获取教材和讲义 PPT 等各种课程资料请访问 http://dblab.xmu.edu.cn/node/422

=课程教材由林子雨老师根据网络资料编著=

厦门大学计算机科学系教师 林子雨 编著

http://www.cs.xmu.edu.cn/linziyu

2013 年 9 月

前言

本教程由厦门大学计算机科学系教师林子雨编著，可以作为计算机专业研究生课程《大数据技术基础》的辅助教材。

本教程的主要内容包括：大数据概述、大数据处理模型、大数据关键技术、大数据时代面临的新挑战、NoSQL 数据库、云数据库、Google Spanner、Hadoop、HDFS、HBase、MapReduce、Zookeeper、流计算、图计算和 Google Dremel 等。

本教程是林子雨通过大量阅读、收集、整理各种资料后精心制作的学习材料，与广大数据库爱好者共享。教程中的内容大部分来自网络资料和书籍，一部分是自己撰写。对于自写内容，林子雨老师拥有著作权。

本教程 PDF 文档及其全套教学 PPT 可以通过网络免费下载和使用（下载地址：http://dblab.xmu.edu.cn/node/422）。教程中可能存在一些问题，欢迎读者提出宝贵意见和建议！

本教程已经应用于厦门大学计算机科学系研究生课程《大数据技术基础》，欢迎访问 2013 班级网站 http://dblab.xmu.edu.cn/node/423。

林子雨的 E-mail 是：ziyulin@xmu.edu.cn。

林子雨的个人主页是：http://www.cs.xmu.edu.cn/linziyu。

林子雨于厦门大学海韵园

2013 年 9 月

第 2 章 大数据关键技术与挑战

厦门大学计算机科学系教师 林子雨 编著

个人主页：http://www.cs.xmu.edu.cn/linziyu

课程网址：http://dblab.xmu.edu.cn/node/422

2013 年 9 月

第 2 章 大数据关键技术与挑战

大数据价值的完整体现需要多种技术的协同。文件系统提供最底层存储能力的支持。为了便于数据管理，需要在文件系统之上建立数据库系统。通过索引等的构建，对外提供高效的数据查询等常用功能。最终通过数据分析技术从数据库中的大数据提取出有益的知识。

本章内容首先介绍大数据处理基本流程，然后介绍大数据处理模型，接下来阐述了大数据关键技术，并讨论了大数据时代面临的新挑战，内容要点如下：

- 大数据处理的基本流程
- 大数据处理模型
- 大数据关键技术
- 大数据处理工具
- 大数据时代面临的新挑战

2.1 大数据处理的基本流程

大数据的数据来源广泛，应用需求和数据类型都不尽相同，但是最基本的处理流程一致。海量 Web 数据的处理是一类非常典型的大数据应用，从中可以归纳出大数据处理的最基本流程，如图 2-1 所示。

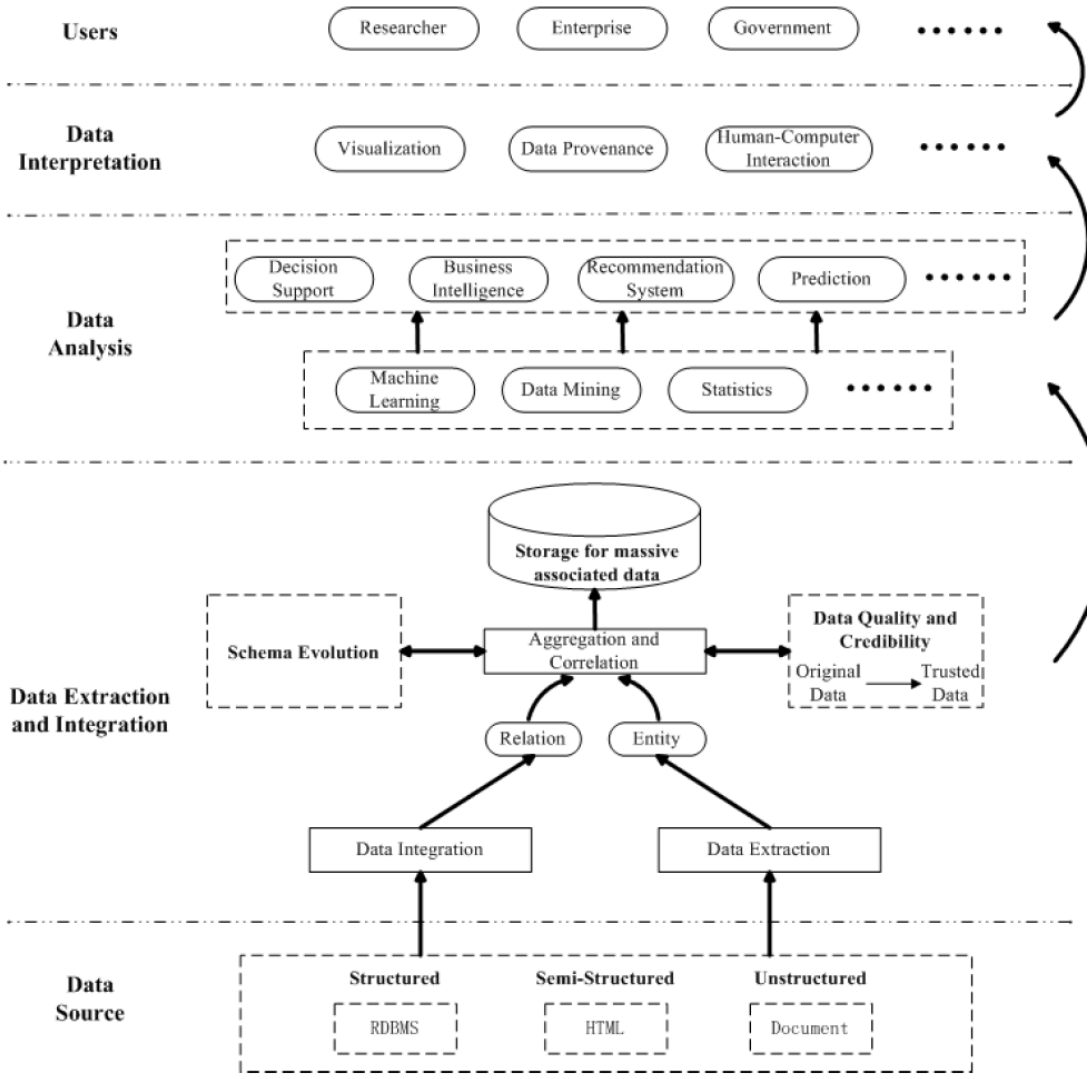

图 2-1 大数据处理的基本流程

整个大数据的处理流程可以定义为：在合适工具的辅助下，对广泛异构的数据源进行抽取和集成，结果按照一定的标准进行统一存储，并利用合适的数据分析技术对存储的数据进行分析，从中提取有益的知识并利用恰当的方式将结果展现给终端用户。具体来说，可以分为数据抽取与集成、数据分析以及数据解释。

2.1.1数据抽取与集成

大数据的一个重要特点就是多样性，这就意味着数据来源极其广泛，数据类型极为繁杂。这种复杂的数据环境给大数据的处理带来极大的挑战。要想处理大数据，首先必须对所需数据源的数据进行抽取和集成，从中提取出关系和实体，经过关联和聚合之后采用统一定义的结构来存储这些数据。在数据集成和提取时需要对数据进行清洗，保证数据质量及可信

