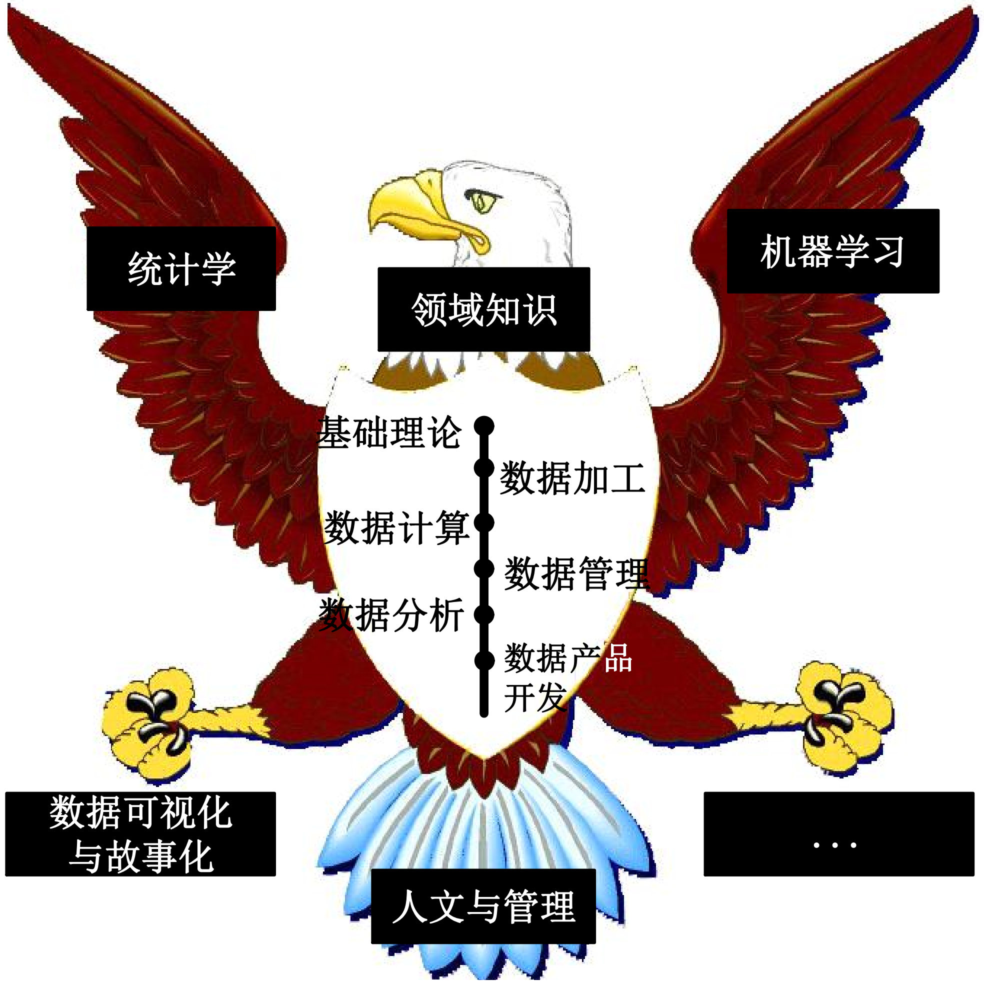
第6章 大数据技术

编写思路

学习方法、要求及建议

疑难知识点的解读

1.本章定位与内容简介



- 6.1 数据科学与大数据技术
- 6.2 Hadoop 生态系统
- 6.3 大数据计算技术与Spark
- 6.4 大数据管理技术与MongoDB
- 6.5 大数据分析技术与Python
- 6.6 Python 编程实践
- 6.7 继续学习本章知识
- 习题

