IoT·인공지능·빅데이터 개론 및 실습

빅데이터 배치, 대화형 질의, 스트림 분석

서울대학교 컴퓨터공학부
전병곤

Contents

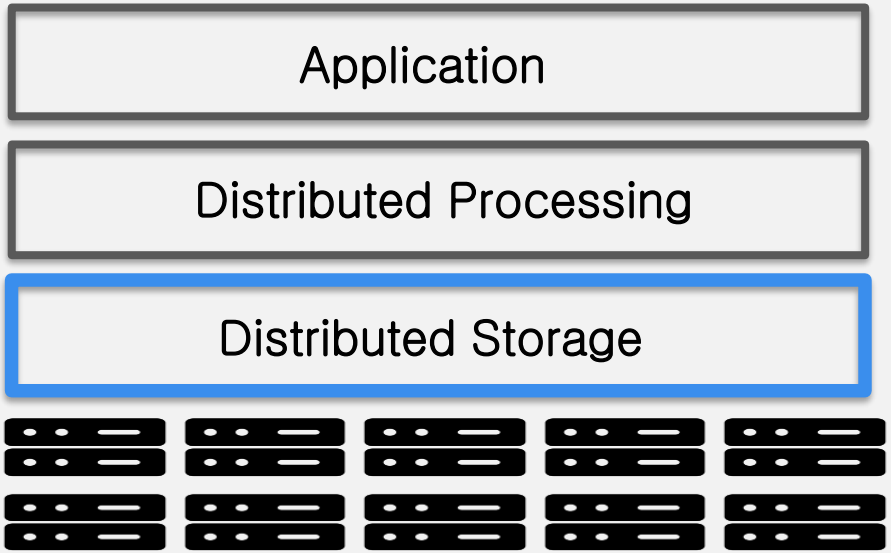
1 빅데이터 스토리지

2 데이터 처리 일반

3 대화형 질의

4 스트림 처리

10,000 Feet View of Big Data Systems



(1) Data Storage

- Write and read data in a distributed storage

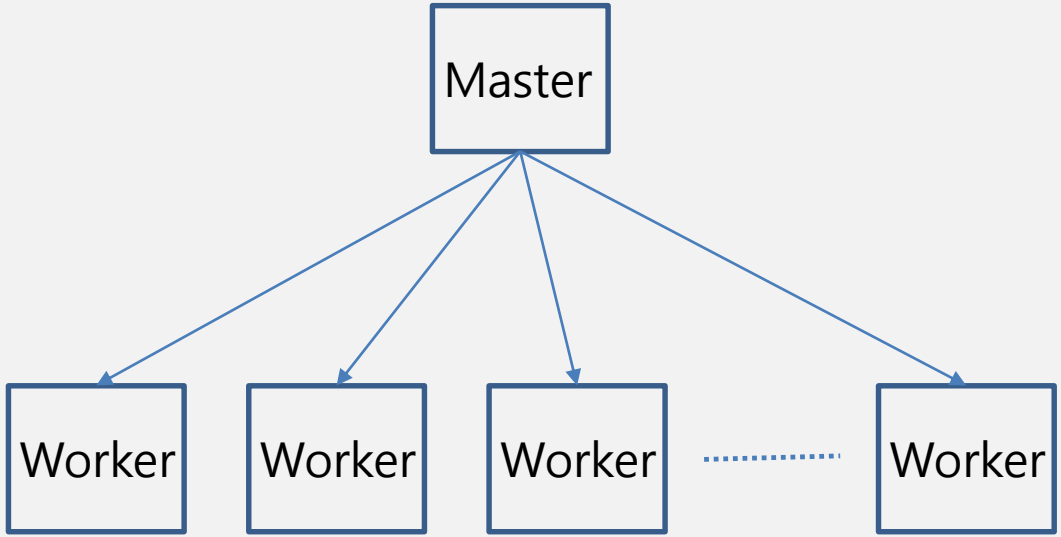
- Distributed file system
  - Google file system (GFS)
  - Hadoop distributed file system (HDFS): an open source implementation of GFS

(2) Hadoop Distributed File System (HDFS)