性。

同时还要特别注意大数据时代模式和数据的关系，大数据时代的数据往往是先有数据再有模式，且模式是在不断的动态演化之中的。数据抽取和集成技术不是一项全新的技术，传统数据库领域已对此问题有了比较成熟的研究。随着新的数据源的涌现，数据集成方法也在不断的发展之中。从数据集成模型来看，现有的数据抽取与集成方式可以大致分为以下四种类型：基于物化或是 ETL 方法的引擎(Materialization or ETL engine)、基于联邦数据库或中间件方法的引擎(Federation engine or Mediator)、基于数据流方法的引擎(Stream engine)及基于搜索引擎的方法(Search engine)。

[拓展阅读]数据集成方式

数据集成方式包括数据整合、数据联邦、数据传播和混合方法等四种：

（1）数据整合（Data Consolidation）：不同数据源的数据被物理地集成到数据目标。利用 ETL 工具把数据源中的数据批量地加载到数据仓库，就属于数据整合的方式。

（2）数据联邦（Data Federation）：在多个数据源的基础上建立一个统一的逻辑视图，对外界应用屏蔽数据在各个数据源的分布细节。对于这些应用而言，只有一个统一的数据访问入口，但是实际上，被请求的数据只是逻辑意义上的集中，在物理上仍然分布在各个数据源中，只有被请求时，才临时从不同数据源获取相关数据，进行集成后提交给数据请求者。当数据整合方式代价太大或者为了满足一些突发的实时数据需求时，可以考虑采用数据联邦的方式建立企业范围内的全局统一数据视图。

（3）数据传播（Data Propagation）：数据在多个应用之间的传播。比如，在企业应用集成（EAI）解决方案中，不同应用之间可以通过传播消息进行交互。

（4）混合方式（A Hybrid Approach）：在这种方式中，对于那些不同应用都使用的数据采用数据整合的方式进行集成，而对那些只有特定应用才使用的数据则采用数据联邦的方式进行集成。

2.1.2数据分析

数据分析是整个大数据处理流程的核心，因为大数据的价值产生于分析过程。从异构数据源抽取和集成的数据构成了数据分析的原始数据。根据不同应用的需求可以从这些数据中选择全部或部分进行分析。传统的分析技术如数据挖掘、机器学习、统计分析等在大数据时

代需要做出调整，因为这些技术在大数据时代面临着一些新的挑战，主要有：

- **数据量大并不一定意味着数据价值的增加，相反这往往意味着数据噪音的增多**。因此在数据分析之前必须进行数据清洗等预处理工作，但是预处理如此大量的数据对于机器硬件以及算法都是严峻的考验。
- **大数据时代的算法需要进行调整**。首先，大数据的应用常常具有实时性的特点，算法的准确率不再是大数据应用的最主要指标。很多场景中算法需要在处理的实时性和准确率之间取得一个平衡，比如在线的机器学习算法(online machine learning)。其次，云计算是进行大数据处理的有力工具，这就要求很多算法必须做出调整以适应云计算的框架，算法需要变得具有可扩展性。最后，在选择算法处理大数据时必须谨慎，当数据量增长到一定规模以后，可以从小量数据中挖掘出有效信息的算法并不一定适用于大数据。统计学中的邦弗朗尼原理(Bonferroni's Principle)就是一个典型的例子。
- **数据结果好坏的衡量**。得到分析结果并不难，但是结果好坏的衡量却是大数据时代数据分析的新挑战。大数据时代的数据量大、类型庞杂，进行分析的时候往往对整个数据的分布特点掌握得不太清楚，这会导致最后在设计衡量的方法以及指标的时候遇到诸多困难。大数据分析已被广泛应用于诸多领域，典型的有推荐系统、商业智能、决策支持等。

[拓展阅读] 邦弗朗尼原理

假定人们有一定量的数据，并期望从该数据中找到某个特定类型的事件。即使数据完全随机，也可以期望该类型事件会发生。随着数据规模的增长，这类事件出现的数目也随之上升。任何随机数据往往都会有一些不同寻常的特征，这些特征看上去虽然很重要，但是实际上并不重要，除此之外，别无他由，从这个意义上说，这些事件的出现纯属"臆造"。统计学上有一个称为邦弗朗尼校正（Bonferroni correction）的定理，该定理给出一个在统计上可行的方法来避免在搜索数据时出现的大部分"臆造"正响应。这里并不打算介绍定理的统计细节，只给出一个非正式的称为邦弗朗尼原理的版本，该原理可以帮助我们避免将随机出现看成真正出现。在数据随机性假设的基础上，可以计算所寻找事件出现次数的期望值。如果该结果显著高于你所希望找到的真正实例的数目，那么可以预期，寻找到的几乎任何事物都是臆造的，也就是说，它们是在统计上出现的假象，而不是你所寻找事件的凭证。上述观察现象是邦弗朗尼原理的非正式

阐述。简单地说，你假设：特定事件的发生预示着特定内容。如果特定事件(例如：在酒店中聚会)发生的概率乘以样本空间得到的数目远远大与你期望的特定内容(例如：歹徒)的数目，那么你的假设是错的。总之，我们不能指望通过大规模统计来发现一些很“稀有”的事情或者规律。例如：恐怖袭击这样的事情，多少年都遇不到一次。通过对某些数据的大规模统计，可能推断出每年要发生很多起恐怖袭击，这本身就不现实。

2.1.3数据解释

数据分析是大数据处理的核心，但是用户往往更关心结果的展示。如果分析的结果正确但是没有采用适当的解释方法，则所得到的结果很可能让用户难以理解，极端情况下甚至会误导用户。数据解释的方法很多，比较传统的就是以文本形式输出结果或者直接在电脑终端上显示结果。这种方法在面对小数据量时是一种很好的选择。但是大数据时代的数据分析结果往往也是海量的，同时结果之间的关联关系极其复杂，采用传统的解释方法基本不可行。可以考虑从下面两个方面提升数据解释能力：

- **引入可视化技术**。可视化作为解释大量数据最有效的手段之一率先被科学与工程计算领域采用。通过对分析结果的可视化用形象的方式向用户展示结果，而且图形化的方式比文字更易理解和接受。常见的可视化技术有标签云(Tag Cloud)、历史流(history flow)、空间信息流(Spatial information flow)等。可以根据具体的应用需要选择合适的可视化技术。
- **让用户能够在一定程度上了解和参与具体的分析过程**。这个既可以采用人机交互技术，利用交互式的数据分析过程来引导用户逐步地进行分析，使得用户在得到结果的同时更好的理解分析结果的由来。也可以采用数据起源技术，通过该技术可以帮助追溯整个数据分析的过程，有助于用户理解结果。