2.本章学习提示及要求

了解

- 数据科学中常用的大数据技术类型
- Hadoop生态系统
- Spark生态系统

理解

- 大数据计算技术及其特征
- 大数据管理技术及其特征
- 大数据分析技术及其特征
- 大数据分析中的陷阱及其应对

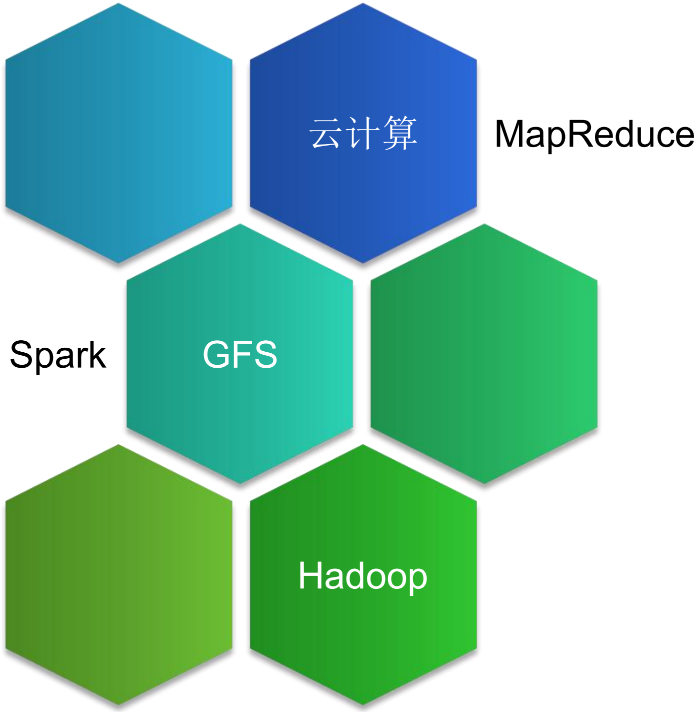
掌握

- Gartner分析学价值扶梯模型
- Lambda架构
- CAP理论与BASE原则
- NoSQL的数据模型
- 分片技术与复制技术
- Analytics3.0