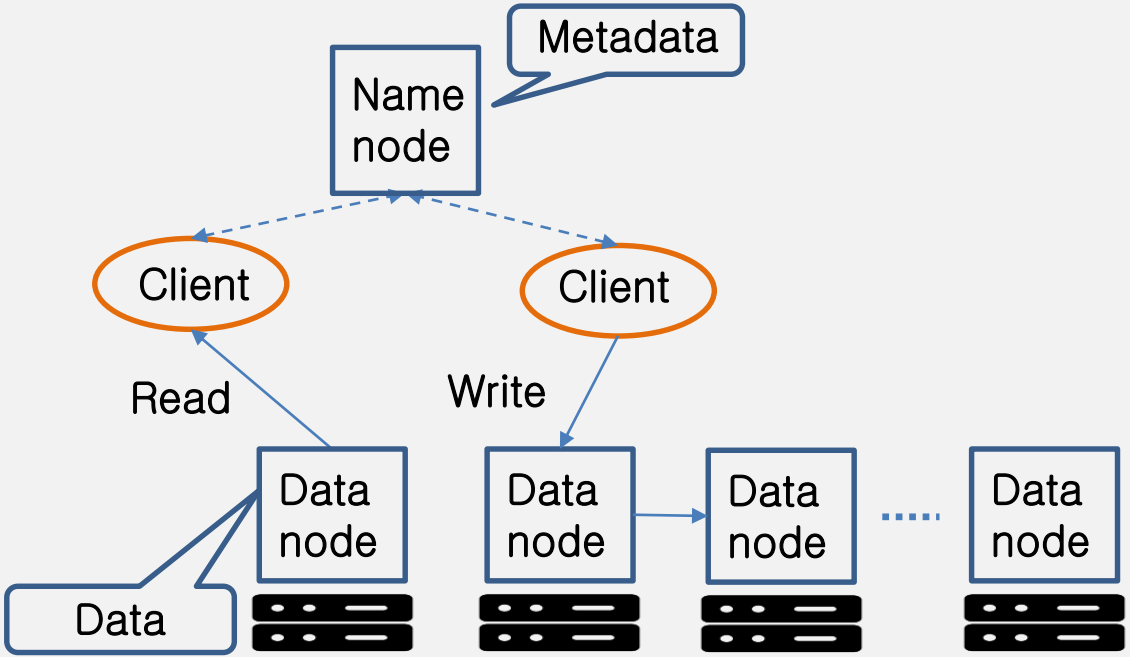
- File system interface: file, directory operations
- Optimized for big data processing
  - Streaming read of large data
  - Large, sequential writes that append data to files
  - Large blocks (e.g., 128MB per block)
  - Fault tolerance
- Master-worker architecture

(2) Hadoop Distributed File System (HDFS)

- Master-worker architecture



(2) Hadoop Distributed File System (HDFS)



(2) Hadoop Distributed File System (HDFS)

HDFS architecture

- Data flow is decoupled from control flow
- Clients interact with the namenode for metadata operations
- Clients interact directly with datanodes for all files operations
- Performance can be improved by scheduling expensive data flow based on the network topology