2.2 大数据处理模型

2.2.1大数据之快和处理模型

天下武功，唯快不破。这句话源自《拳经》，经过金山公司董事长雷军等人的演绎，几乎成了互联网时代商业致胜的不二法则。那么，大数据的快又从何说起呢？

“快”，来自几个朴素的思想：

- **时间就是金钱**。时间在分母上，越小，单位价值就越大。面临同样大的数据矿山，“挖矿”效率是竞争优势。Zara 与 H&M 有相似的大数据供应，Zara 胜出的原因毫无疑问就是“快”。
- **像其它商品一样，数据的价值会折旧**。过去一天的数据，比过去一个月的数据可能都更有价值。更普遍意义上，它就是时间成本的问题：等量数据在不同时间点上价值不等。NewSQL 的先行者 VoltDB 发明了一个概念叫做 Data Continuum：数据存在于一个连续时间轴（time continuum）上，每一个数据项都有它的年龄，不同年龄的数据有不同的价值取向，“年轻”（最近）时关注个体的价值，“年长”（久远）时注重集合价值。
- **数据跟新闻和金融行情一样，具有时效性**。炒股软件免费版给你的数据有十几秒的延迟，这十几秒是快速猎食者宰割散户的机会；而华尔街大量的机构使用高频机器交易（70%的成交量来自高频交易），能发现微秒级交易机会的吃定毫秒级的。物联网这块，很多传感器的数据，产生几秒之后就失去意义了。美国国家海洋和大气管理局的超级计算机能够在日本地震后 9 分钟计算出海啸的可能性，但 9 分钟的延迟对于瞬间被海浪吞噬的生命来说还是太长了。

大家知道，购物篮分析是沃尔玛横行天下的绝技，其中最经典的就是关联产品分析：从大家耳熟能详的“啤酒加尿布”，到飓风来临时的“馅饼（pop-tarts）加手电筒”和“馅饼加啤酒”。可是，此“购物篮”并非顾客拎着找货的那个，而是指你买完帐单上的物品集合。对于快消品等有定期消费规律的产品来说，这种“购物篮”分析尚且有效，但对绝大多数商品来说，找到顾客“触点（touch points）”的最佳时机并非在结帐以后，而是在顾客还领着篮子扫街逛店的正当时。电子商务具备了这个能力，从点击流（clickstream）、浏览历史和行为（如放入购物车）中实时发现顾客的即时购买意图和兴趣。这就是“快”的价值。

设想我们站在某个时间点上，背后是静静躺着的老数据，面前是排山倒海扑面而来的新数据。在令人窒息的数据海啸面前，我们的数据存储系统如同一个小型水库，而数据处理系统则可以看作是水处理系统。数据涌入这个水库， 如果不能很快处理，只能原封不动地排出。对于数据拥有者来说，除了付出了存储设备的成本，没有收获任何价值。

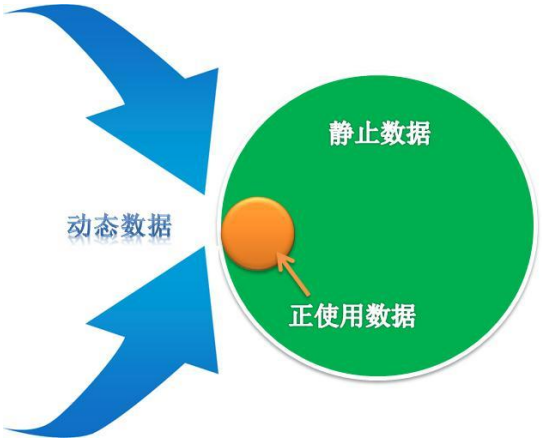

图 2-2 数据的三种状态

如图 2-2 所示，按照数据的三状态定义，水库里一平如镜（非活跃）的水是“静止数据（data at rest）”，水处理系统中上下翻动的水是“正使用数据（data inuse）”，汹涌而来的新水流就是“动态数据（data in motion）”。

“快”说的是两个层面：

- **一个是“动态数据”来得快**。动态数据有不同的产生模式。有的是“爆发(burst)”模式，极端的例子如欧洲核子研究中心（CERN）的大型强子对撞机(Large Hadron Collider，简称 LHC)，此机不撞则已，一撞惊人，工作状态下每秒产生 PB 级的数据。也有的动态数据是涓涓细流的模式，典型的如 clickstream、日志、RFID 数据、GPS 位置信息和 Twitter 的 firehose 流数据等。
- **二是对“正使用数据”处理得快**。水处理系统可以从水库调出水来进行处理（“静止数据”转变为“正使用数据”），也可以直接对涌进来的新水流处理（“动态数据”转变为“正使用数据”）。这对应着两种大相迥异的处理范式：批处理和流处理。

如图 2-3 所示，左半部是批处理：以“静止数据”为出发点，数据是任尔东西南北风、我自岿然不动，处理逻辑进来，算完后价值出去。Hadoop 就是典型的批处理范式：HDFS 存放已经沉淀下来的数据，MapReduce 的作业调度系统把处理逻辑送到每个节点进行计算。这非常合理，因为搬动数据比发送代码更昂贵。

右半部则是流数据处理范式。这次不动的是逻辑，“动态数据”进来，计算完后价值留下，原始数据加入“静止数据”，或索性丢弃。流处理品类繁多，包括传统的消息队列（绝大多数的名字以 MQ 结尾）、事件流处理（Event Stream Processing）/复杂事件处理（Complex Event Processing 或 CEP）（如 Tibco 的 BusinessEvents 和 IBM 的 InfoStreams）、分布式发布/订阅系

统（如 Kafka）、专注于日志处理的（如 Scribe 和 Flume）、通用流处理系统（如 Storm 和 S4）等。

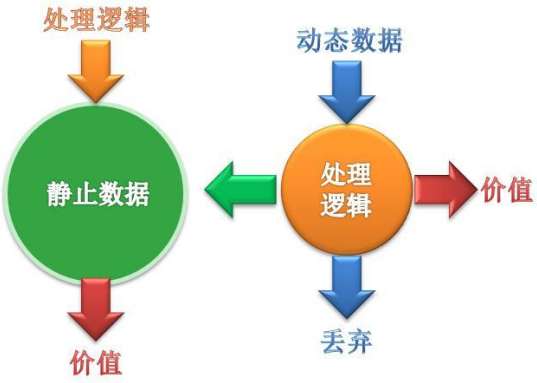

图 2-3 数据的两种处理模型

这两种范式与我们日常生活中的两种信息处理习惯相似：有些人习惯先把信息存下来（如书签、To Do 列表、邮箱里的未读邮件），稍后一次性地处理掉（也有可能越积越多，旧的信息可能永远不会处理了）；有些人喜欢任务来一件做一件，信息来一点处理一点，有的直接过滤掉，有的存起来。

没有定规说哪种范式更好，对于批量数据，多数是先进入存储系统，然后再来处理，因此以批处理范式为主；而对于流数据，多采用流范式。传统上认为流处理的方式更快，但流范式能处理的数据常常局限于最近的一个数据窗口，只能获得实时智能（real-time intelligence），不能实现全时智能（all-time intelligence）。批处理擅长全时智能，但翻江倒海捣腾数据肯定慢，所以亟需把批处理加速。

两种范式常常组合使用，而且形成了一些定式：

- **流处理作为批处理的前端**：比如大型强子对撞机，每秒 PB 级的数据先经过流处理范式进行过滤，只有那些科学家感兴趣的撞击数据保留下来进入存储系统，留待批处理范式处理。这样，欧洲核子研究中心每年的新增存储量可以减到 25PB。
- **流处理与批处理肩并肩**：流处理负责动态数据和实时智能，批处理负责静止数据和历史智能，实时智能和历史智能合并成为全时智能。

