UNIVERSITÀ DEGLI STUDI FIRENZE

DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

Hadoop

Gianni Pantaleo, Imad Zaza

Distributed [Systems and internet | Data Intelligence and] Technologies Lab
Dipartimento di Ingegneria dell'Informazione (DINFO),
Universita' degli Studi di Firenze
Fax: 0039-055-2758570, tel: 0039-3355668674
http://www.disit.dinfo.unifi.it , http://www.km4city.org

Summary

- Problem Scope
- Hadoop
- Hadoop Architecture
  - HDFS
  - MapReduce
- How Hadoop Works

Problem scope

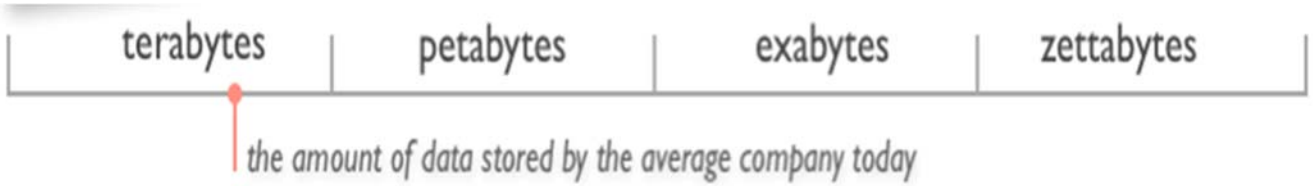
Big Data: Definizione

Con "***Big Data***" si intende un set di dati la cui dimensione, diversità e complessità richiede nuove soluzioni, tecniche, algoritmi e architetture di elaborazione specifiche per poter estrarre informazione e conoscenza

Volume

- **Data Volume**
  - 44x increase from 2009 to 2020
  - From 0.8 zettabytes to 35zb

Il volume dei dati generati cresce esponenzialmente



Eterogeneità di Dati e Formati

- Numerosi formati, tipi e strutture di dati differenti
- Formati testuali, numerici, immagini, file multimediali (audio, video), serie temporali di dati, metadata da user profiles, dati da social media, array multidimensoniali, ecc…
- Dati Statici vs Dati Dinamici *(streaming, real time…)*
- Una singola applicazione generica può generare o collezionare molti tipi e formati diversi di dati

Estrarre Conoscenza dai Dati

I vari tipi e formati di dati devono essere interoperabilmente connessi tra di loro

Velocità

- Set di Big Data vengono generati sempre più frequentemente, e pertanto devono essere anche processati velocemente

- Online Data Analytics

- *Monitoraggio, Detection e Decisioni Tardive*
  ➔ *Opportunità Mancate!*

Primi Tentativi di Large Scale Computing

- Tradizionalmente il calcolo computazionale è stato inteso come "*CPU-bound*"
  - Efficiente per volumi di dati relativamente ristretti

- Gli avanzamenti nella computer technology si sono incentrati inizialmente nel migliorare le prestazioni e la potenza di calcolo di una singola macchina

Avanzamento Tecnologico dei Processori

- Legge di Moore
  - Il numero di transistor presenti in un circuito integrato raddoppia mediamente ogni due anni

- Si arriva al limite della tecnologia → la computazione Single-core non riesce a scalare secondo le attuali necessità computazionali

Limitazioni Architetture Single-Core

L'elevato consumo di potenza limita l'aumento delle prestazioni (in termini di velocità computazionale) data dalla maggiore densità dei transistor

Sistemi Distribuiti

Permettono agli sviluppatori di usare più macchine per processare un singolo task

Quali Soluzioni Tecnologiche !?

- Soluzioni Big Data sono nativamente più efficienti in ambito real-time delle tradizionali applicazioni Data Warehouse (DW)

- Architetture DW architectures (ad es. Oracle Exadata → *shared everything*, Teradata → *shared nothing*) non risultano adatte per operare sui Big Data (sono più efficienti su dataset strutturati)

- Architetture scalabili, Massive Parallel Processing sono soluzioni che si adattano in maniera più efficiente ad operare sui Big Data.

Challenges:

- **Il Collo di Bottiglia sta nella Tecnologia…**
  - Sono necessary nuovi tipi di architettura, nuovi algoritmi e tecniche di elaborazione

- **…ma anche nelle capacità tecniche di programmazione**
  - Acquisire esperienza nell'utilizzo delle nuove tecnologie e nella gestione dei Big Data

UNIVERSITÀ DEGLI STUDI FIRENZE

DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

Hadoop

MASTER: Big Data Analytics And
Technologies For Management - MABIDA

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

MASTER: Big Data Analytics And
Technologies For Management - MABIDA

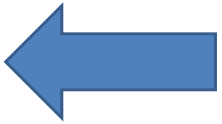
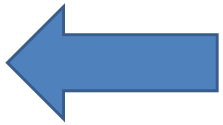
- Hadoop è una piattaforma Open Source progettata e realizzata per l'elaborazione, analisi e archiviazione di grandi quantità di dati distribuiti e non strutturati.

