BERLIN BIG DATA CENTER

The Stratosphere Platform: Big Data Analytics at Scale

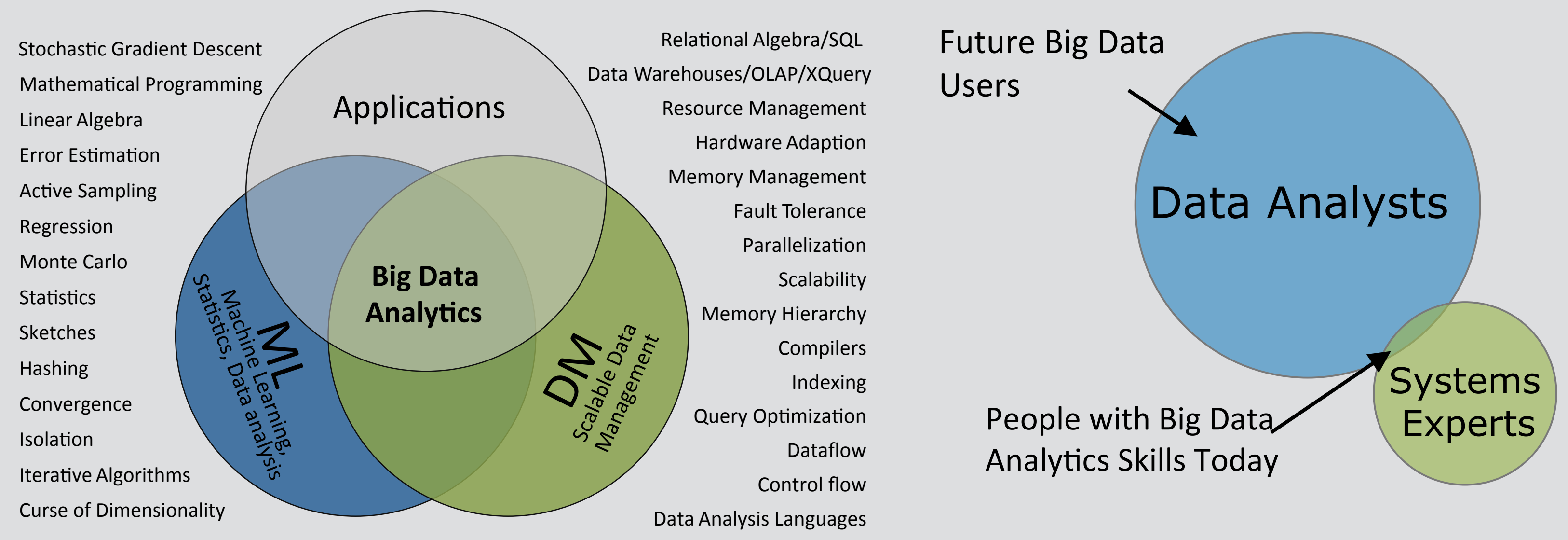
Database Systems and Information Management, Technische Universität Berlin

Big Data looks tiny from Stratosphere

Open Source Platform for Big Data analytics in massively parallel cluster and cloud environments. Combines key technologies of **MapReduce**, **Compilers**, **Distributed Systems** and **Parallel Database Systems**.