那么，如何实现“快”的数据处理呢？

首先，“快”是个相对的概念，可以是实时，也可以秒级、分钟级、小时级、天级甚至更长的延迟。实现不同级别的“快”采用的架构和付出的代价也不一样。所以，对于每一个

面临“快”问题的决策者和架构师来说，第一件事情就是要搞清楚究竟要多“快”。“快”无止境，找到足够“快”的那个点，那就够了。

其次，考虑目前的架构是不是有潜力改造到足够“快”。很多企业传统的关系型数据库中数据量到达 TB 级别，就慢如蜗牛了。在转向新的架构（如 NoSQL 数据库）之前，可以先考虑分库分表（sharding）和内存缓存服务器（如 memcached）等方式延长现有架构的生命。

如果预测未来数据的增长必将超出现有架构的上限，那就要规划新的架构了。这里不可避免要做出选择，或者选择流处理结构，或者选择批处理结构，当然也可以选择两者兼具。Intel 有一位老法师说：any big data platform needs to be architected for particular problems（任何一个大数据平台都需要为特定的问题度身定做）。这是非常有道理的。为什么呢？比如说大方向决定了要用流处理架构，落实到具体产品少说有上百种，所以要选择最适合的流处理产品。再看批处理架构，MapReduce 也不能包打天下，碰到多迭代、交互式计算就无能为力了；NoSQL 更是枝繁叶茂，有名有姓的 NoSQL 数据库好几十种。这时候请一个好的大数据咨询师很重要，让他帮助企业选择一个量身定制的大数据解决方案。

总体上讲，还是有一些通用的技术思路来实现“快”：

- **如果数据流入量太大，在前端就地采用流处理进行即时处理、过滤掉非重要数据**。
- **把数据预处理成适于快速分析的格式**。预处理常常比较耗时，但对不常改动的惰性数据，预处理的代价在长期的使用中可以被分摊到很小，甚至可以忽略不计。谷歌的 Dremel 就是把只读的嵌套数据转成类似于列式数据库的形式，实现了 PB 级数据的秒级查询。
- **增量计算--也即先顾眼前的新数据，再去更新老数据**。对传统的批处理老外叫做 reboil the ocean，每次计算都要翻江倒海把所有数据都捣腾一遍，自然快不了。而增量计算把当前重点放在新数据上，先满足“快”；同时抽空把新数据（或新数据里提炼出来的信息）更新到老数据（或历史信息库）中，又能满足“全”。谷歌的 Web 索引自 2010 年起从老的 MapReduce 批量系统升级成新的增量索引系统，能够极大地缩短网页被爬虫爬到和被搜索到之间的延迟。前面说的“流处理和批处理肩并肩”也是一种增量计算。
- **很多批处理系统慢的根源是磁盘和 I/O，把原始数据和中间数据放在内存里，一定能极大地提升速度**。这就是内存计算（In-memory computing）。内存计算最简单的形式是内存缓存，Facebook 超过 80%的活跃数据就在 memcached 里。比较复杂的有内存数据库和数据分析平台，如 SAP 的 HANA，NewSQL 的代表 VoltDB 和伯克利

的开源内存计算框架 Spark（Intel 也开始参与）。斯坦福的 John Ousterhout（Tcl/Tk 以及集群文件系统 Lustre 的发明者）搞了个更超前的 RAMCloud，号称所有数据只生活在内存里。未来新的非易失性内存（断电数据不会丢失）会是个游戏规则改变者。Facebook 在 3 月宣布了闪存版的 Memcached，叫 McDipper，比起单节点容量可以提升 20 倍，而吞吐量仍能达到每秒数万次操作。另一种非易失性内存，相变内存（Phase Change Memory），在几年内会商用，它的每比特成本可以是 DRAM 的 1/10，性能比 DRAM 仅慢 2-10 倍，比现今的闪存（NAND）快 500 倍，寿命长 100 倍。除内存计算外，还有其它的硬件手段来加速计算、存储和数据通讯，如 FPGA（IBM 的 Netezza 和 Convey 的 Hybrid-Core），SSD 和闪存卡（SAP HANA 和 Fusion IO），压缩 PCIe 卡，更快和可配置的互联（Infiniband 的 RDMA 和 SeaMicro SM15000 的 Freedom Fabrics）等。此处不再细表。

- **降低对精确性的要求**。大体量、精确性和快不可兼得，顶多取其二。如果要在大体量数据上实现“快”，必然要适度地降低精确性。对数据进行采样后再计算就是一种办法，伯克利 BlinkDB 通过独特的采用优化技术实现了比 Hive 快百倍的速度，同时能把误差控制在 2-10%。

2.2.2流处理

大数据的应用类型很多，主要的处理模式可以分为流处理(Stream Processing)和批处理(Batch Processing)两种。批处理是先存储后处理(Store-then-process)，而流处理则是直接处理(Straight-through processing)。

流处理的基本理念是数据的价值会随着时间的流逝而不断减少。因此尽可能快的对最新的数据做出分析并给出结果是所有流数据处理模式的共同目标。需要采用流数据处理的大数据应用场景主要有网页点击数的实时统计、传感器网络、金融中的高频交易等。

流处理的处理模式将数据视为流，源源不断的数据组成了数据流。当新的数据到来时就立刻处理并返回所需的结果。图 2-4 是流处理中基本的数据流模型：

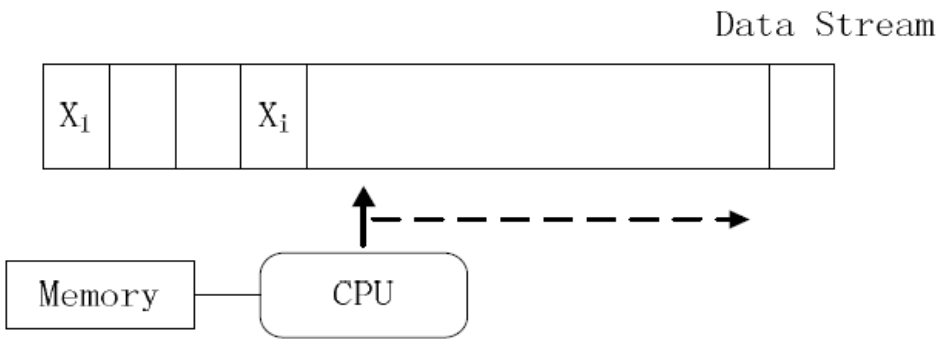

图 2-4 流处理中基本的数据流模型

数据的实时处理是一个很有挑战性的工作，数据流本身具有持续达到、速度快且规模巨大等特点，因此通常不会对所有的数据进行永久化存储，而且数据环境处在不断的变化之中，系统很难准确掌握整个数据的全貌。由于响应时间的要求，流处理的过程基本在内存中完成，其处理方式更多的依赖于在内存中设计巧妙的概要数据结构(Synopsis data structure)，内存容量是限制流处理模型的一个主要瓶颈。以 PCM(相变存储器)为代表的 SCM(Storage Class Memory，储存级内存)设备的出现或许可以使内存未来不再成为流处理模型的制约。

数据流的理论及技术研究已经有十几年的历史，目前仍旧是研究热点。于此同时很多实际系统也已开发和得到广泛的应用，比较代表性的开源系统如 Twitter 的 Storm、Yahoo 的 S4 以及 Linkedin 的 Kafka 等。