- Hadoop è stato progettato per essere fortemente scalabile: da una singola fino a migliaia di macchine (*nodi*), con un elevato grado di *fault tolerance*.

- Con Hadoop è possibile analizzare ed elaborare vasti dataset attraverso processi batch distribuiti e altamente scalabili.

- Hadoop è nato dal framework MapReduce, implementato agli inizi del 2000 da Google, alla base della sua architettura distribuita per l'indicizzazione delle pagine html su Web.
- Attualmente Hadoop è un progetto Open Source di Apache Software Foundation → centinaia di contributi che migliorano continuamente la ***core technology***
- Idea chiave: invece di processare un ingente blocco di dati con una singola macchina in una volta sola, Hadoop "frammenta" Big Data in varie parti in modo da poter processare ogni parte parallelamente dalle varie macchine che compongono l'architettura (***cluster***).

Principali Usi di Hadoop

- Data-intensive text processing
- Assembly of large genomes
- Graph mining
- Machine learning and data mining 
- Large scale social network analysis 

Chi Usa Hadoop?

Hadoop Ecosystem

Hadoop Common → Contiene le librerie e i moduli principali

HDFS → File System distribuito di Hadoop

Hadoop MapReduce → Paradigma di programmazione di Hadoop per processare grandi quantità di dati

Sistemi Distribuiti: Limitazioni

- La programmazione di un Sistema distribuito è generalmente più complessa
  - Sincronizzazione dello scambio di dati
  - Gestione di una banda limitata
  - Controllo dei tempi computazionali

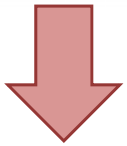
Sistemi Distribuiti: Limitazioni

"You know you have a distributed system when the crash of a computer you've never heard of stops you from getting any work done." – Leslie Lamport

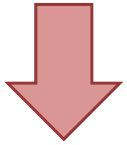
- E' necessario progettare un Sistema Distribuito aspettandosi fallimenti nello scambio e archiviazione distribuita dei dati.

Sistemi Distribuiti: Data Storage

- Tipicamente sono previste risorse specifiche per l'archiviazione dei dati (*Data Nodes*) e per la parte computazionale (*Compute Nodes*)



Durante l'esecuzione di un processo, I dati vengono copiati nei Compute Nodes



OK per quantità di dati non eccessive

Quanti Dati ?

- Facebook
  - 500 TB al giorno
- Yahoo
  - Oltre 170 PB
- eBay
  - Oltre 6 PB

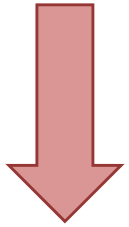
- Il collo di bottiglia risiede nel copiare i dati verso i nodi adibiti alla parte computazionale

Requisiti di Hadoop

- Deve essere scalabile
- Deve prevedere una politica di gestione dei fallimenti

Peculiarità di Hadoop

"Moving Computation is Cheaper than Moving Data"



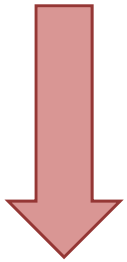
Non vengono copiati i dati verso la parte computazionale; al contrario, la parte computazionale viene spostata vicino alle porzioni di dati da processare

Problematiche Tipiche

- Elevato numero di nodi nel cluster
- Probabilità significativa di fallimento:
  - CPU: centinaia o migliaia di nodi presenti
  - Rete: una congestione della rete o un fallimento/guasto di un apparato di rete possono portare a non fornire i risultati e di dati attesi in output nei tempi previsti.
  - Memoria: Rischio di "*Run out of space*"
  - Archiviazione: Rischio di "*Run out of space*", fallimento di un nodo, corruzione di dati, fallimento nella trasmissione dei dati
  - Clock: mancanza di sincronizzazione tra parti di dati, file ecc… con conseguente probabile perdita di consistenza
- Specifiche strategie di computazione parallela offrono support per la gestione e il recupero di alcuni di questi fallimenti

Hadoop: Gestione dei Fallimenti

Il fallimento di un singolo nodo o componente non deve causare il fallimento dell'intero Sistema, ma al Massimo un decadimento delle prestazioni dell'applicazione



Un fallimento non deve causare una perdita di dati !!

Component Recovery

- Se un componente introduce un failure, dovrebbe anche essere in grado di recuperarlo "a caldo", cioè senza riavviare l'intero sistema.

- Il fallimento di un componente o le operazioni di recovery non devono alterare il risultato finale in output.

- Hadoop ha cambiato le dinamiche della computazione su grandi quantità di dati, definendo una soluzione che risulta:
  - **Scalabile:** si possono aggiungere nuovi nodi all'occorrenza, senza la necessità di cambiare il format dei dati, le modalità di caricamento dei dati o il funzionamento degli applicativi.
  - **Efficiente nei costi:** Hadoop ha portato la computazione parallela di larga scala sui cosiddetti *commodity servers*, con un drastico abbassamento dei costi per Terabyte di storage.
  - **Flessibile:** Hadoop risulta essere schema-less, e può gestire ogni tipo di format di dati, strutturato o non strutturato, da un qualsiasi numero di sorgenti. Dati proveniente da fonti diverse possono essere uniti e aggregati, permettendo un'analisi e una gestione più approfondita di quanto sia possibile con un Sistema non distribuito.
  - **Tolleranza ai fallimenti:** Quando si ha un fallimento di un nodo, il Sistema ridireziona il lavoro su un'altra porzione di dati a caldo, senza interrompere il workflow di processing.