Contents

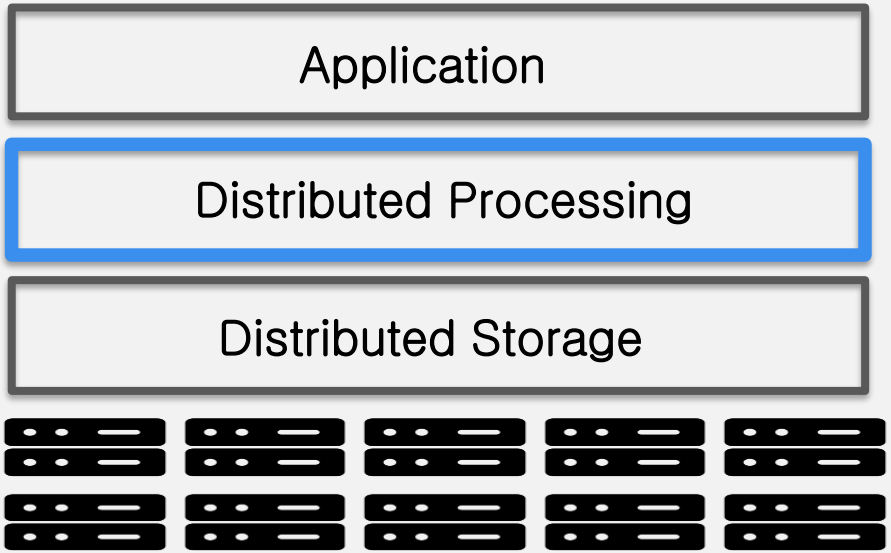
1 빅데이터 스토리지

2 데이터 처리 일반

3 대화형 질의

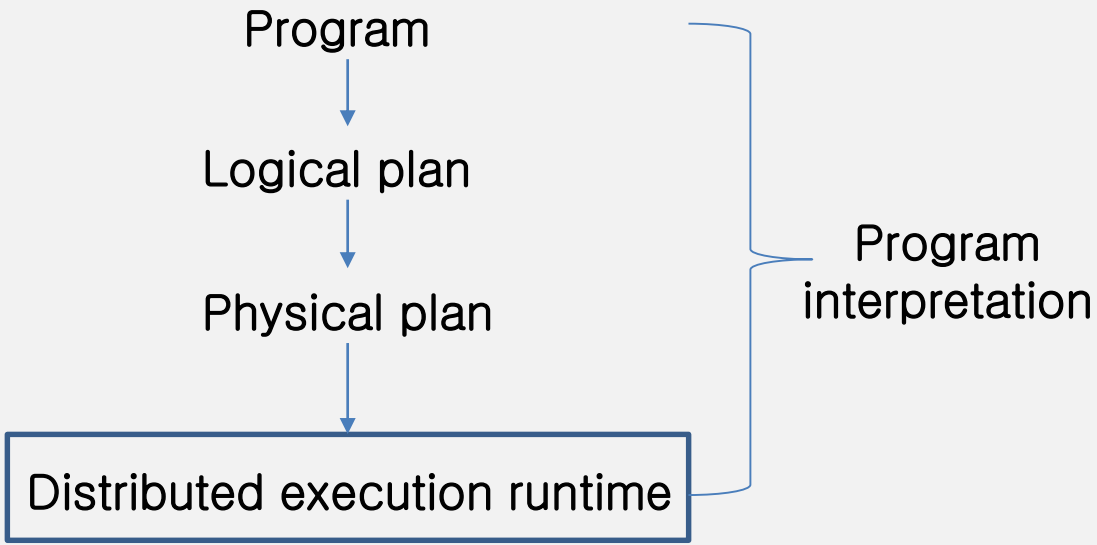
4 스트림 처리

10,000 Feet View of Big Data Systems



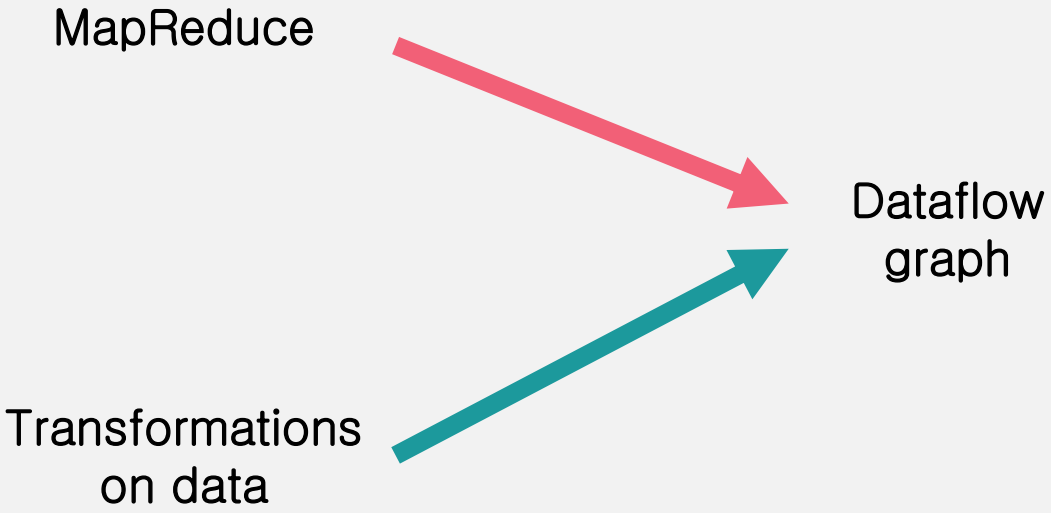
(1) 데이터플로우 모델

› Data processing



(1) 데이터플로우 모델

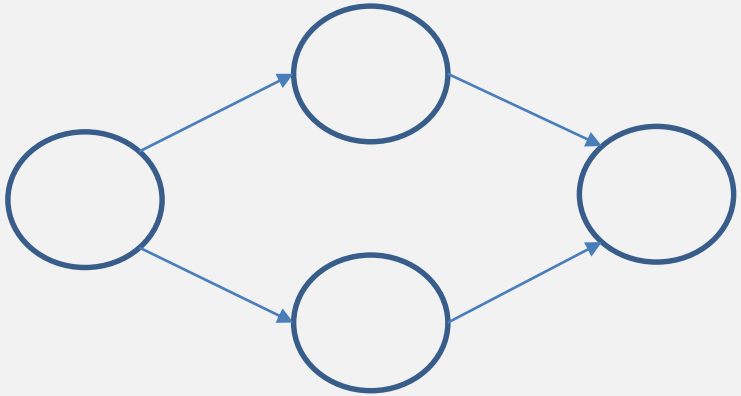
› Data processing programming model



(1) 데이터플로우 모델

Dataflow graph

- A Directed Acyclic Graph of operators
  - Vertex: operator
  - Edge: data dependency



(2) 맵리듀스

MapReduce programming model

- Map function: process a key/value pair to generate a set of intermediate key/value pairs

- Reduce function: merges all intermediate values associated with the same intermediate key