2.2.3批处理

Google 公司在 2004 年提出的 MapReduce 编程模型是最具代表性的批处理模式。一个完整的 MapReduce 过程如图 2-5 所示。

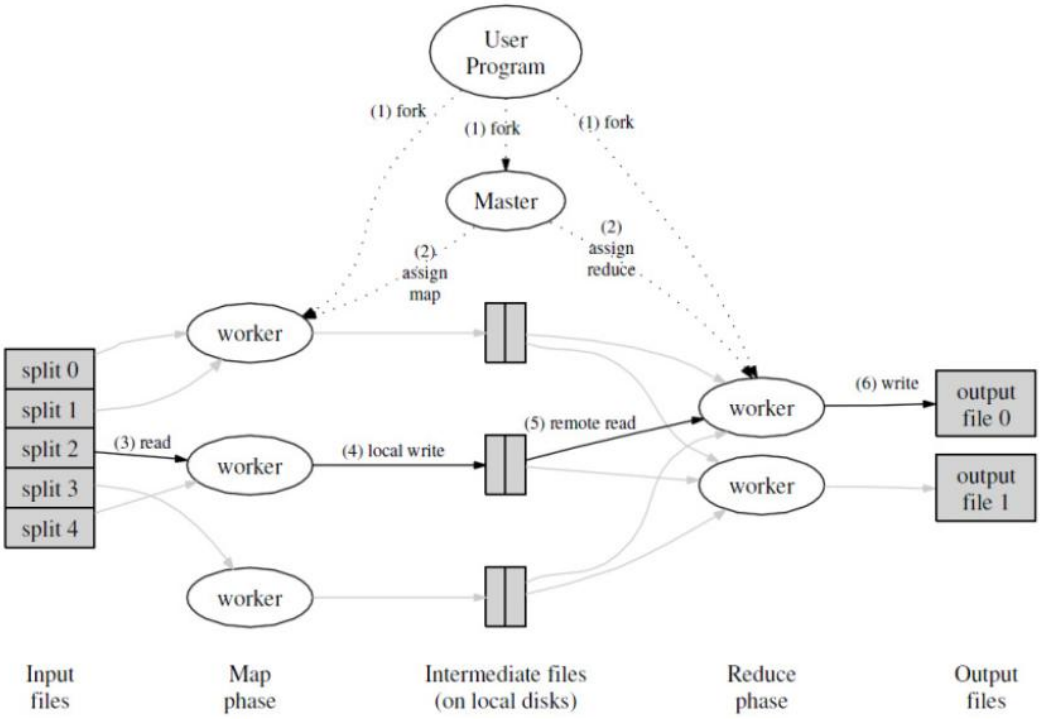

图 2-5 MapReduce 过程

MapReduce 模型首先将用户的原始数据源进行分块，然后分别交给不同的 Map 任务区处理。Map 任务从输入中解析出 Key/Value 对集合，然后对这些集合执行用户自行定义的 Map 函数得到中间结果，并将该结果写入本地硬盘。Reduce 任务从硬盘上读取数据之后，会根据 key 值进行排序，将具有相同 key 值的组织在一起。最后用户自定义的 Reduce 函数会作用于这些排好序的结果并输出最终结果。

从 MapReduce 的处理过程我们可以看出，MapReduce 的核心设计思想在于：1)将问题分而治之；2)把计算推到数据而不是把数据推到计算，有效地避免数据传输过程中产生的大量通讯开销。MapReduce 模型简单，且现实中很多问题都可用 MapReduce 模型来表示。因此该模型公开后，立刻受到极大的关注，并在生物信息学、文本挖掘等领域得到广泛的应用。无论是流处理还是批处理，都是大数据处理的可行思路。大数据的应用类型很多，在实际的大数据处理中，常常并不是简单的只使用其中的某一种，而是将二者结合起来。互联网是大数据最重要的来源之一，很多互联网公司根据处理时间的要求将自己的业务划分为在线(Online)、近线(Nearline)和离线(Offline)，比如著名的职业社交网站 Linkedin。这种划分方式是按处理所耗时间来划分的。其中在线的处理时间一般在秒级，甚至是毫秒级，因此通常采用上面所说的流处理。离线的处理时间可以以天为基本单位，基本采用批处理方式，这种方式可以最大限度地利用系统 I/O。近线的处理时间一般在分钟级或者是小时级，对其处理模

型并没有特别的要求，可以根据需求灵活选择。但在实际中多采用批处理模式。

2.3 大数据关键技术

如果将各种大数据的应用比作一辆辆“汽车”的话，支撑起这些“汽车”运行的“高速公路”就是云计算。正是云计算技术在数据存储、管理与分析等方面的支撑，才使得大数据有用武之地。在所有的“高速公路”中，Google 无疑是技术最为先进的一个。需求推动创新，面对海量的 Web 数据， Google 于 2006 年首先提出了云计算的概念。支撑 Google 内部各种大数据应用的正是其自行研发的一系列云计算技术和工具。难能可贵的是 Google 并未将这些技术完全封闭，而是以论文的形式逐步公开其实现。正是这些公开的论文，使得以 GFS、MapReduce、Bigtable 为代表的一系列大数据处理技术被广泛了解并得到应用，同时还催生出以 Hadoop 为代表的一系列云计算开源工具。云计算技术很多，但是通过 Google 云计算技术的介绍能够快速、完整地把握云计算技术的核心和精髓。这里以 Google 的相关技术介绍为主线，详细介绍 Google 以及其他众多学者和研究机构在大数据技术方面已有的一些工作。根据 Google 已公开的论文及相关资料，结合大数据处理的需求，可以看出 Google 的技术演化过程，如图 2-6 所示。

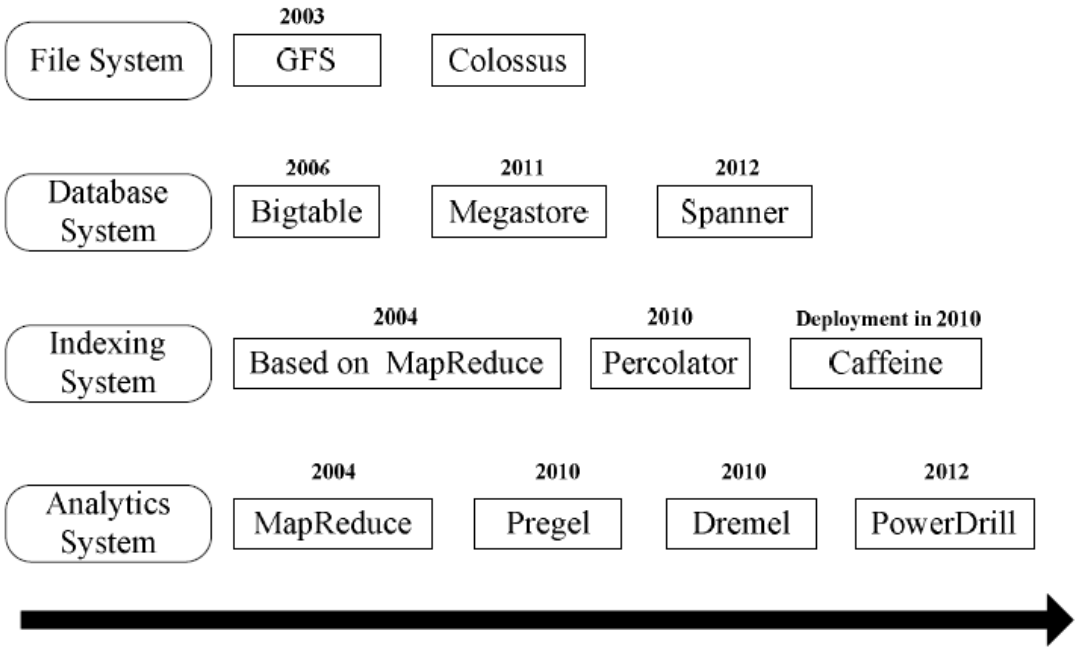

图 2-6 Google 的技术演化过程

2.3.1文件系统

文件系统是支撑上层应用的基础。在 Google 之前，尚未有哪个公司面对过如此多的海量数据。因此，对于 Google 而言并没有完全成熟的存储方案可以直接使用。Google 认为系统组件失败是一种常态而不是异常，基于此思想，Google 自行设计开发了 Google 文件系统 GFS(Google File System)。GFS 是构建在大量廉价服务器之上的一个可扩展的分布式文件系统，GFS 主要针对文件较大，且读远大于写的应用场景，采用主从(Master-Slave)结构。