Hadoop - Problematiche Ricorrenti

- Hadoop non prevede un modello di sicurezza o salvaguardia da malware.
  - E' progettato per gestire in maniera robusta solo:
    - hardware failure
    - Problemi di congestione del traffico dati
- Lo spazio di archiviazione potrebbe localmente esaurirsi:
  - Necessario meccanismi di reindirizzamento e redistribuzione
  - Necessaria una sincronizzazione tra i vari nodi
  - Questo può causare:
    - Congestione della rete
    - Stallo durante lo scambio di dati

Recupero da failure

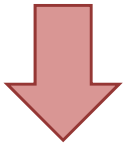
- Se qualche nodo fallisce nel fornire i dati nei tempi previsti, gli altri nodi dovrebbero completare il lavoro del nodo mancante

- Questo processo di Recovering deve essere automatico

- Limiti fisici:
  - Singoli hard disk possono sostenere velocità di lettura tra 60 e 100 MB al secondo;
  - Considerando ad esempio 4 canali I/O indipendenti per il cluster distribuito, questo riesce a trasmetterere un totale di 400 MB al secondo.

Processi vs Dati

- I vari processi operano su porzioni specifiche dei dati. L'allocazione dei singoli processi dipende dalla posizione di queste porzioni di dati a cui devono accedere e che devono elaborare.



Data locality

  - Minimizza il flusso di dati attraverso i nodi
  - Evita comunicazioni tra i nodi per servire i nodi adibiti alla computazione
  - Uno dei fondamenti peculiari di Hadoop sta nel fatto di spostare la parte computazionale verso i dati e ***NON*** viceversa !

Scalabilità di Hadoop

- Un aumento significativo delle prestazioni nell'utilizzo di un cluster Hadoop potrebbe non essere significativo con un numero limitato di nodi.
- Paradigmi di programmazione parallela/distribuita come MPI (Message Passing Interface) potrebbero avere prestazioni migliori su un cluster di 2,4, fino a una decina di nodi.
- Hadoop è appositamente progettato per avere una curva di scalabilità lineare.
- Se un processo è scritto e lavora per (ad esempio) 10 nodi, può lavorare altrettanto efficientemente per migliaia di nodi senza grandi sforzi sulla rete.

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

Hadoop Architecture

- L'architettura distribuita di Hadoop è di tipo ***master-slave*** ed è costituita da due component principali**:**
  - **HDFS** Hadoop Distributed File System (HDFS) per l'archiviazione.
  - **MapReduce** paradigma di computazione parallela.

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

HDFS - Panoramica

- L'HDFS è responsabile dell'archiviazione dei dati nel cluster
- I file contenenti dati sono suddivisi in blocchi e distribuiti tra i vari nodi del cluster
- Ogni blocco è replicato un numero di volte che è definito in un file di configurazione (ridondanza per la gestione dei fallimenti)

HDFS – Concetti Basilari

- HDFS è un file system scritto in Java basato sul Google GFS

- Offre funzionalità di archiviazione ridondata per grandi quantità di dati.

HDFS – Concetti Basilari

- HDFS lavora meglio con un *piccolo* numero di *grandi* file
  - Tipicamente, 100MB e oltre per ogni file
- I file all'interno di HDFS vengono scritti una volta sola
- HDFS è ottimizzato per leggere streaming di file di grandi dimensioni (e non per letture ad accesso casuale)

Archiviazione dei File

- I file sono suddivisi in blocchi
- I vari blocchi sono a loro volta suddivisi tra I vari nodi in fase di caricamento
  - Tipicamente, blocchi differenti di uno stesso file vengono archiviati Different in nodi diversi
- I blocchi sono replicati (ridondanza) su più nodi
- Il Sistema tiene traccia di quali blocchi costituiscono un intero file, e dove essi sono archiviati

NameNode

- Memorizza le informazioni di HDFS in un immagine del filesystem
- Gli aggiornamenti (Aggiunta/rimozione blocchi) non cambiano l'immagine del filesystem
  - Sono invece scritti su log file
- Quando il namenode viene fatto partire, carica l'immagine del filesystem e applica i cambiamenti presi dal log file

Secondary NameNode

- **Non è un nodo di backup**

- Legge periodicamente i file di log e applica i cambiamenti all'immagine del filesystem aggiornandolo