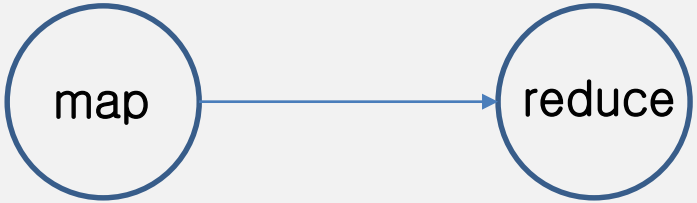
(2) 맵리듀스

MapReduce WordCount

- Wordcount problem: compute how many times each word appears in a collection of documents
- Map: lambda word: (word, 1)
- Reduce: lambda a, b: a + b

(2) 맵리듀스

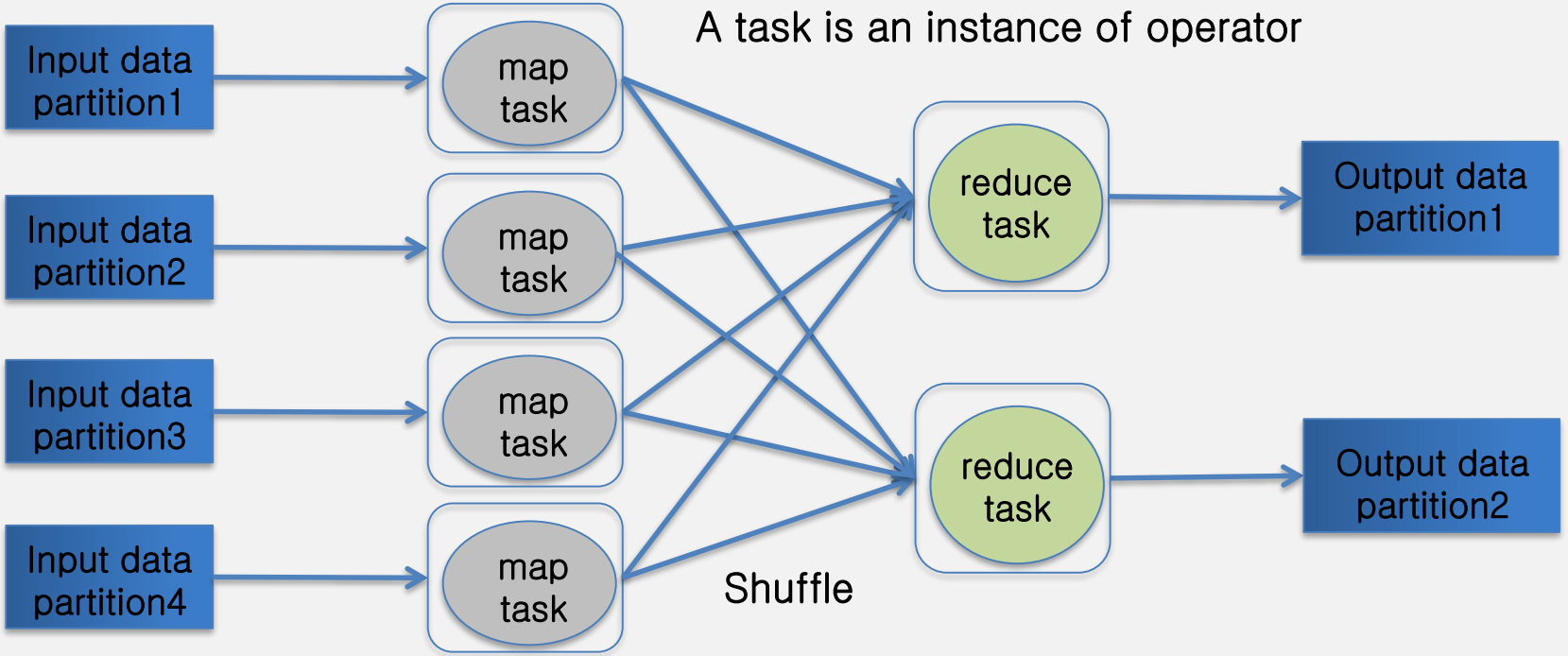
- MapReduce : logical plan



lambda word: (word, 1) lambda a, b: a + b

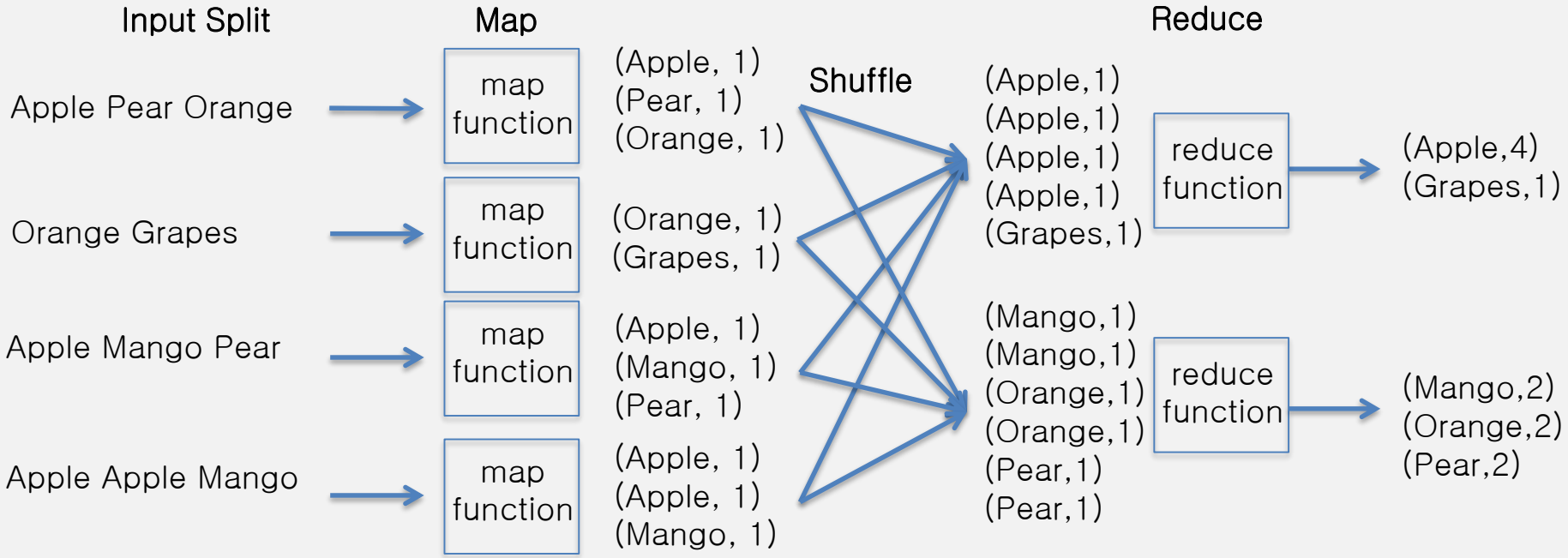
(2) 맵리듀스

- MapReduce : physical plan



(2) 맵리듀스

MapReduce WordCount execution example



(2) 맵리듀스

- MapReduce runtime handles