GFS 通过数据分块、追加更新(Append-Only)等方式实现了海量数据的高效存储。随着时间推移，GFS 的架构逐渐开始无法适应需求。Google 对 GFS 进行了重新的设计，该系统正式的名称为 Colossus，其中，GFS 的单点故障(指仅有一个主节点容易成为系统的瓶颈)、海量小文件的存储等问题在 Colossus 中均得到了解决。除了 Google，众多企业和学者也从不同方面对满足大数据存储需求的文件系统进行了详尽的研究。微软自行开发的 Cosmos 支撑着其搜索、广告等业务。HDFS 和 CloudStore 都是模仿 GFS 的开源实现。GFS 类的文件系统主要是针对较大文件设计的，而在图片存储等应用场景，文件系统主要存储海量小文件，此时 GFS 等文件系统因为频繁读取元数据等原因，效率很低。针对这种情况，Facebook 推出了专门针对海量小文件的文件系统 Haystack，通过多个逻辑文件共享同一个物理文件、增加缓存层、部分元数据加载到内存等方式有效的解决了 Facebook 海量图片存储问题。淘宝推出了类似的文件系统 TFS(Tao File System)，通过将小文件合并成大文件、文件名隐含部分元数据等方式实现了海量小文件的高效存储。FastDFS 针对小文件的优化类似于 TFS。

2.3.2数据库系统

原始的数据存储在文件系统之中，但是，用户习惯通过数据库系统来存取文件。因为这样会屏蔽掉底层的细节，且方便数据管理。直接采用关系模型的分布式数据库并不能适应大数据时代的数据存储，主要因为：

1)**规模效应所带来的压力**。大数据时代的数据量远远超过单机所能容纳的数据量，因此必须采用分布式存储的方式。这就需要系统具有很好的扩展性，但这恰恰是传统数据库的弱势之一。因为传统的数据库产品对于性能的扩展更倾向于 Scale-Up(纵向扩展)的方式，而这种方式对于性能的增加速度远低于需要处理数据的增长速度，且性能提升存在上限。适应大数据的数据库系统应当具有良好的 Scale-Out(横向扩展)能力，而这种性能扩展方式恰恰是传

统数据库所不具备的。即便是性能最好的并行数据库产品其 Scale-Out 能力也相对有限。

2)**数据类型的多样化**。传统的数据库比较适合结构化数据的存储，但是数据的多样性是大数据时代的显著特征之一。这也就是意味着除了结构化数据，半结构化和非结构化数据也将是大数据时代的重要数据类型组成部分。如何高效地处理多种数据类型是大数据时代数据库技术面临的重要挑战之一。

3)**设计理念的冲突**。关系数据库追求的是“One size fits all”的目标，希望将用户从繁杂的数据管理中解脱出来，在面对不同的问题时不需要重新考虑数据管理问题，从而可以将重心转向其他的部分。但在大数据时代不同的应用领域在数据类型、数据处理方式以及数据处理时间的要求上有极大的差异。在实际的处理中几乎不可能有一种统一的数据存储方式能够应对所有场景。比如对于海量 Web 数据的处理就不可能和天文图像数据采取同样的处理方式。在这种情况下，很多公司开始尝试从“One size fits one”和“One size fits domain”的设计理念出发来研究新的数据管理方式，并产生了一系列非常有代表性的工作。

4)**数据库事务特性**。众所周知，关系数据库中事务的正确执行必须满足 ACID 特性，即原子性(Atomicity)、一致性(Consistency)、隔离性(Isolation)和持久性(Durability)。对于数据强一致性的严格要求使其在很多大数据场景中无法应用。这种情况下出现了新的 BASE 特性，即只要求满足 Basically Available(基本可用), Soft state(柔性状态)和 Eventually consistent(最终一致)。从分布式领域著名的 CAP 理论的角度来看，ACID 追求一致性 C，而 BASE 更加关注可用性 A。正是在事务处理过程中对于 ACID 特性的严格要求，使得关系型数据库的可扩展性极其有限。

面对这些挑战，以 Google 为代表的一批技术公司纷纷推出了自己的解决方案。Bigtable 是 Google 早期开发的数据库系统，它是一个多维稀疏排序表，由行和列组成，每个存储单元都有一个时间戳，形成三维结构。不同的时间对同一个数据单元的多个操作形成的数据的多个版本之间由时间戳来区分。除了 Bigtable，Amazon 的 Dynamo 和 Yahoo 的 PNUTS 也都是非常具有代表性的系统。Dynamo 综合使用了键/值存储、改进的分布式哈希表(DHT)、向量时钟(Vector Clock)等技术实现了一个完全的分布式、去中心化的高可用系统。PNUTS 是一个分布式的数据库，在设计上使用弱一致性来达到高可用性的目标，主要的服务对象是相对较小的记录，比如在线的大量单个记录或者小范围记录集合的读和写访问。不适合存储大文件、流媒体等。Bigtable、Dynamo、PNUTS 等的成功促使人们开始对关系数据库进行反思，由此产生了一批未采用关系模型的数据库，这些方案现在被统一的称为 NoSQL(Not Only SQL)。NoSQL 并没有一个准确的定义，但一般认为 NoSQL 数据库应当具有以下的特征：模式自由

(schema-free)、支持简易备份(easy replication support)、简单的应用程序接口(simple API)、最终一致性(或者说支持 BASE 特性，不支持 ACID)、支持海量数据(Huge amount of data)。

Bigtable 的模型简单，但是，相较传统的关系数据库其支持的功能非常有限，不支持 ACID 特性。因此，Google 开发了 Megastore 系统，虽然其底层数据存储依赖 Bigtable，但是它实现了类似 RDBMS 的数据模型，同时提供数据的强一致性解决方案。Megastore 将数据进行细粒度的分区，数据更新会在机房间进行同步复制。Spanner 是已知的 Google 的最新的数据库系统，Google 在 OSDI2012 上公开了 Spanner 的实现。Spanner 是第一个可以实现全球规模扩展(Global Scale)并且支持外部一致的事务(support externally-consistent distributed transactions)的数据库。通过 GPS 和原子时钟(atomic clocks)技术，Spanner 实现了一个时间 API。借助该 API，数据中心之间的时间同步能够精确到 10ms 以内。Spanner 类似于 Bigtable，但是它具有层次性的目录结构以及细粒度的数据复制。对于数据中心之间不同操作会分别支持强一致性或弱一致性，且支持更多的自动操作。Spanner 的目标是控制一百万到一千万台服务器，最多包含大约 10 万亿目录和一千万亿字节的存储空间。另外，在 SIGMOD2012 上，Google 公开了用于其广告系统的新数据库产品 F1，作为一种混合型数据库，F1 融合兼有 Bigtable 的高扩展性以及 SQL 数据库的可用性和功能性。该产品的底层存储正是采用 Spanner，具有很多新的特性，包括全局分布式、同步跨数据中心复制、可视分片和数据移动、常规事务等。有些比较激进的观点认为“关系数据库已死”，但是，一般而言，关系数据库和 NoSQL 并不是矛盾的对立体，而是可以相互补充的、适用于不同应用场景的技术。例如，实际的互联网系统往往都是 ACID 和 BASE 两种系统的结合。近些年来，以 Spanner 为代表的若干新型数据库的出现，给数据存储带来了 SQL、NoSQL 之外的新思路。这种融合了一致性和可用性的 NewSQL 或许会是未来大数据存储新的发展方向。