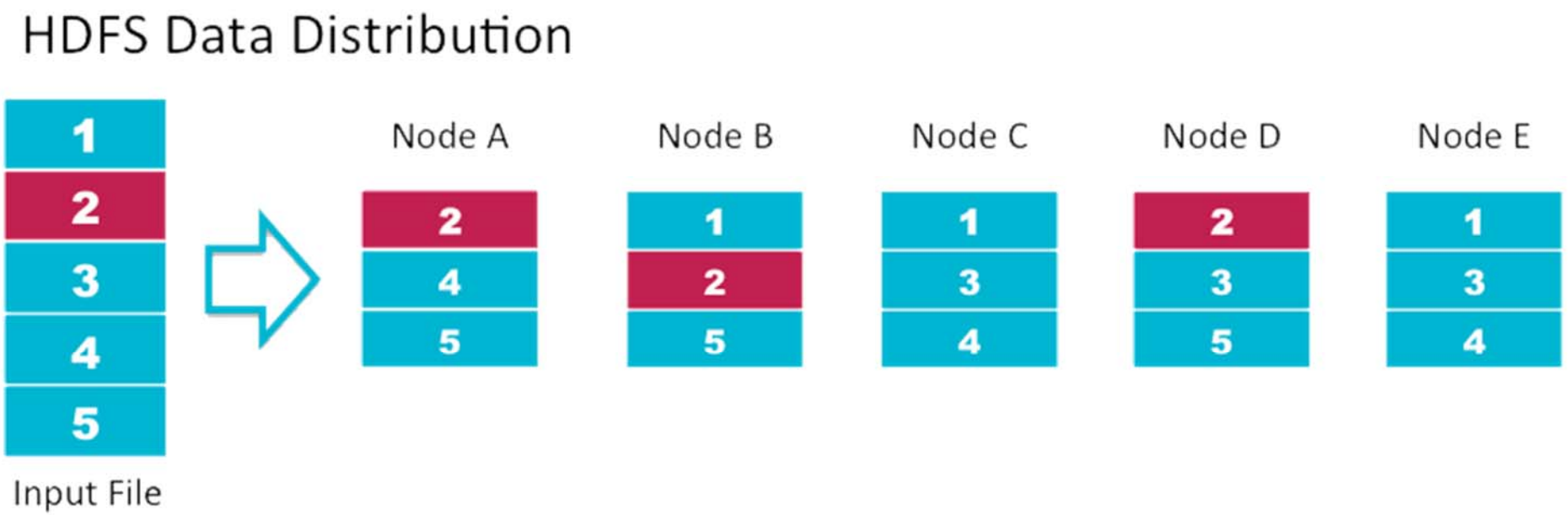
- Permette di far ripartire il namenode più velocemente

Datanode

Nodo dove vengono fisicamente allocati i dati

Replicazione dei Dati

- Di Default, la replicazione dei dati prevede la creazione di 3 copie di ridondanza.



Recupero dei Dati

- Quando un client vuole recuperare dei dati:
  - Comunica con il Nodo principale (master) per stabilire quali blocchi compongono il/i file richiesti e su quali nodo sono archiviati.
  - Successivamente, il client comunica direttamente con quei nodi per leggere i dati richiesti.

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

MASTER: Big Data Analytics And
Technologies For Management - MABIDA

MapReduce - Panoramica

- MapReduce è un paradigma di programmazione che distribuisce la parte computazionale su più nodi
- Ogni nodo processa i dati archiviati nel nodo stesso.
- MapReduce consiste di due fasi principali:
  - Map
  - Reduce

MapReduce - Funzionalità

- Parallelizzazione e Distribuzione automatica
- Politiche di Tolleranza ai fallimenti / errori
- Offre un elevato grado di astrazione per il programmatore

UNIVERSITÀ DEGLI STUDI FIRENZE

DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

How Hadoop Works

MASTER: Big Data Analytics And
Technologies For Management - MABIDA

Mapper

- Legge e interpreta i dati come coppie chiave/valore
- Restituisce in uscita coppie chiave/valore

