CLOUD COMPUTING APPLICATIONS

Apache Spark

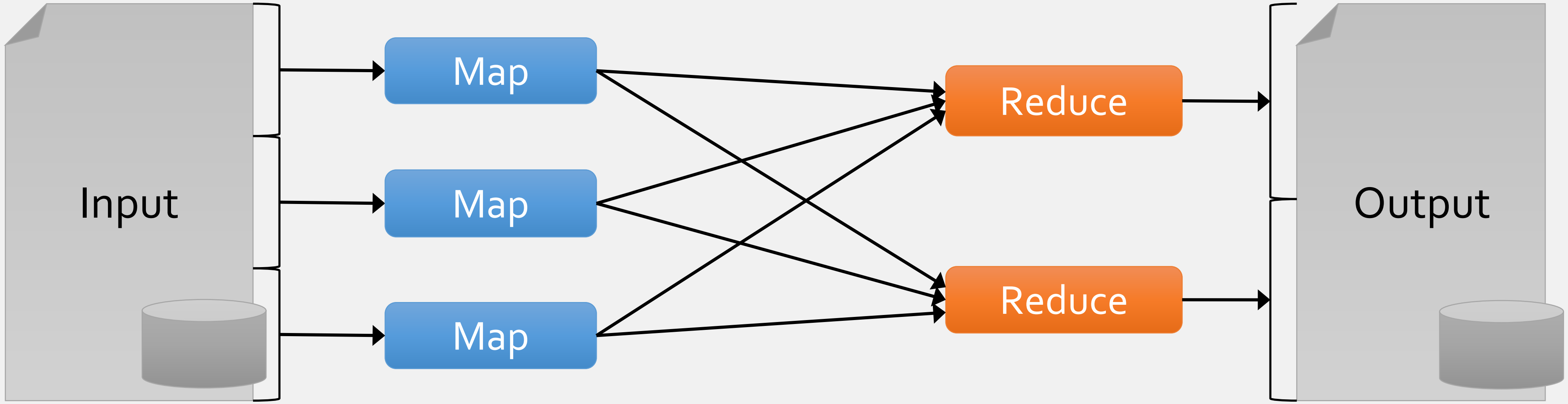
Roy Campbell & Reza Farivar

Apache Spark

- Extend the MapReduce model to better support two common classes of analytics apps:
  - **Iterative** algorithms (machine learning, graphs)
  - **Interactive** data mining
- Enhance programmability:
  - Integrate into Scala programming language
  - Allow interactive use from Scala interpreter

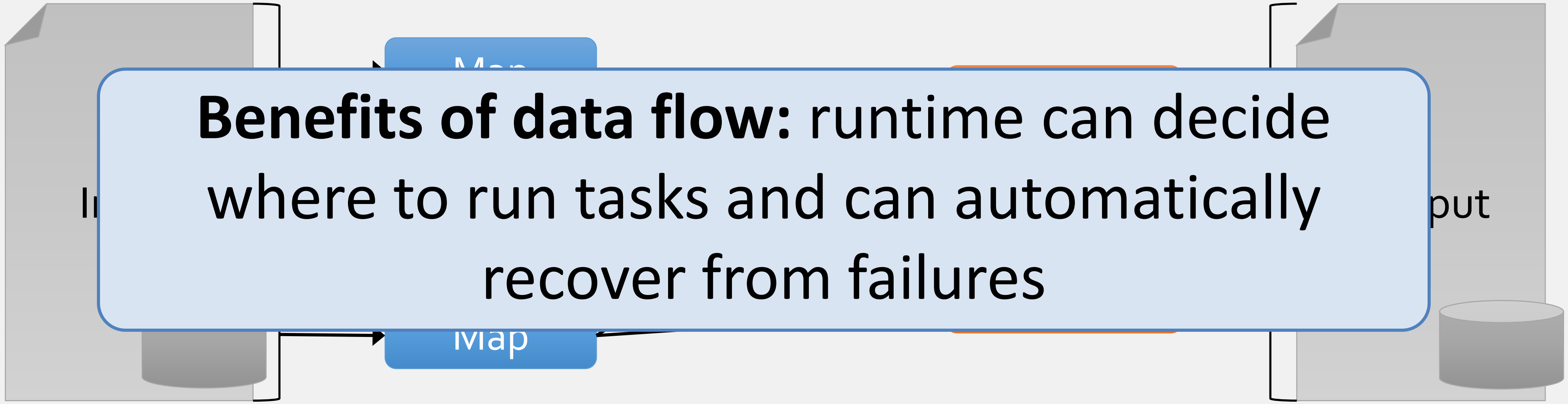
Motivation

Most current cluster programming models are based on *acyclic data flow* from stable storage to stable storage



Motivation

Most current cluster programming models are based on *acyclic data flow* from stable storage to stable storage



Benefits of data flow: runtime can decide where to run tasks and can automatically recover from failures

Motivation

- Acyclic data flow is inefficient for applications that repeatedly reuse a *working set* of data:
  - **Iterative** algorithms (machine learning, graphs)
  - **Interactive** data mining tools (R, Excel, Python)
- With current frameworks, apps reload data from stable storage on each query

Solution: Resilient Distributed Datasets (RDDs)

- Allow apps to keep working sets in memory for efficient reuse
- Retain the attractive properties of MapReduce
  - Fault tolerance, data locality, scalability
- Support a wide range of applications

Programming Model

- Resilient distributed datasets (RDDs)
  - Immutable, partitioned collections of objects
  - Created through parallel *transformations* (map, filter, groupBy, join, …) on data in stable storage
  - Can be *cached* for efficient reuse
- Actions on RDDs
  - Count, reduce, collect, save, …