熟练掌握

- Spark技术的原理及Python编程
- MongoDB技术的原理及Python编程
- Spark+MongoDB+Python+MLib的综合应用

3.几个核心概念的区别



云计算

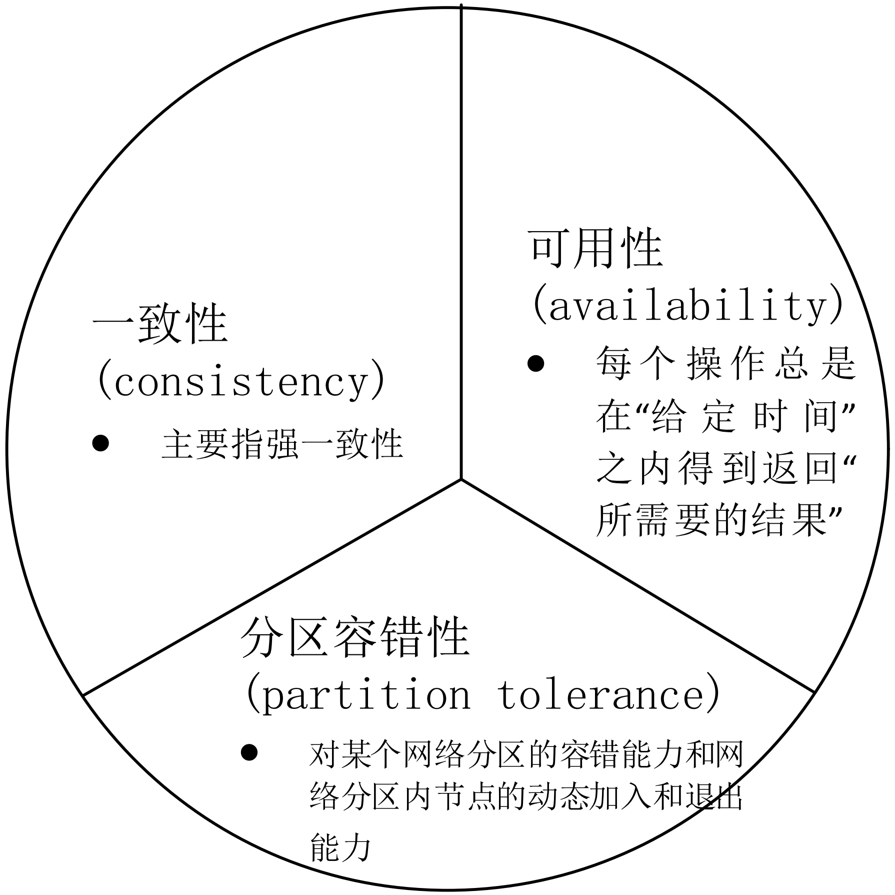
MapReduce

Spark

GFS

Hadoop

4.CAP理论



一致性(consistency)

- 主要指强一致性

可用性(availability)

- 每个操作总是在“给定时间”之内得到返回“所需要的结果”

分区容错性(partition tolerance)