Fase Shuffle and Sort

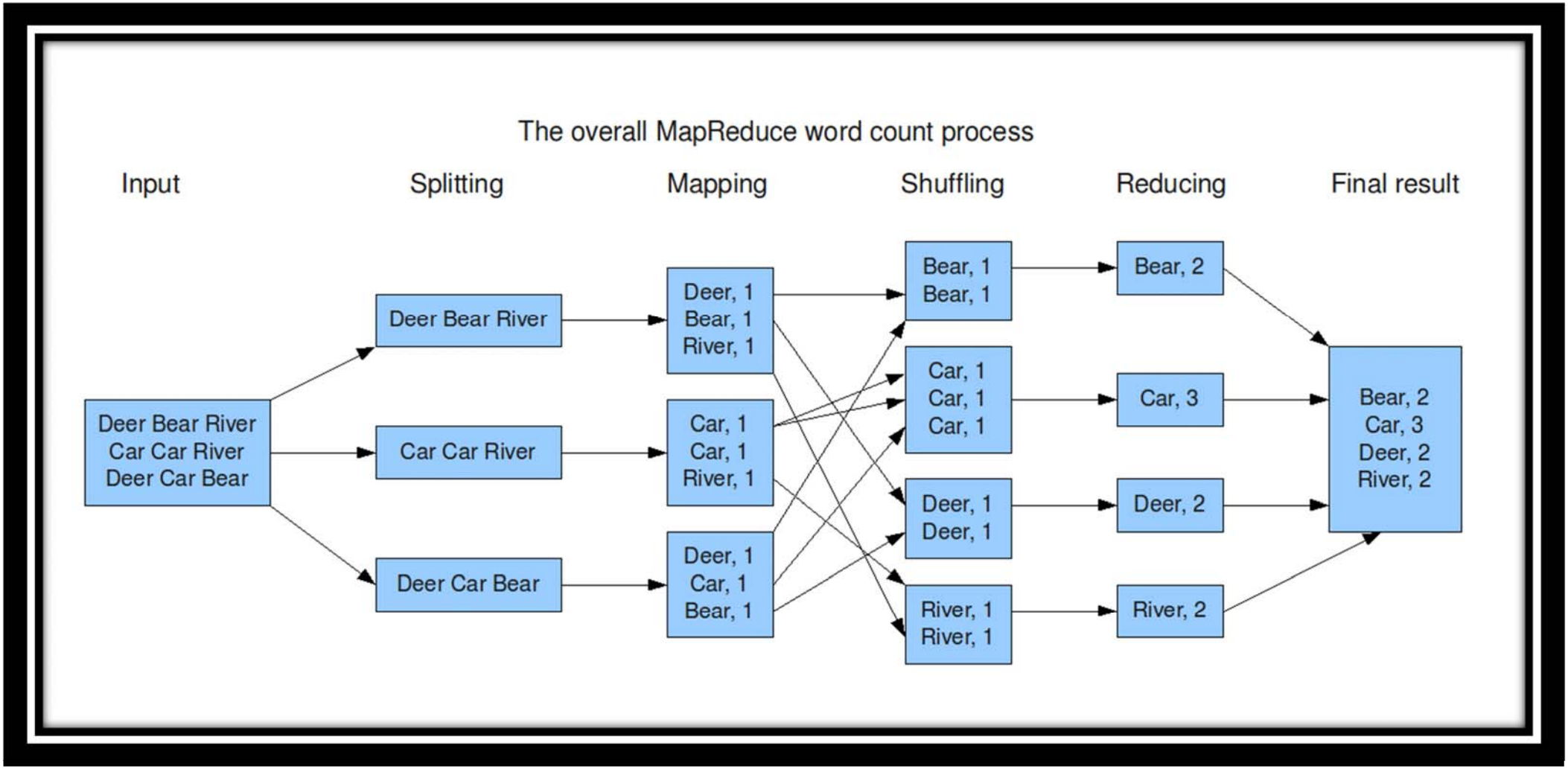
- I dati di Output in uscita dal mapper sono ordinati per chiave

- Tutti i valori con la medesima chiave vengono indirizzati ad uno stesso nodo per la successiva elaborazione (*reduce*)

Reducer

- Invocato una sola volta per ogni chiave unica
- Riceve in ingresso una lista con tutti i valori associate alla chiave di ingresso
- Restituisce in output le coppie chiave/valore finali

MapReduce: Conteggio di Parole



JobTracker and TaskTracker

- JobTracker
  - Stabilisce il piano di esecuzione dei job
  - Assegna I task individuali
- TaskTracker
  - Tiene traccia delle performance di ogni singolo mapper e reducer