  - Scheduling
  - Fault Tolerance
  - Scalability
  - Elasticity

(2) 맵리듀스

Limitations of MapReduce

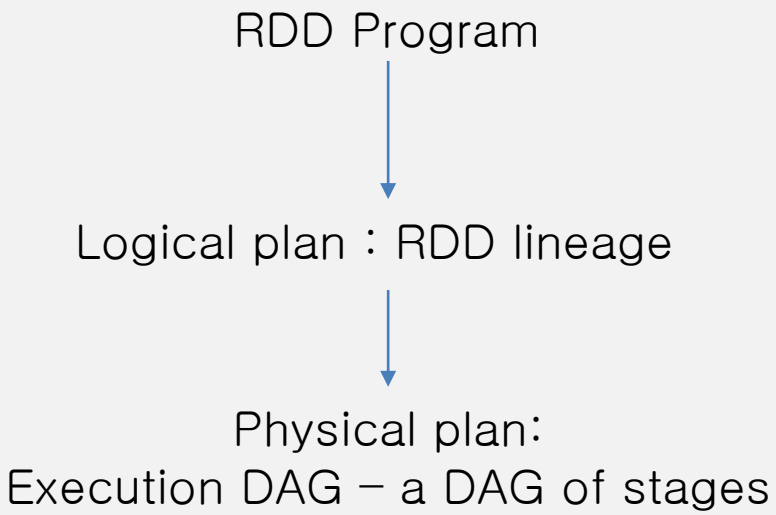
- Limited programming flexibility
  - In particular, inefficient for multi-pass algorithms
- No efficient primitives for data sharing
  - State between MapReduce passes goes through distributed file systems

(3) 스파크

- Generic dataflow graph: a DAG of operators
- In-memory computing with Resilient Distributed Datasets (RDDs)
- Easy-to-use programming interface: transformation on RDDs