- 对某个网络分区的容错能力和网络分区内节点的动态加入和退出能力

CAP理论

Cassandra，Dynamo

- 选择AP（放弃C）

BigTable，MongoDB

- 满足CP（放弃A）

Mysql和Postgres

- 满足AC（放弃P）

5.BASE原则

- BASE原则可理解为CAP原则的特例
- NoSQL实际应用中需要权衡一致性与可用性

Basically Available

- 是指可以容忍系统的短期不可用，并不追求全天候服务

Soft State

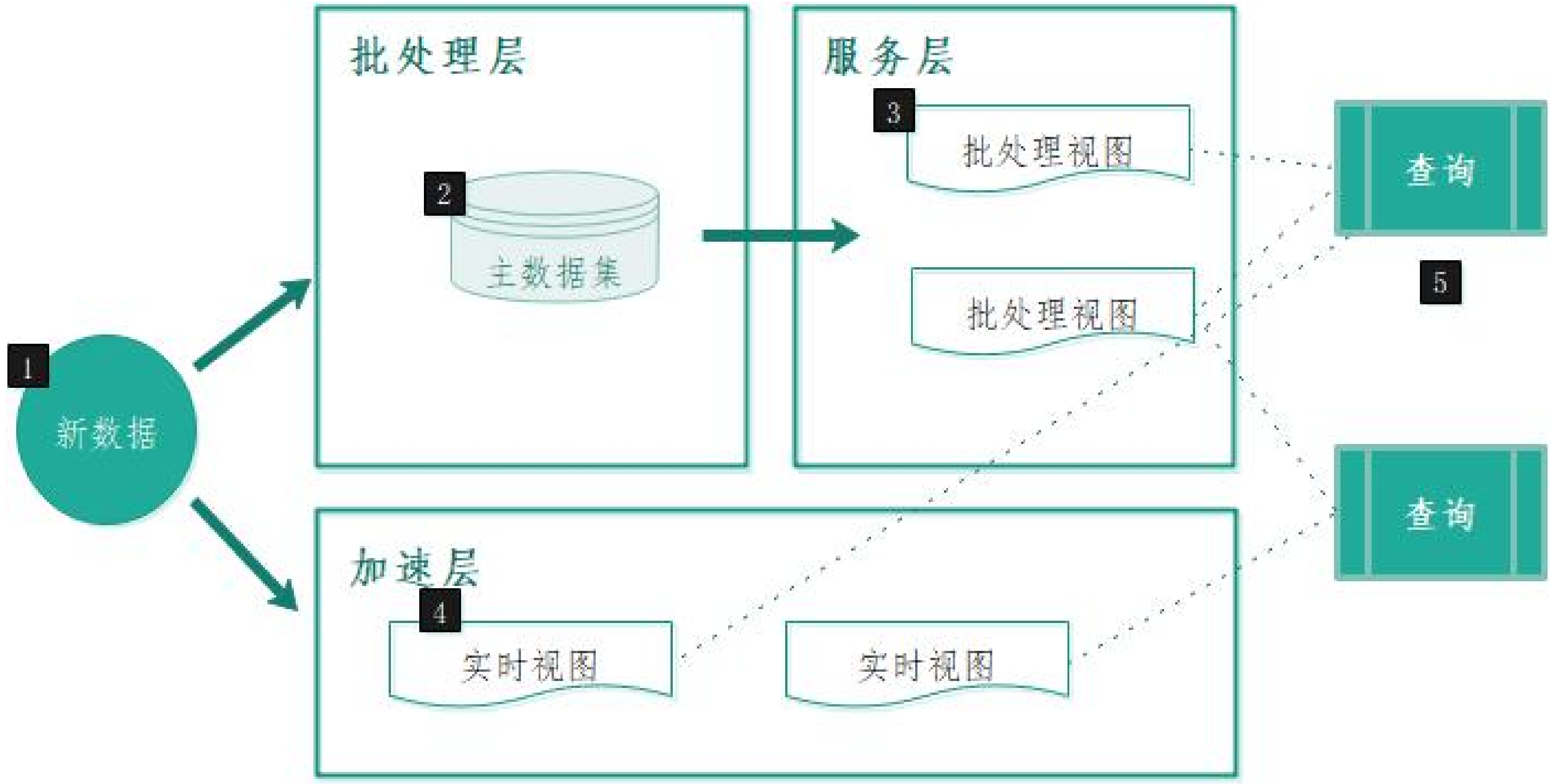
- 是指不要求一直保持强一致状态

Eventually Consistent

- 是指最终数据一致，而不是严格的实时一致，系统在某一个时刻后达到一致性要求即可

6.Lambda 架构（Lambda Architecture

可靠性和实时性之间矛盾
Storm创始人Nathan Marz，结合Twitter和BackType的工作经验



7.Spark

MapReduce 的局限

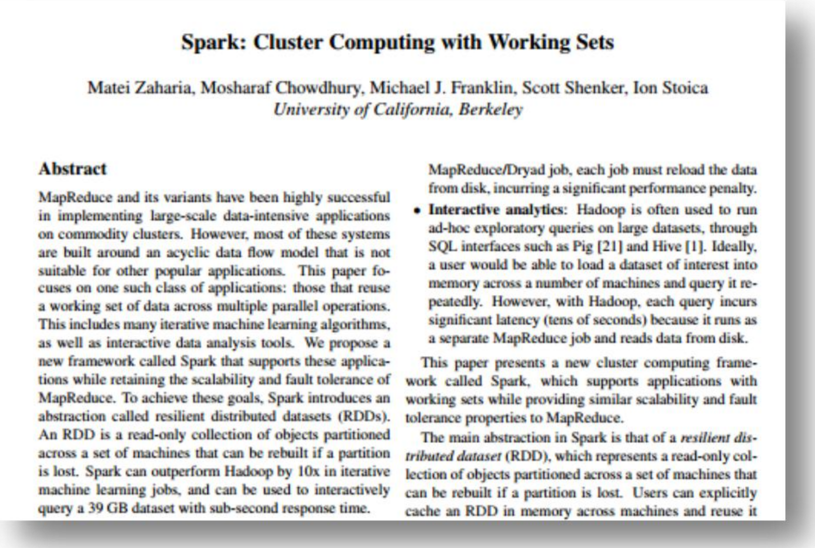
- 在MapReduce中直接编程难度大
- 仅适用于批处理
- 不适用于流计算、交互计算、图计算

解决思路

- 面向特定任务的专用系统，如Storm（实时计算）, Impala（交互分析）, Giraph（图计算）等
- 融合式通用系统，如Spark

Spark的出现

- 2010： Spark 论文
- 2014：Apache Spark 顶级



Spark: Cluster Computing with Working Sets

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley

Abstract

MapReduce and its variants have been highly successful in implementing large-scale data-intensive applications on commodity clusters. However, most of these systems are built around an acyclic data flow model that is not suitable for other popular applications. This paper focuses on one such class of applications: those that reuse a working set of data across multiple parallel operations. This includes many iterative machine learning algorithms, as well as interactive data analysis tools. We propose a new framework called Spark that supports these applications while retaining the scalability and fault tolerance of MapReduce. To achieve these goals, Spark introduces an abstraction called resilient distributed datasets (RDDs). An RDD is a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost. Spark can outperform Hadoop by 10x in iterative machine learning jobs, and can be used to interactively query a 39 GB dataset with sub-second response time.

