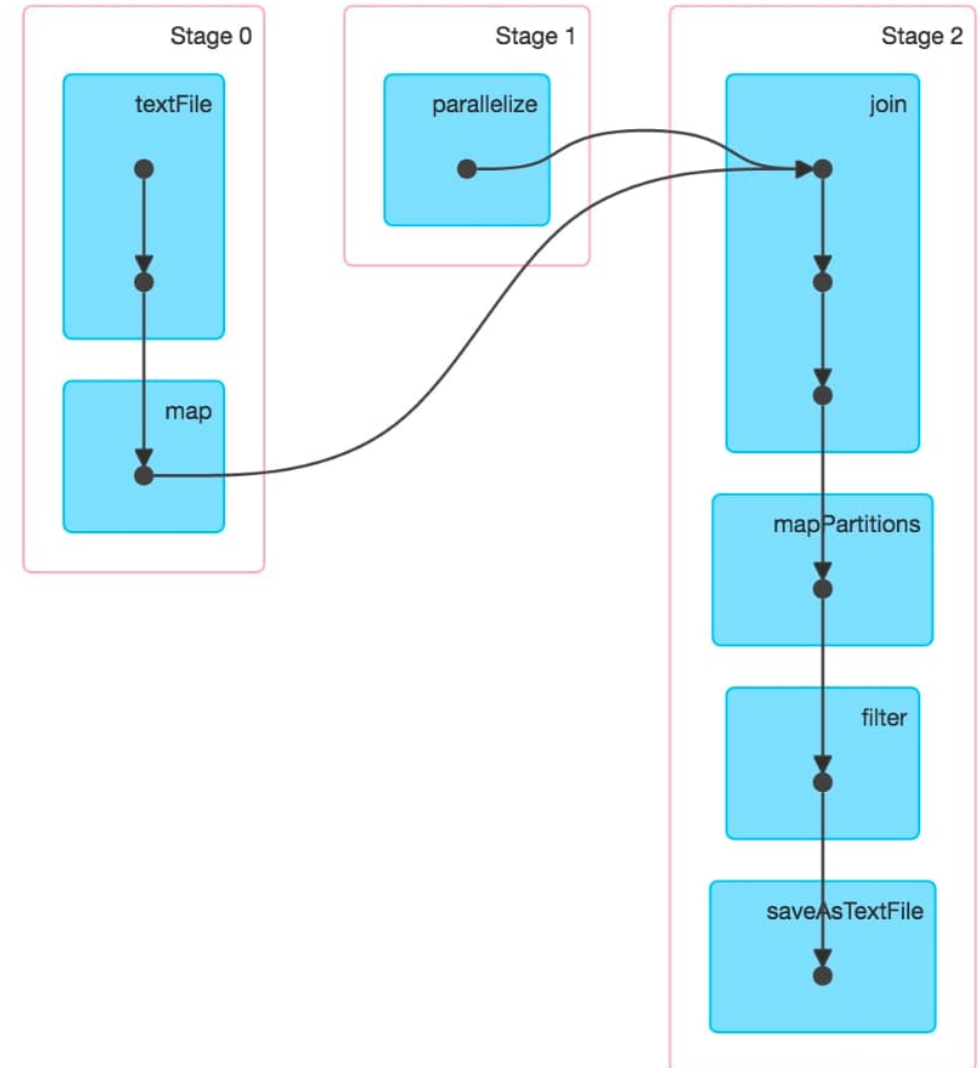
- countByKey
- countByValue
- countByValueApprox
- countApproxDistinctByKey
- countApproxDistinctByKey
- countByKeyApprox
- sampleByKeyExact