2.3.3索引和查询技术

数据查询是数据库最重要的应用之一。而索引则是解决数据查询问题的有效方案。就 Google 自身而言，索引的构建是提供搜索服务的关键部分。Google 最早的索引系统是利用 MapReduce 来更新的。根据更新频率进行层次划分，不同的层次对应不同的更新频率。每次需要批量更新索引，即使有些数据并未改变也需要处理掉。这种索引更新方式效率较低。

随后 Google 提出了 Percolator，这是一种增量式的索引更新器，每次更新不需要替换所有的索引数据，效率大大提高。虽然不是所有的大数据应用都需要索引，但是这种增量计算

的思想非常值得我们借鉴。Google 当前正在使用的索引系统为 Caffeine，构建在 Spanner 之上，采用 Percolator 更新索引。效率相较上一代索引系统而言有大幅度提高。

关系数据库也是利用对数据构建索引的方式较好地解决了数据查询的问题。不同的索引方案使得关系数据库可以满足不同场景的要求。索引的建立以及更新都会耗费较多的时间，在面对传统数据库的小数据量时这些时间和其所带来的查询便利性相比是可以接受的，但是这些复杂的索引方案基本无法直接应用到大数据之上。表 2-1 是对一些索引方案直接应用在 Facebook 上的性能估计。

表 2-1 一些索引方案直接应用在 Facebook 上的性能估计

Algorithms	Index Size for Facebook	Index Time for Facebook	Query Time on Facebook(s)
Ullmann[Ullmann 76]	-	-	>1000
VF2[CordellaFSV04]	-	-	>1000
RDF-3X[NeumannW10]	1T	>20 days	>48
BitMat[AtreCZH10]	2.4T	>20days	>269
Subdue[HolderCD94]	-	>67 years	-
SpiderMine[ZhuQLYHY11]	-	>3 years	-
R-Join[ChengYDYW08]	>175T	$>10^{15}$ years	>200
Distance-Join[ZouCO09]	>175T	$>10^{15}$ years	>4000
GraphQL[HeS08]	>13T(r=2)	>600 years	>2000
Zhao[ZhaoH10]	>12T(r=2)	>600 years	>600
GADDI[ZhangLY09]	$>2*10^5$T(L=4)	$>4*10^5$ years	>400

从上表可以看出不太可能将已有的成熟索引方案直接应用于大数据。NoSQL 数据库针对主键的查询效率一般较高，因此有关的研究集中在 NoSQL 数据库的多值查询优化上。针对 NoSQL 数据库上的查询优化研究主要有两种思路：

1）采用 MapReduce 并行技术优化多值查询：当利用 MapReduce 并行查询 NoSQL 数据库时，每个 MapTask 处理一部分的查询操作，通过实现多个部分之间的并行查询来提高多值查询的效率。此时每个部分的内部仍旧需要进行数据的全扫描。

2）采用索引技术优化多值查询：很多的研究工作尝试从添加多维索引的角度来加速 NoSQL 数据库的查询速度。

就已有方案来看，针对 NoSQL 数据库上的查询优化技术都并不成熟，仍有很多关键性问题亟待解决。

2.3.4数据分析技术

数据分析是 Google 最核心的业务，每一次简单的网络点击背后都需要进行复杂的分析

过程，因此 Google 对其分析系统进行不断地升级改造之中。MapReduce 是 Google 最早采用的计算模型，适用于批处理。图是真实社会中广泛存在的事物之间联系的一种有效表示手段，因此，对图的计算是一种常见的计算模式，而图计算会涉及到在相同数据上的不断更新以及大量的消息传递，如果采用 MapReduce 去实现，会产生大量不必要的序列化和反序列化开销。现有的图计算系统并不适用于 Google 的应用场景，因此 Google 设计并实现了 Pregel 图计算模型。Pregel 是 Google 继 MapReduce 之后提出的又一个计算模型，与 MapReduce 的离线批处理模式不同，它主要用于图的计算。该模型的核心思想源于著名的 BSP 计算模型。Dremel 是 Google 提出的一个适用于 Web 数据级别的交互式数据分析系统，通过结合列存储和多层次的查询树，Dremel 能够实现极短时间内的海量数据分析。Dremel 支持着 Google 内部的一些重要服务，比如 Google 的云端大数据分析平台 Big Query。Google 在 VLDB 2012 发表的文章中介绍了一个内部名称为 PowerDrill 的分析工具，PowerDrill 同样采用了列存储，且使用了压缩技术将尽可能多的数据装载进内存。PowerDrill 与 Dremel 均是 Google 的大数据分析工具，但是其关注的应用场景不同，实现技术也有很大差异。Dremel 主要用于多数据集的分析，而 PowerDrill 则主要应用于大数据量的核心数据集分析，数据集的种类相较于 Dremel 的应用场景会少很多。由于 PowerDrill 是设计用来处理少量的核心数据集，因此对数据处理速度要求极高，所以其数据应当尽可能的驻留在内存，而 Dremel 的数据则存储在磁盘中。除此之外，PowerDrill 与 Dremel 在数据模型、数据分区等方面都有明显的差别。从实际的执行效率来看，Dremel 可以在几秒内处理 PB 级的数据查询，而 PowerDrill 则可以在 30 至 40 秒里处理 7820 亿个单元格的数据，处理速度快于 Dremel。二者的应用场景不同，可以相互补充。

微软提出了一个类似 MapReduce 的数据处理模型，称之为 Dryad，Dryad 模型主要用来构建支持有向无环图(Directed Acycline Graph，DAG)类型数据流的并行程序。Cascading 通过对 Hadoop MapReduce API 的封装，支持有向无环图类型的应用。Sector/sphere 可以视为一种流式的 MapReduce，它由分布式文件系统 Sector 和并行计算框架 sphere 组成。Nephele/PACTs 则包括 PACTs(Parallelization Contracts)编程模型和并行计算引擎 Nephele。MapReduce 模型基本成为了批处理类应用的标准处理模型，很多应用开始尝试利用 MapReduce 加速其数据处理。

实时数据处理是大数据分析的一个核心需求。很多研究工作正是围绕这一需求展开的。前面介绍了大数据处理的两种基本模式，而在实时处理的模式选择中，主要有三种思路：

1) 采用流处理模式。虽然流处理模式天然适合实时处理系统，但其适用领域相对有限。

流处理模型的应用主要集中在实时统计系统、在线状态监控等。

2) 采用批处理模式。近几年来，利用批处理模型开发实时系统已经成为研究热点并取得了很多成果。从增量计算的角度出发，Google 提出了增量处理系统 Percolator，微软则提出了 Nectar 和 DryadInc。三者均实现了大规模数据的增量计算。