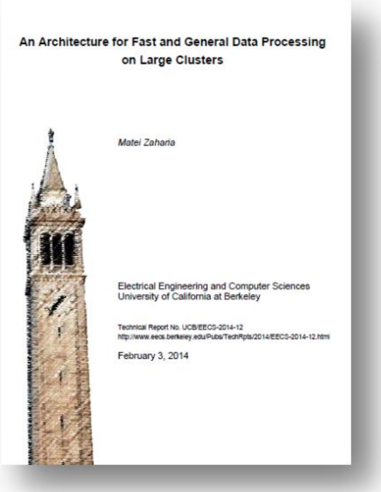
MapReduce/Dryad job, each job must reload the data from disk, incurring a significant performance penalty.

- **Interactive analytics**: Hadoop is often used to run ad-hoc exploratory queries on large datasets, through SQL interfaces such as Pig [21] and Hive [1]. Ideally, a user would be able to load a dataset of interest into memory across a number of machines and query it repeatedly. However, with Hadoop, each query incurs significant latency (tens of seconds) because it runs as a separate MapReduce job and reads data from disk.

This paper presents a new cluster computing framework called Spark, which supports applications with working sets while providing similar scalability and fault tolerance properties to MapReduce.

The main abstraction in Spark is that of a *resilient distributed dataset* (RDD), which represents a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost. Users can explicitly cache an RDD in memory across machines and reuse it

Zaharia M, Chowdhury M, Franklin M J, et al. Spark: cluster computing with working sets[J]. HotCloud, 2010, 10: 10-10.



An Architecture for Fast and General Data Processing on Large Clusters

Matei Zaharia

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-12
http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-12.html