(3) 스파크

- Spark logical plan, physical plan



(3) 스파크

RDDs

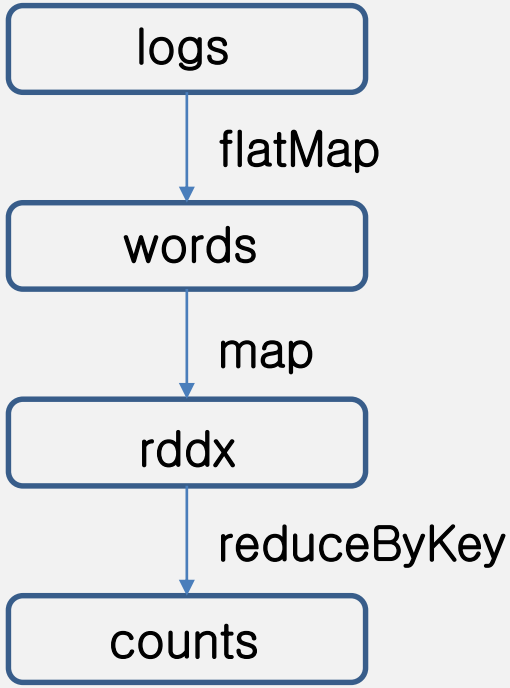
- Collections of “immutable” objects spread across a cluster
- Statically typed: RDD[T] has objects of type T
- Built through parallel transformations
- All transformations are lazy
- Automatically rebuilt on a failure through lineage
- Two types of operations
  - Transformations (e.g., map, filter, groupBy)
  - Actions (e.g., count, collect)
- RDD can be persisted into storage in memory or disk

(3) 스파크

- RDD lineage example

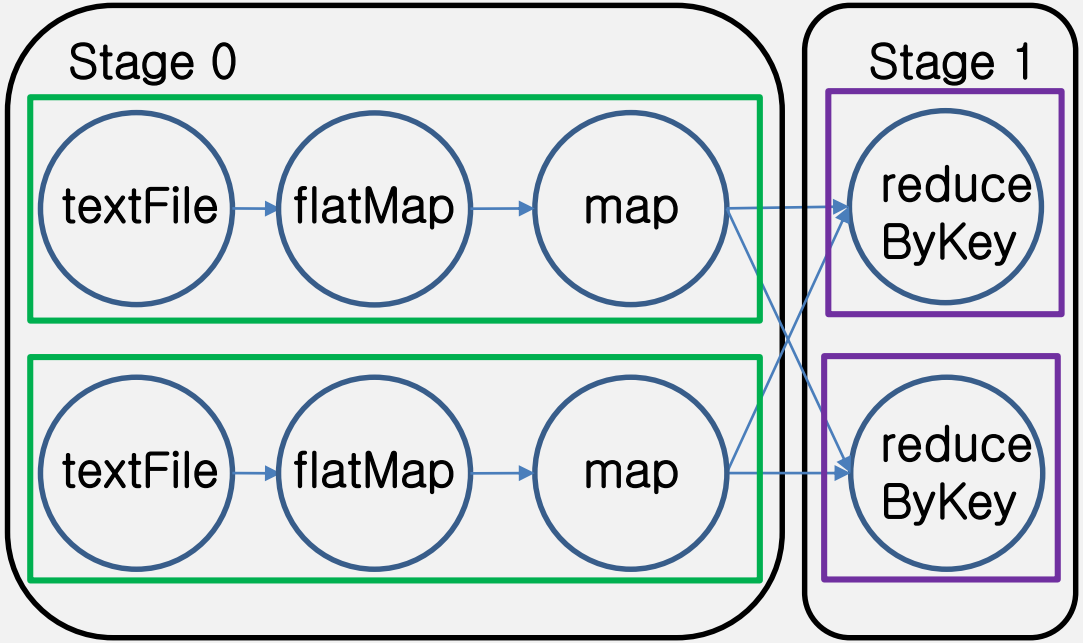
```
logs = sc.textFile(“hdfs://data/logs”)

words = logs.flatMap(lambda line: line.split())
counts = words.map(lambda word: (word, 1))
                .reduceByKey(lambda c1, c2: c1 +
c2)
counts.saveAsTextFile(“hdfs://result/counts”)
```



(3) 스파크

› Execution DAG



(3) 스파크

Batch processing

- Data preprocessing
- Extract-Transform-Load (ETL)
- Data processing to do in bulk
  (not latency-critical)

Contents

1 빅데이터 스토리지

2 데이터 처리 일반

3 대화형 질의

4 스트림 처리

3 대화형 질의

(1) Interactive Query

- Structured data
- Use SQL to express queries on structured data
- SQL: a domain-specific language used for managing data in relational models

3 대화형 질의

(2) Structured Data

- Data managed in a table
- A table consists of rows and columns

Table name: customer

id	name	age	Place
1	Alex Kim	32	Seoul
2	Jane Lee	25	Seoul
3	Matt Park	23	Incheon

(2) Structured Data

Operations on structured data

- Projection: select specified columns from a table
  - E.g., $\Pi_{\{name,age\}}$ (customer)

name	age
Alex Kim	32
Jane Lee	25
Matt Park	23

(2) Structured Data

Operations on structured data

- Selection: select a subset of rows from a table that meets a predicate
  - E.g., $\sigma_{age > 30}(customer)$

id	name	age	place
1	Alex Kim	32	Seoul

(2) Structured Data

Operations on structured data

- Aggregation: aggregate on a column
  - E.g., $\mathbf{G}_{\mathbf{sum(place)}}$ (customer)

place	count
Seoul	2
Incheon	1

(2) Structured Data

Operations on structured data

- Join: compute rows in two tables that are equal on their common column names
  - Natural join
  - Left outer join
  - Right outer join