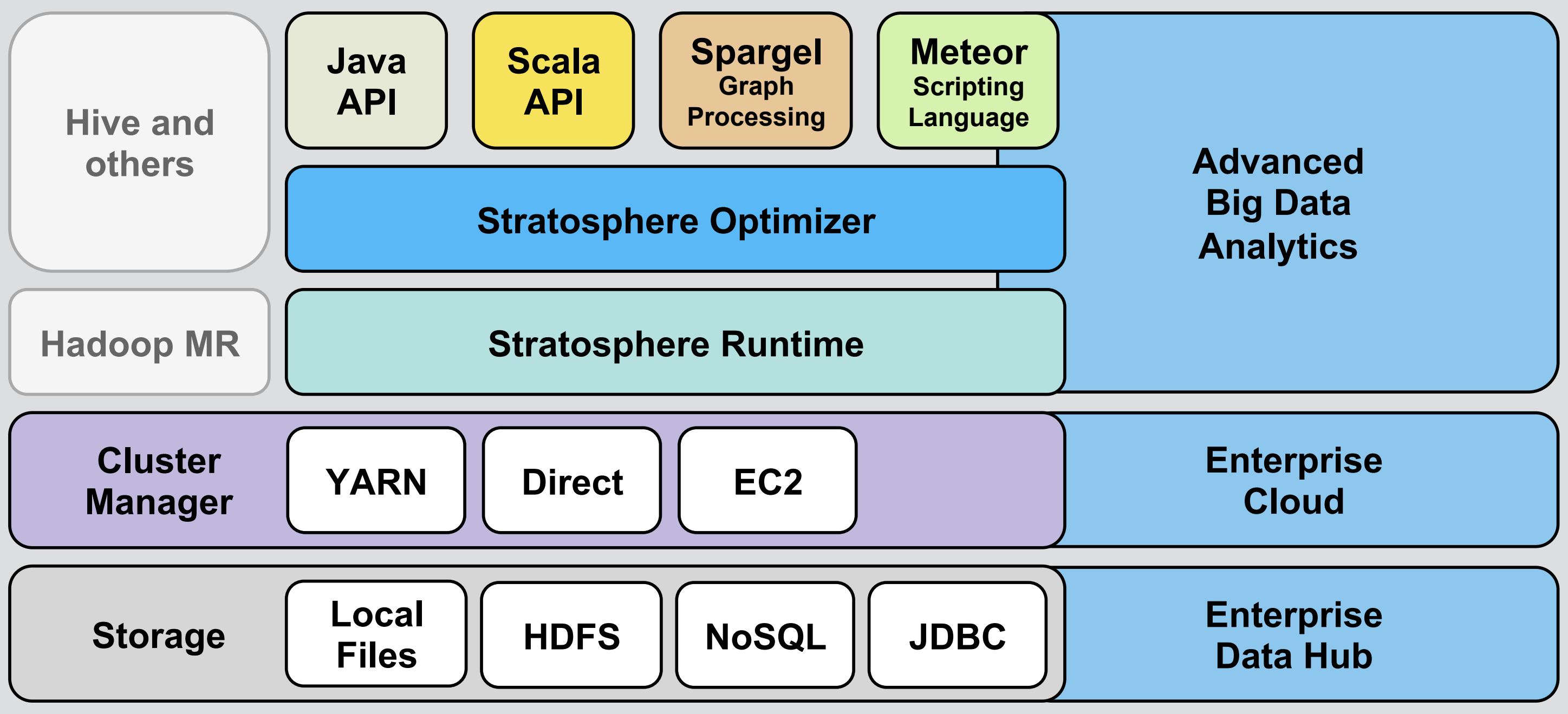
Enables scalable data management, machine learning and deep analysis of Big Data.

Big Data Analysts Still Hard to Find



Our goal: empower data analysts to focus on their domain problem without requiring systems programming.