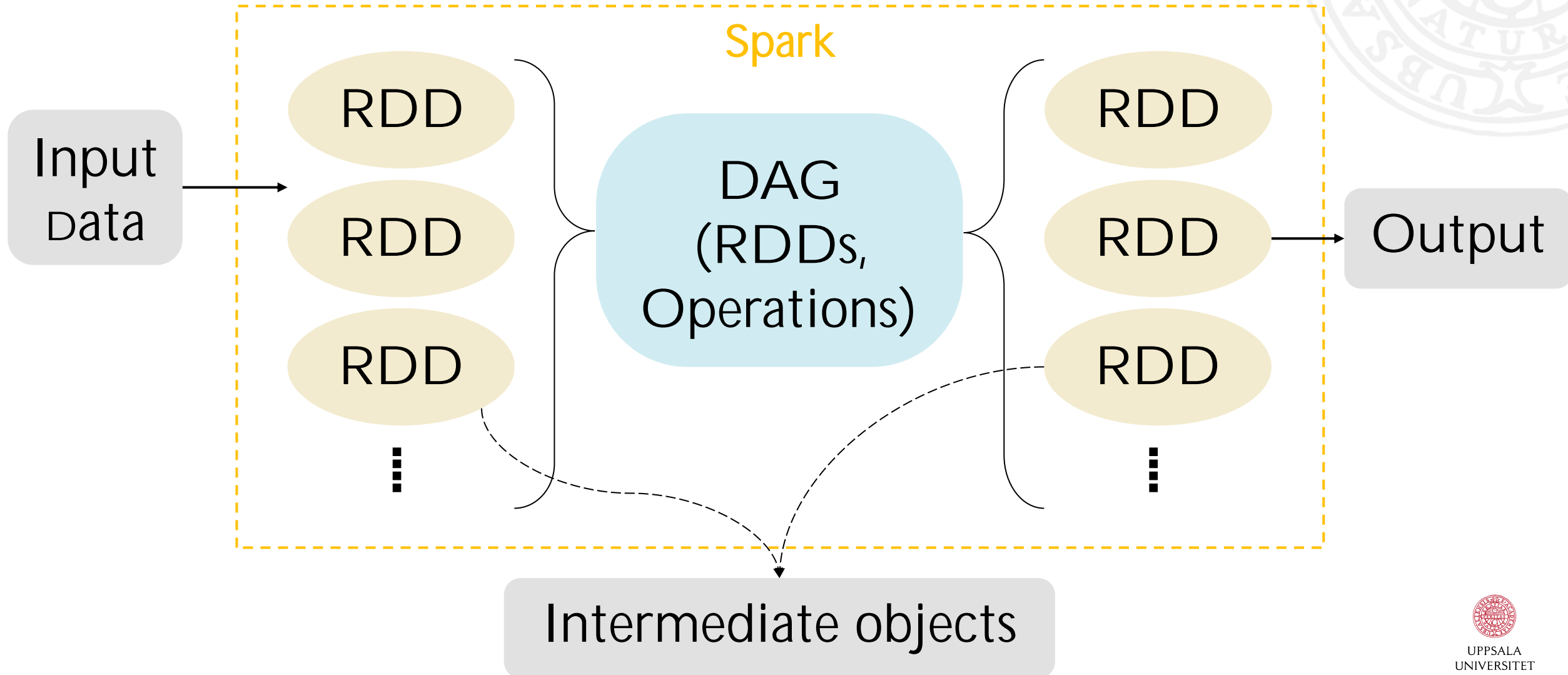
How does Spark work?

Instead of shuffle in MapReduce,
Spark use **Directed Acyclic Graph**
as the jobs scheduler and manager.

A DAG = (RDDs, operations).