February 3, 2014

Spark的特点

速度快

- 内存计算
- 让计算靠近数据

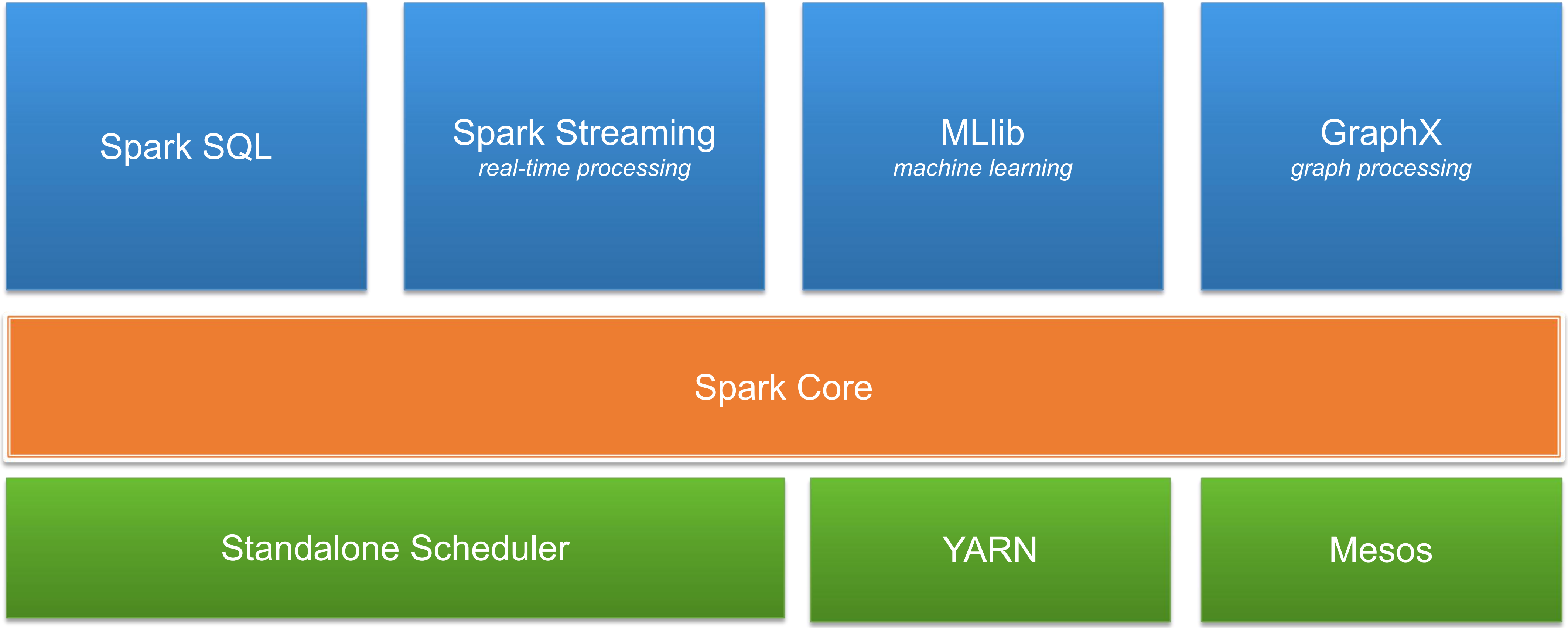
通用性

- 流计算
- 交互计算
- 图计算

易用性

- Scala, Python, Java 的API
- 提供丰富的操作，如filter、sort、join、save、count、groupByKey..
- SQL, 机器学习, 流处理和图形处理的库
- 在Hadoop或者在单机上运行

Spark体系结构



Spark SQL

Spark Streaming
real-time processing

MLlib
machine learning

GraphX
graph processing

Spark Core

Standalone Scheduler

YARN

Mesos

8.如何继续学习本章知识

1. 大数据技术的可扩展性

- 横向扩展与纵向扩展

2. 大数据的实时处理

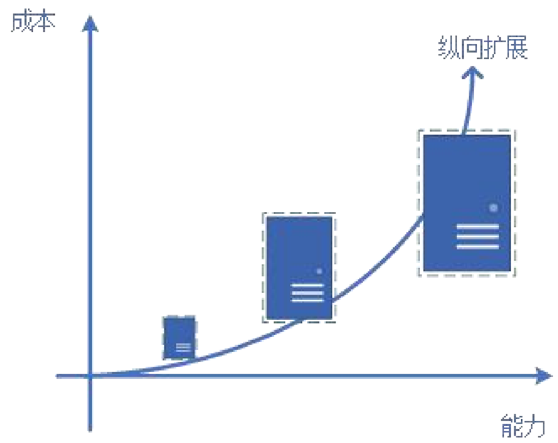
- 批处理与流处理

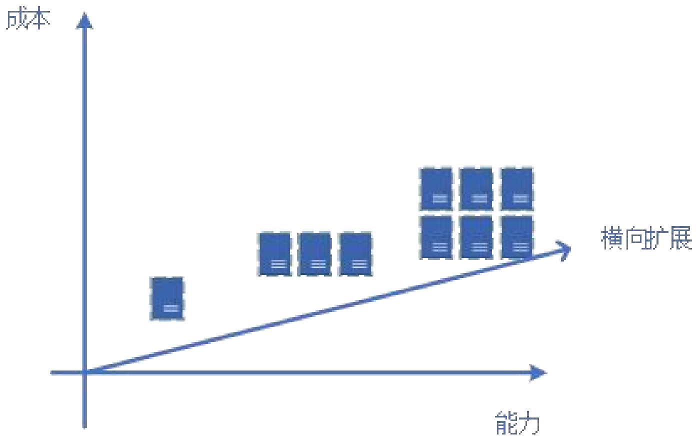
3. 大数据技术的多样性

- 大数据产业全景图（Big Data Landscape）

4. 统一分析

- 基于Databricks 的统一分析平台的架构





小结

1.本章定位与内容简介

2.本章学习提示及要求

3.几个核心概念的区别

4.CAP理论

5.BASE原则

6.Lambda 架构（Lambda Architecture）

7.Spark

8.如何继续学习本章知识