Software Stack


Hive and others

Java API

Scala API

Spargel Graph Processing

Meteor Scripting Language

Stratosphere Optimizer

Advanced Big Data Analytics

Hadoop MR

Stratosphere Runtime

Cluster Manager

YARN

Direct

EC2

Enterprise Cloud

Storage

Local Files

HDFS

NoSQL

JDBC

Enterprise Data Hub

Programming Model/Operators

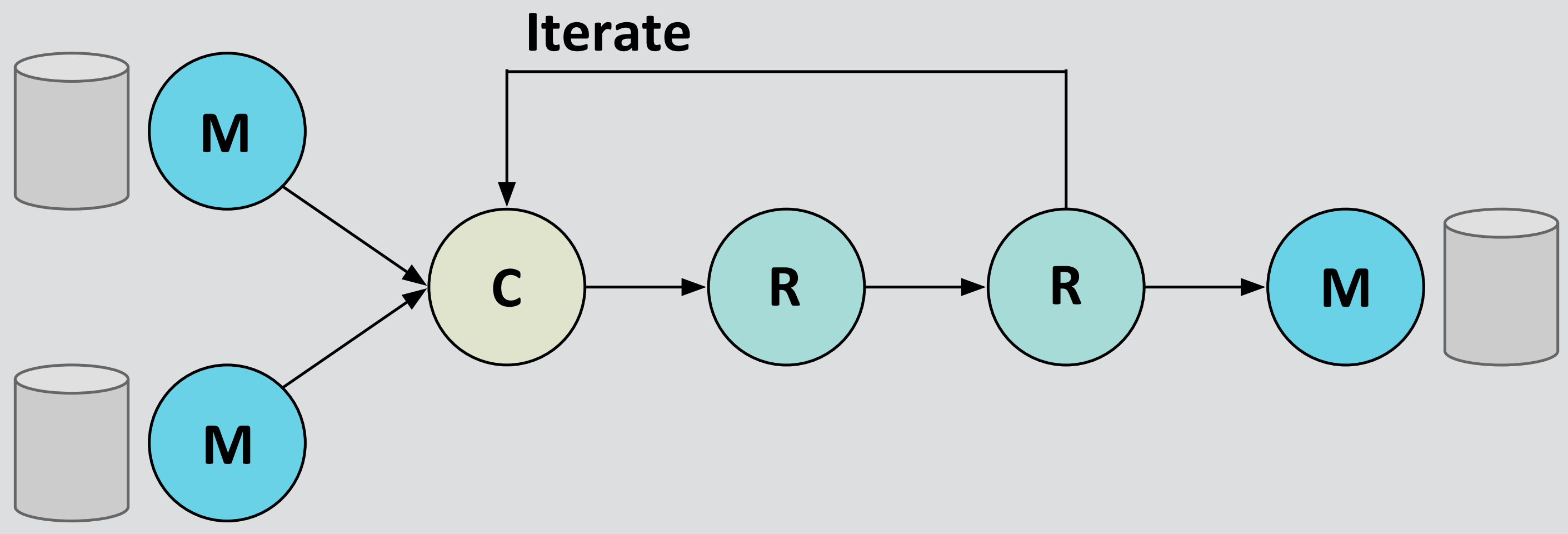
Stratosphere extends the well-known MapReduce model with new operators. These operators represent many common data analysis tasks more naturally and efficiently. All operators will start working in memory and gracefully go out of core under memory pressure.