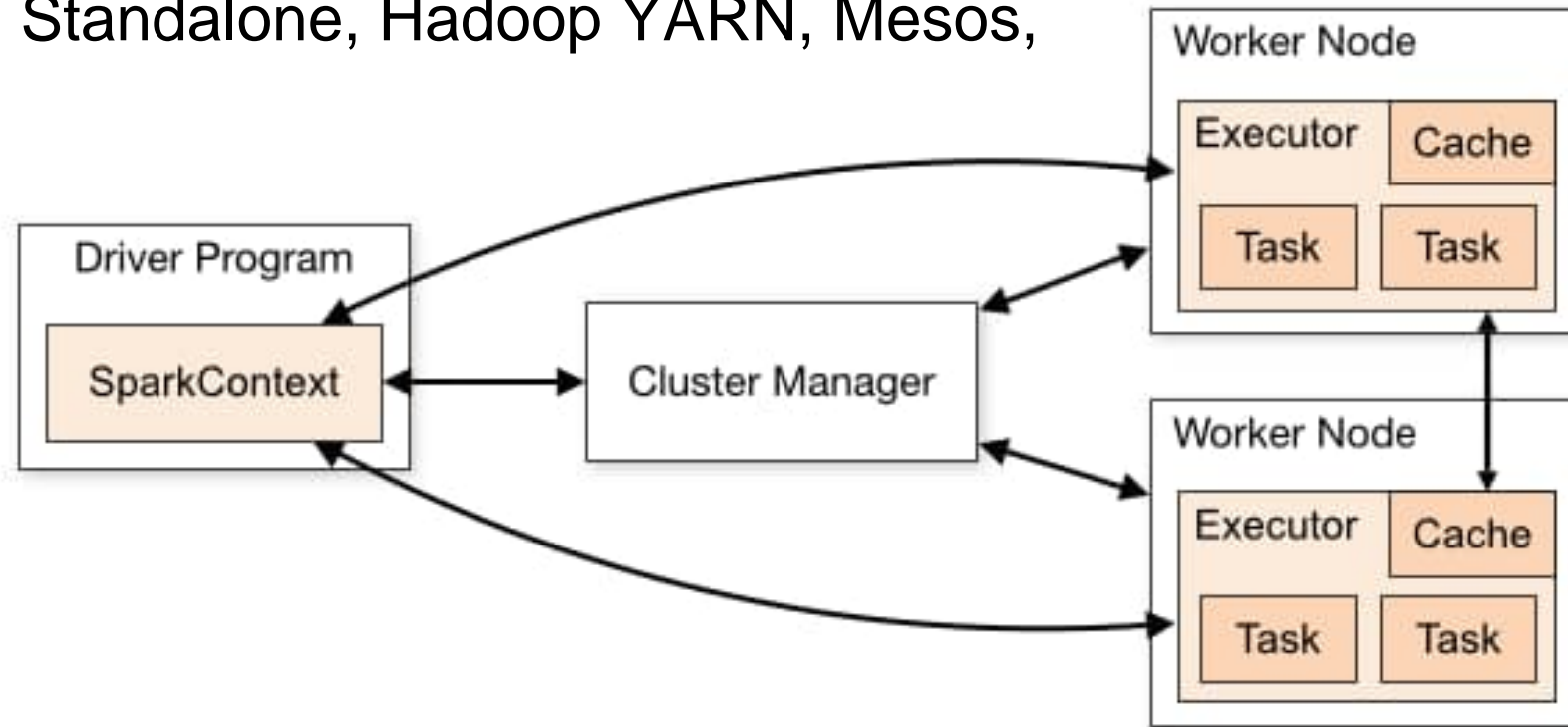
How does Spark work?