Pros & Cons

Pros

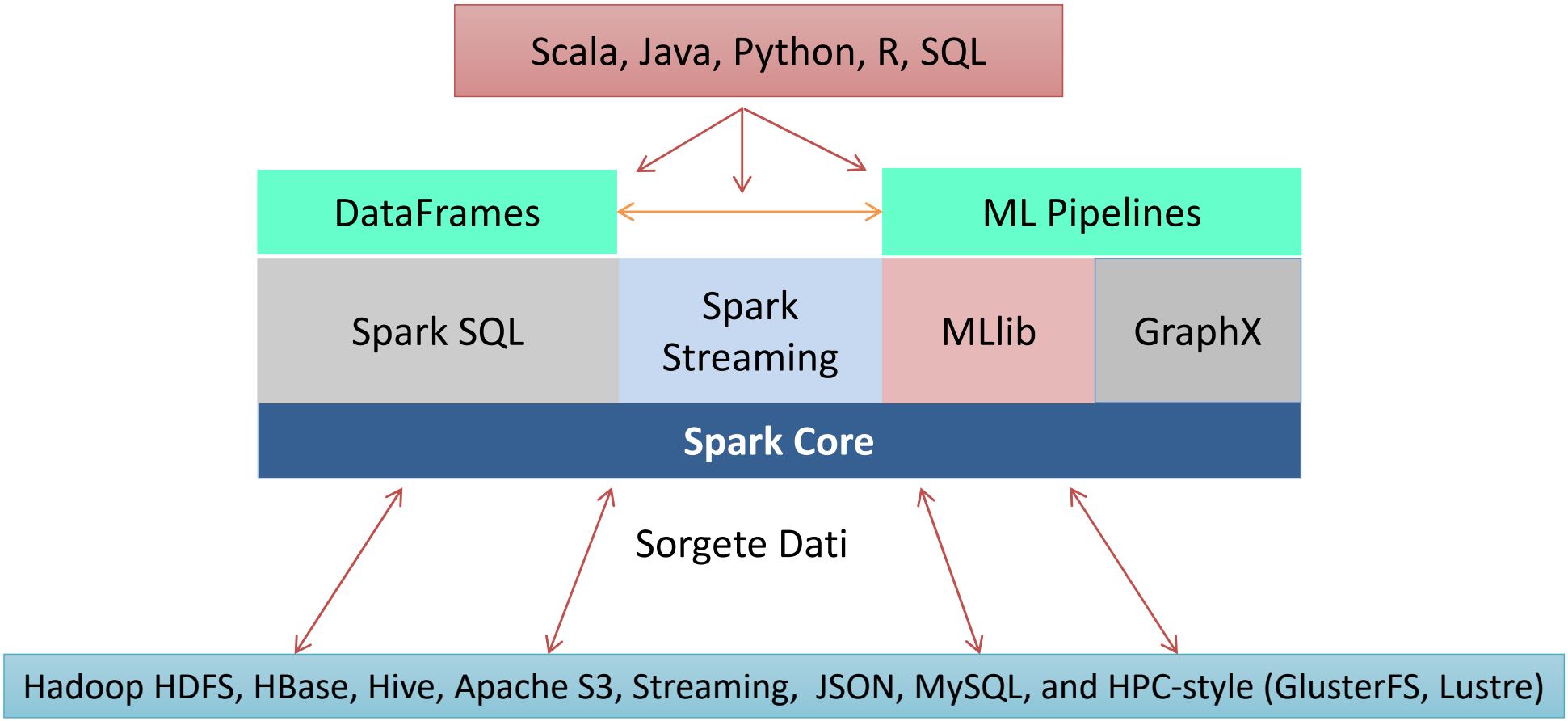
- Permette alle aziende di analizzare grandi volumi di dati non strutturati o semi strutturati, in maniera economica e in tempi ragionevoli
- I cluster Hadoop possono scalare verso l'ordine di petabytes; Le aziende possono processare e analizzare **tutti** i dati rilevanti (non dei campioni)
- Gli sviluppatori possono scaricare il software e sperimentare delle demo in meno di un giorno

Cons

Implementare applicazioni e gestire Hadoop per eseguire analisi avanzate su grandi volumi di dati non strutturati richiede esperienza, competenza e una buona preparazione

Apache Spark

Supporta data analisi, machine learning, grafici flusso di dati, etc. Può leggere/scrivere da una varietà di dati e permette di sviluppare in più linguaggi.



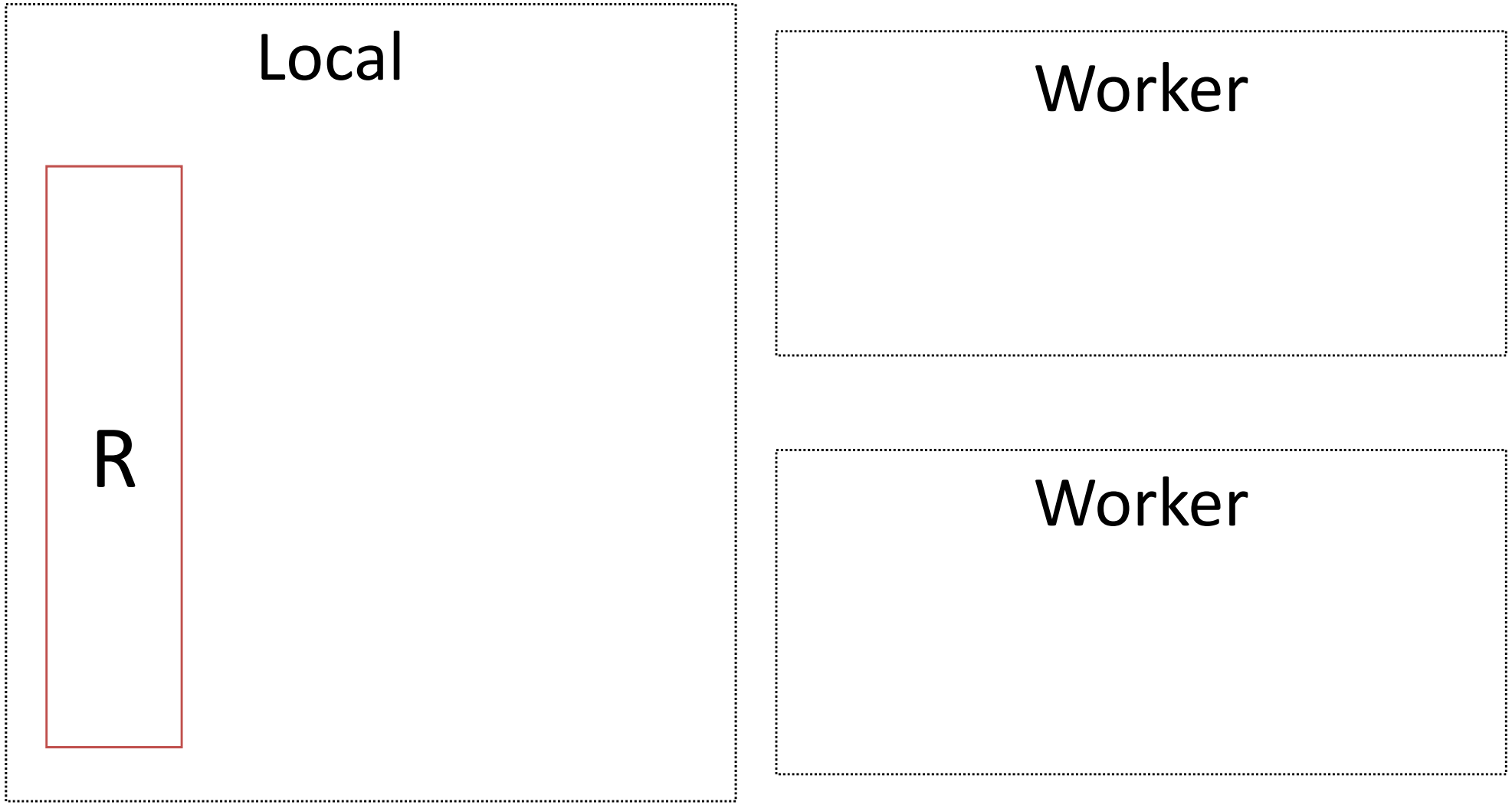
Resilient Distributed Datasets (RDDs)