(2) Structured Data

Operations on structured data

- Natural join
  - E.g., customer ⋈ department

Table name: customer

id	name	age	place
1	Alex Kim	32	Seoul
2	Jane Lee	25	Seoul
3	Matt Park	23	Incheon

Table name: department

place	lead
Seoul	Bob
Incheon	Alice

id	name	age	place	lead
1	Alex Kim	32	Seoul	Bob
2	Jane Lee	25	Seoul	Bob
3	Matt Park	23	Incheon	Alice

customer ⋈ department

(3) SparkSQL: Interactive Query in Spark

- DataFrame: a Dataset organized into named columns
- Conceptually equivalent to a table in a relational database or a data frame in R/Python but with richer optimizations under the hood
- Can be constructed from various sources such as structured data files, tables in Hive, external databases, or existing RDDs
- Support relational operations
- An SQL query is translated into RDD transformations to be executed

Contents

1 빅데이터 스토리지

2 데이터 처리 일반

3 대화형 질의

4 스트림 처리

4 스트림 처리

(1) Stream Processing

Stream data

- Unbounded data
- Event time != data processing time

Real-time processing

- Window operation

(1) Stream Processing

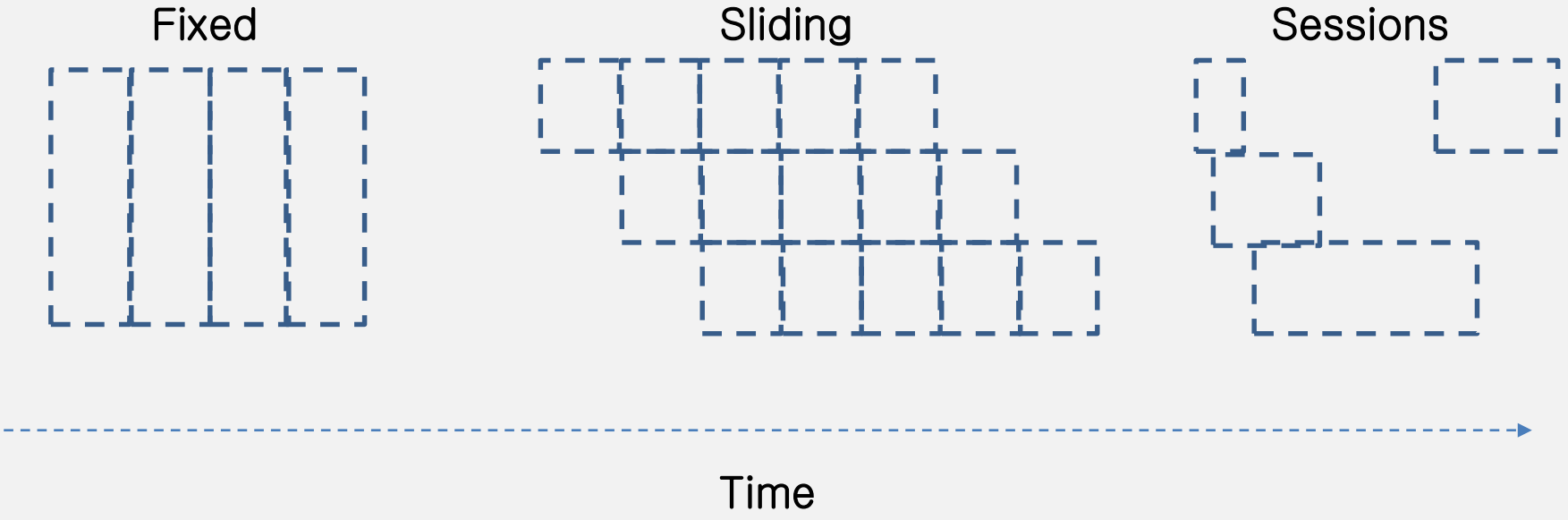
- What: stream processing model

- How: stream processing execution engine
  - Continuous query: e.g., Storm, Heron, Flink
  - Micro-batch: e.g., SparkStreaming