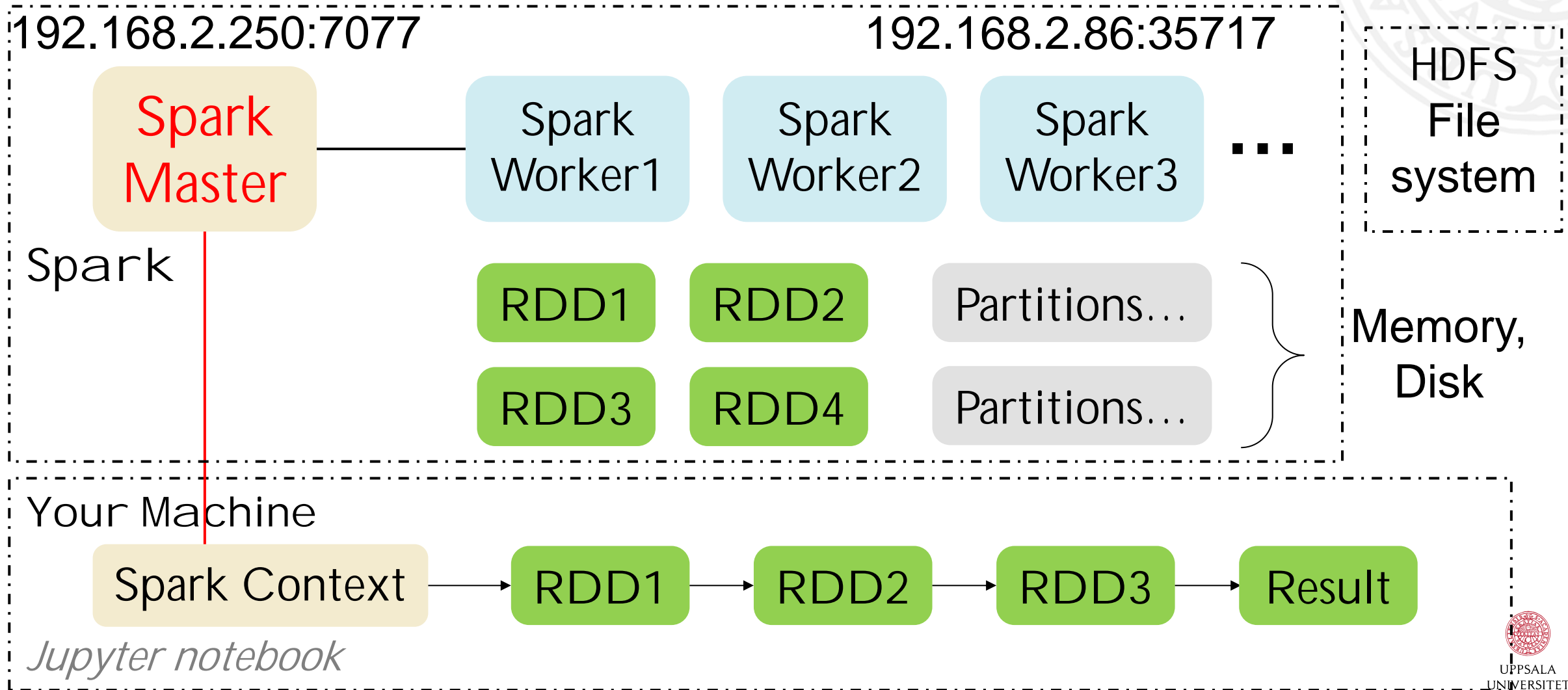
How does Spark work?

Spark requires a **cluster manager** and a **distributed storage system**.

Cluster manager: Standalone, Hadoop YARN, Mesos, Kubernetes, etc.



How does Spark work?



Examples 3–5: Methods on RDD



Notebooks: [Examples](#)

- Lecture1_Example3_RDD_methods.ipynb
- Lecture1_Example4_RDD_Gutenberg.ipynb
- Lecture1_Example5_RDD_CommonCrawl_Demo.ipynb



Summary: RDD Operations

Action	Transformation
Can have side effects (like saving to disk)	No side effects
Imperative Evaluation (triggers jobs)	Lazily Evaluated. Can be re-evaluated.
Invocation to request output, or to 'do' something	Describes how to transform from one step to the next.
Returns or outputs data	Returns another RDD
reduce(..), save(...), collect(..), count(..),	map(...), sort(..), filter(..), reduceByKey(..)
https://spark.apache.org/docs/latest/rdd-programming-guide.html#actions	https://spark.apache.org/docs/latest/rdd-programming-guide.html#transformations



Key Links

Spark WebUI:

Spark Cluster Master: <http://localhost:8080/>

Spark Cluster Worker: <http://localhost:8081/>

Spark Application: <http://localhost:4040/> or :4041, :4042, :4043, ...

Additional Resources

Check the GitHub repository and understand each demo.

<https://github.com/usamazf/DE1-Spark>

Spark official examples:

<https://github.com/jkthompson/pyspark-pictures>

RDD Operations with Visualization:

<https://github.com/jkthompson/pyspark-pictures>

PySpark cheat sheet: *See in Studium*

Additional Resources

Spark API:

- [Spark Scala API \(Scaladoc\)](#)
- [Spark Java API \(Javadoc\)](#)
- [Spark Python API \(Sphinx\)](#)
- [Spark R API \(Roxygen2\)](#)
- [Spark SQL, Built-in Functions \(MkDocs\)](#)





End of first lecture.