- RDDs (Resilient Distributed Datasets) sono contenitori dati
- Tutti i processi differenti di Spark condividono lo stesso RDD
  - Si possono mischiare differenti transformazioni per creare nuovi RDD
  - Queste sono create parallelizzando collezioni esistenti o leggendo files
- Fault tolerant

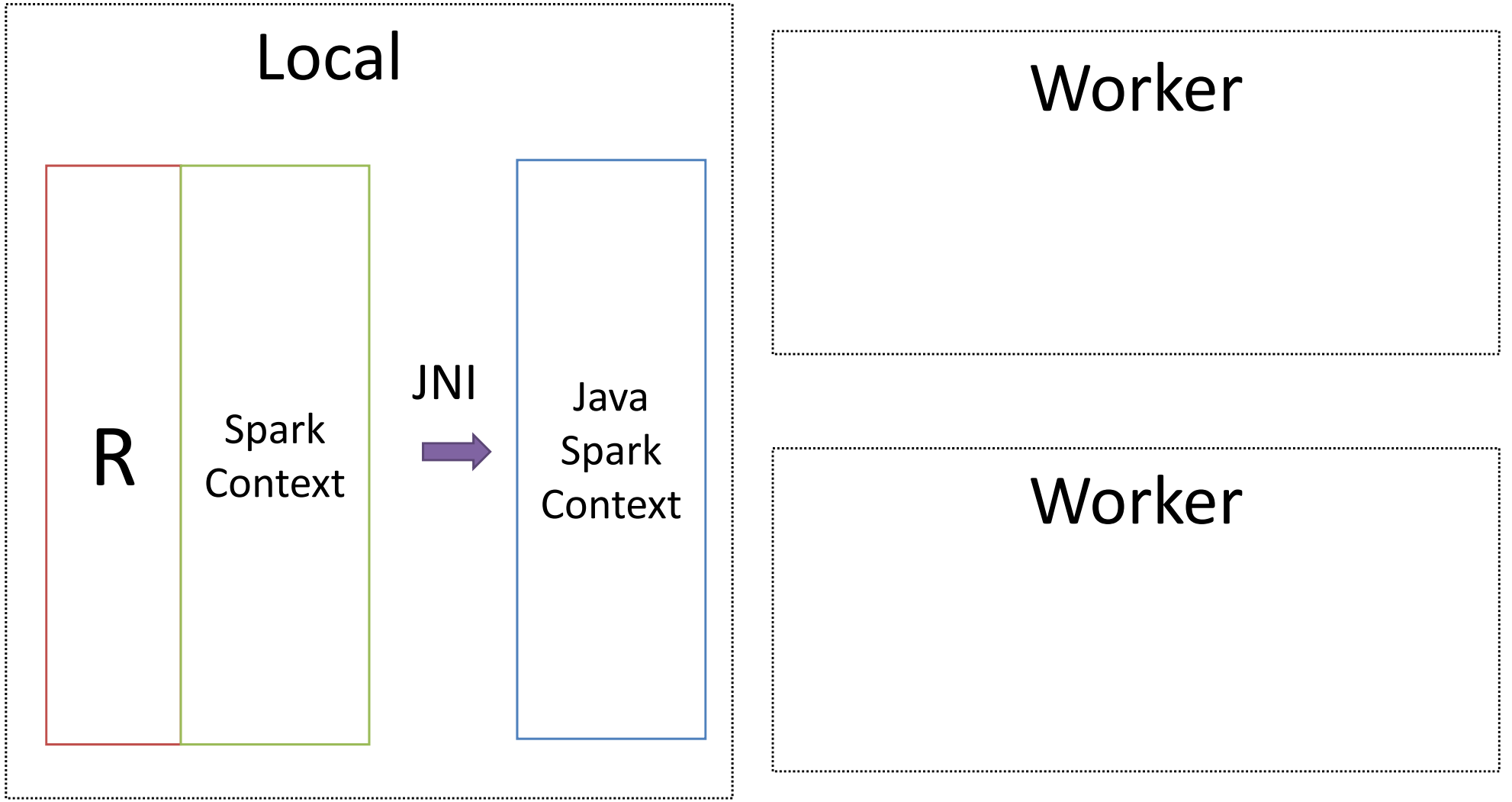
Operazioni di Spark

Trasformazioni (creazione RDD)	map filter sample groupByKey