(2) Streaming Patterns

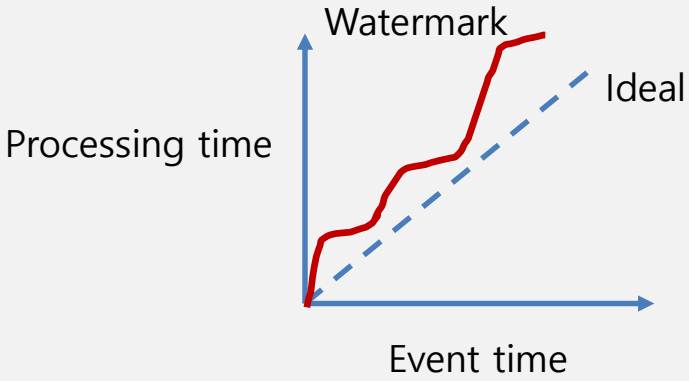
- Element-wise transformations
- Processing-time based windows
- Event-time based windows
- Session windows

(3) Windowing



(4) Watermarks

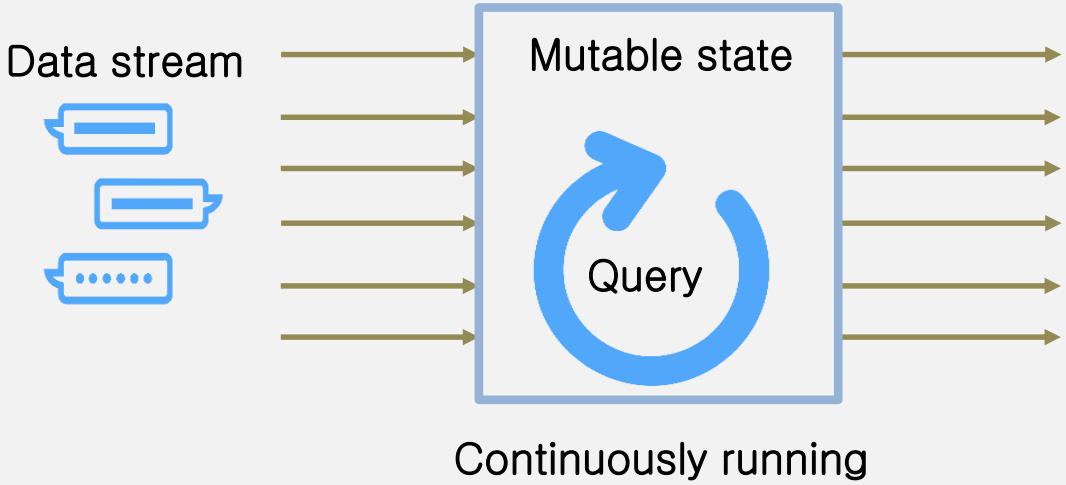
- Watermark: no timestamp earlier than the watermark will be seen



- Watermark too slow: results delayed
- Watermark too fast: some data late

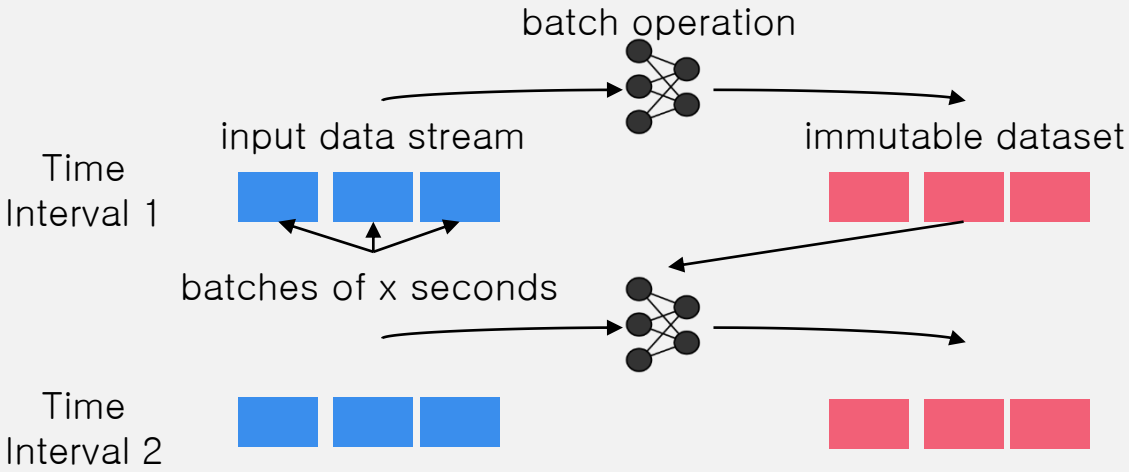
(5) Stream Query Execution Model

- Continuous Query: e.g,. Storm, Heron, Flink



(5) Stream Query Execution Model

- Micro-batch: e.g., SparkStreaming
- Dissociate computation from state: make state immutable and break computation into small, deterministic, stateless tasks



요약

- 빅데이터 스토리지
- 데이터 처리 일반
- 대화형 질의
- 스트림 처리