map | reduce | join | cross

union | iterate | iterate delta | cogroup

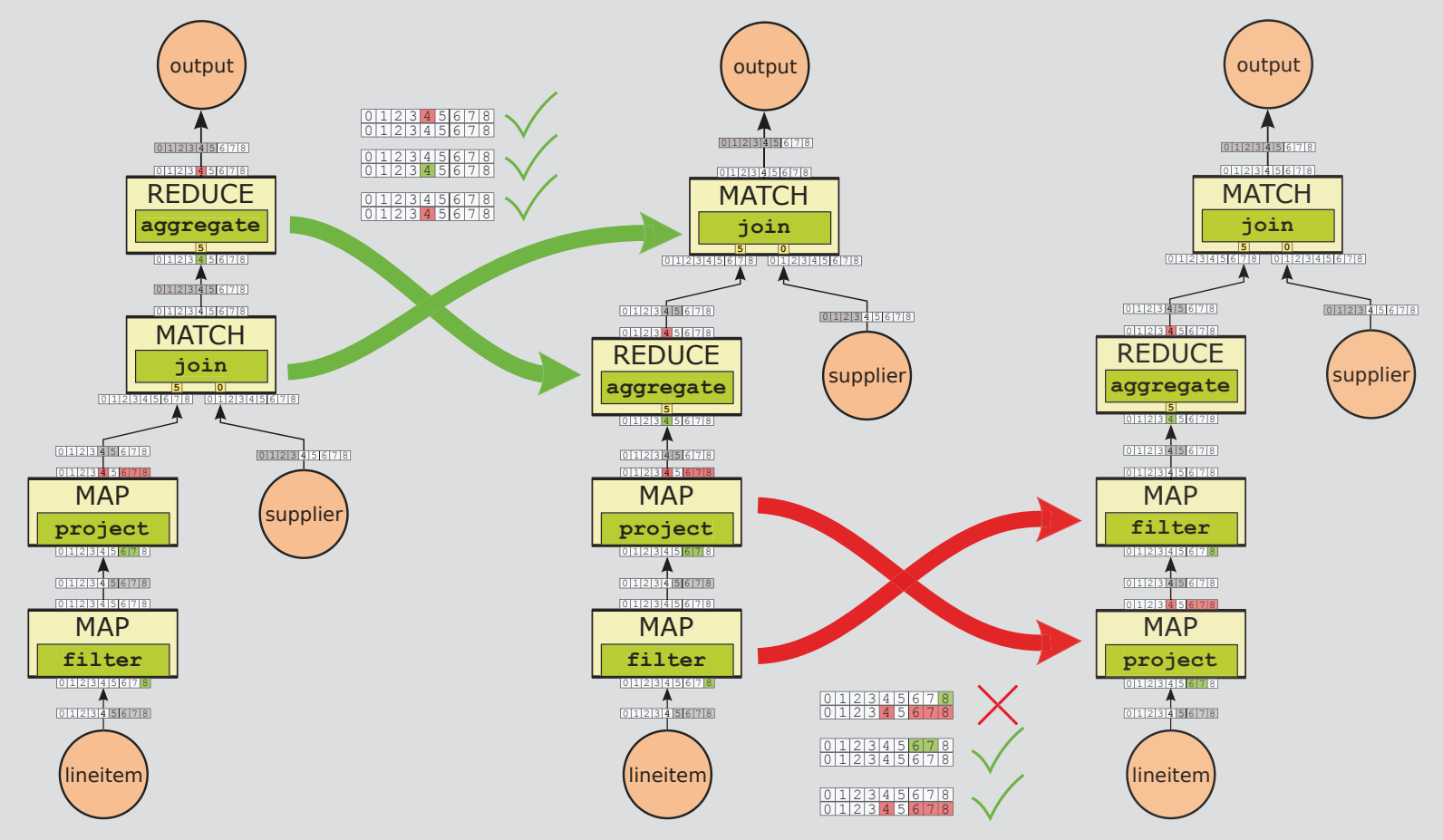
Iterative Algorithms

Incremental Iterations can exploit sparse computation dependencies without sacrificing dataflow programming abstraction. The performance of incremental iterations in Stratosphere matches that of specialized engines.



Built-In Optimizer

- Cost-based optimizer choice of operators and shipping strategies.
- In-memory pipelining of operators
- Reduction of shipped and written data volume
- Input Sampling to determine cardinalities



Example: KMeans Clustering (Scala)

```
val dataPoints = DataSource(dataPointInput, DelimitedInputFormat(parseInput))
val clusterPoints = DataSource(clusterInput, DelimitedInputFormat(parseInput))

def computeNewCenters(centers: DataSet[(Int, Point)]) = {
   val distances = dataPoints.cross(centers)
                             map computeDistance
   val nearestCenters = distances.groupBy { case (pid, _) => pid }
                             .reduceGroup { ds => ds.minBy(_._2.distance) }
                             .map asPointSum tupled
   val newCenters = nearestCenters groupBy { case (cid, _) => cid }
                             .reduceGroup(sumPointSums)
                             .map { case (cid, pSum) => cid -> pSum.toPoint() }
   return newCenters
}
val finalCenters = clusterPoints.iterate(numIterations, computeNewCenters)
```

Contact & Further Information

Participating in the
Google Summer of Code 2014

Contact Us:
e-mail: contact@stratosphere.eu

More Information:
Project: http://stratosphere.eu
Source Code: http://github.com/stratosphere

Current and Future Work

Declarative Analytics

- Empower data analysts to use Big Data by unifying data and programming models in a declarative abstraction
- Cross-optimize data extraction, querying and modeling

Data Streaming

- Streaming semantics for advanced analytical functions
- Modeling and managing distributed operator state
- Optimizing data analysis program workloads

Iterative Processing

- Fault tolerance and numerical stability for iterative algorithms
- Advanced optimization techniques for iterative jobs

Funding Organisations

Partners and Collaborations