reduceByKey sortByKey intersection	flatMap union join cogroup cross mapValues reduceByKey
Azioni (ritorna un risultato al programma chiamante)	collect Reduce Count takeSample take lookupKey	first take takeOrdered countByKey save foreach

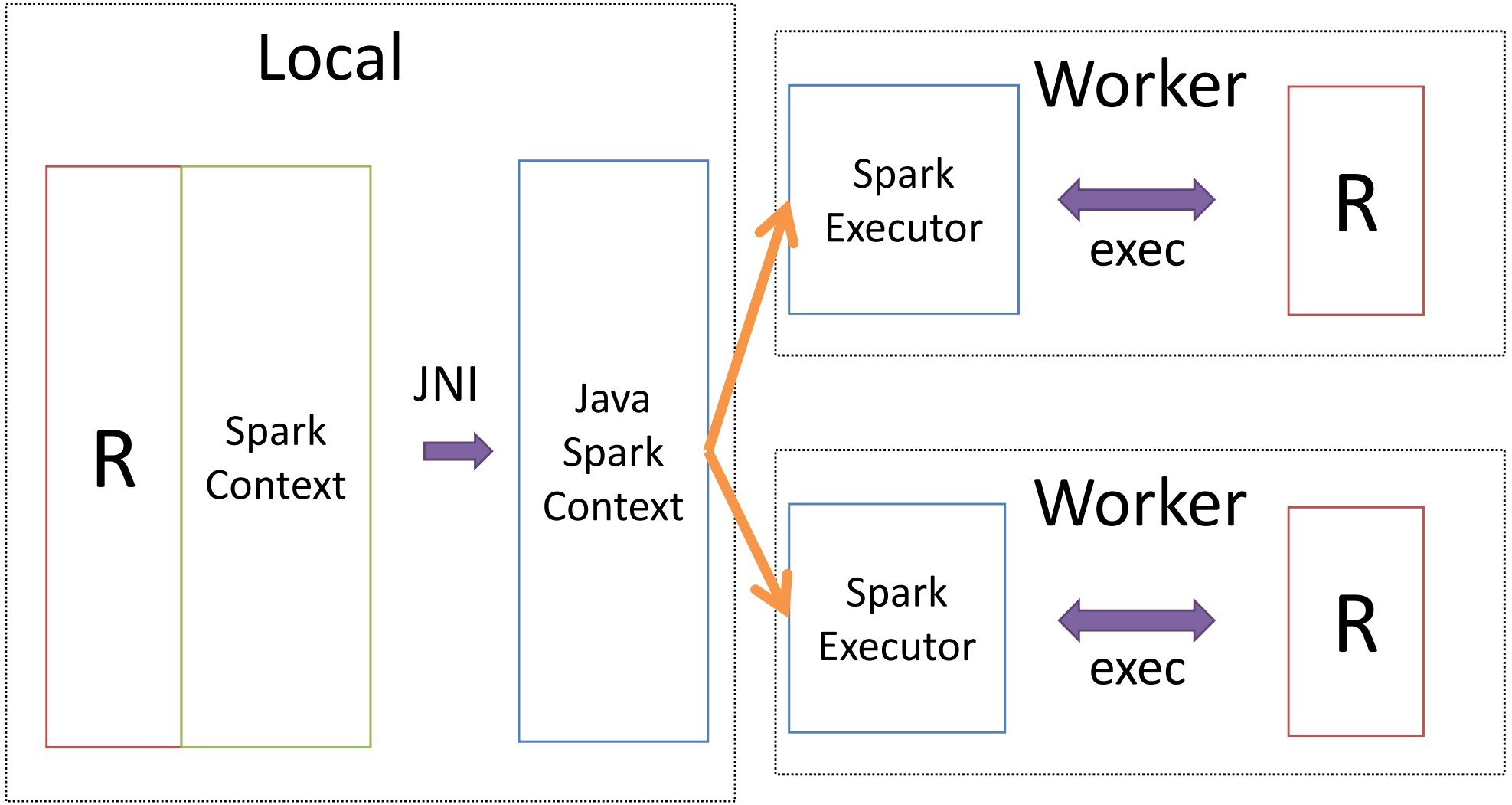
Dataflow



Dataflow



Dataflow


Local
R
Spark Context
JNI
Java Spark Context
Worker
Spark Executor
exec
R
Worker
Spark Executor
exec
R

fine

fine