3) 二者的融合。有不少研究人员尝试将流处理和批处理模式进行融合，主要思路是利用 MapReduce 模型实现流处理。

2.4 大数据处理工具

关系数据库在很长的时间里成为数据管理的最佳选择，但是在大数据时代，数据管理、分析等的需求多样化使得关系数据库在很多场景不再适用。这里对现今主流的大数据处理工具进行一个简单的归纳和总结。

Hadoop 是目前最为流行的大数据处理平台。Hadoop 最先是 Doug Cutting 模仿 GFS、MapReduce 实现的一个云计算开源平台，后贡献给 Apache。Hadoop 已经发展成为包括文件系统(HDFS)、数据库(HBase、Cassandra)、数据处理(MapReduce)等功能模块在内的完整生态系统(Ecosystem)。某种程度上可以说 Hadoop 已经成为了大数据处理工具事实上的标准。

对 Hadoop 改进并将其应用于各种场景的大数据处理已经成为新的研究热点，主要的研究成果集中在对 Hadoop 平台性能的改进、高效的查询处理、索引构建和使用、在 Hadoop 之上构建数据仓库、Hadoop 和数据库系统的连接、数据挖掘、推荐系统等。

除了 Hadoop，还有很多针对大数据的处理工具。这些工具有些是完整的处理平台，有些则是专门针对特定的大数据处理应用。表 2-2 归纳总结了现今一些主流的处理平台和工具，这些平台和工具或是已经投入商业使用，或是开源软件。在已经投入商业使用的产品中，绝大部分也是在 Hadoop 基础上进行功能扩展，或者提供与 Hadoop 的数据接口。

表 2-2 大数据处理平台和工具

Category		Examples
Platform	Local	Hadoop、MapR、Cloudera、Hortonworks、InfoSphere BigInsights、ASTERIX
	Cloud	AWS、Google Compute Engine、Azure
Database	SQL	Greenplum、Aster Data、Vertica
	NoSQL	HBase、Cassandra、MongoDB、Redis
	NewSQL	Spanner、Megastore、F1
Data Warehouse		Hive、HadoopDB、Hadapt
Data Processing	Batch	MapReduce、Dryad
	Stream	Storm、S4、Kafka
Query Language		HiveQL、Pig Latin、DryadLINQ、MRQL、SCOPE
Statistic and Machine Learning		Mahout、Weka、R
Log Processing		Splunk、Loggly

2.5 大数据时代面临的新挑战

大数据时代的数据存在着如下几个特点：多源异构、分布广泛、动态增长、先有数据后有模式。正是这些与传统数据管理迥然不同的特点，使得大数据时代的数据管理面临着新的挑战，下面会对其中的主要挑战进行详细分析。

4.1.1大数据集成

数据的广泛存在性使得数据越来越多的散布于不同的数据管理系统中，为了便于进行数据分析需要进行数据的集成。数据集成看起来并不是一个新的问题，但是大数据时代的数据集成却有了新的需求，因此也面临着新的挑战。

（1）广泛的异构性。传统的数据集成中也会面对数据异构的问题，但是在大数据时代这种异构性出现了新的变化。主要体现在：1）数据类型从以结构化数据为主转向结构化、半结构化、非结构化三者的融合。2）数据产生方式的多样性带来的数据源变化。传统的数据主要产生于服务器或者是个人电脑，这些设备位置相对固定。随着移动终端的快速发展，手机、平板电脑、GPS 等产生的数据量呈现爆炸式增长，且产生的数据带有很明显的时空特性。3）数据存储方式的变化。传统数据主要存储在关系数据库中，但越来越多的数据开始采用新的数据存储方式来应对数据爆炸，比如存储在 Hadoop 的 HDFS 中。这就必然要求在集成的过程中进行数据转换，而这种转换的过程是非常复杂和难以管理的。

（2）数据质量。数据量大不一定就代表信息量或者数据价值的增大，相反很多时候意味着信息垃圾的泛滥。一方面很难有单个系统能够容纳下从不同数据源集成的海量数据；另

一方面如果在集成的过程中仅仅简单地将所有数据聚集在一起而不做任何数据清洗，会使得过多的无用数据干扰后续的数据分析过程。大数据时代的数据清洗过程必须更加谨慎，因为相对细微的有用信息混杂在庞大的数据量中。如果信息清洗的粒度过细，很容易将有用的信息过滤掉。清洗粒度过粗，又无法达到真正的清洗效果，因此在质与量之间需要进行仔细的考量和权衡。

4.1.2大数据分析

传统意义上的数据分析(analysis)主要针对结构化数据展开，且已经形成了一整套行之有效的分析体系。首先利用数据库来存储结构化数据，在此基础上构建数据仓库，根据需要构建数据立方体进行联机分析处理 (OLAP, Online Analytical Processing)，可以进行多个维度的下钻(Drill-down)或上卷(Roll-up)操作。对于从数据中提炼更深层次的知识的需求促使数据挖掘技术的产生，并发明了聚类、关联分析等一系列在实践中行之有效的方法。这一整套处理流程在处理相对较少的结构化数据时极为高效。但是，随着大数据时代的到来，半结构化和非结构化数据量的迅猛增长，给传统的分析技术带来了巨大的冲击和挑战，主要体现在：

（1）数据处理的实时性(Timeliness)。随着时间的流逝，数据中所蕴含的知识价值往往也在衰减，因此很多领域对于数据的实时处理有需求。随着大数据时代的到来，更多应用场景的数据分析从离线(offline)转向了在线(online)，开始出现实时处理的需求，比如 KDD 2012 最佳论文所探讨的实时广告竞价问题。大数据时代的数据实时处理面临着一些新的挑战，主要体现在数据处理模式的选择及改进。在实时处理的模式选择中，主要有三种思路：即流处理模式、批处理模式以及二者的融合。虽然已有的研究成果很多，但是仍未有一个通用的大数据实时处理框架。各种工具实现实时处理的方法不一，支持的应用类型都相对有限，这导致实际应用中往往需要根据自己的业务需求和应用场景对现有的这些技术和工具进行改造才能满足要求。

（2）动态变化环境中索引的设计。关系数据库中的索引能够加速查询速率，但是传统的数据管理中模式基本不会发生变化，因此在其上构建索引主要考虑的是索引创建、更新等的效率。大数据时代的数据模式随着数据量的不断变化可能会处于不断的变化之中，这就要求索引结构的设计简单、高效，能够在数据模式发生变化时很快的进行调整来适应。目前存在一些通过在 NoSQL 数据库上构建索引来应对大数据挑战的一些方案，但总的来说，这些方案基本都有特定的应用场景，且这些场景的数据模式不太会发生变化。在数据模式变更的

假设前提下设计新的索引方案将是大数据时代的主要挑战之一。

（3）先验知识的缺乏。传统分析主要针对结构化数据展开，这些数据在以关系模型进行存储的同时就隐含了这些数据内部关系等先验知识。比如我们知道所要分析的对象会有哪些属性，通过属性我们又能大致了解其可能的取值范围等。这些知识使得我们在数据分析之前就已经对数据有了一定的理解。而在面对大数据分析时，一方面是半结构化和非结构化数据的存在，这些数据很难以类似结构化数据的方式构建出其内部的正式关系；另一方面很多数据以流的形式源源不断的到来，这些需要实时处理的数据很难有足够的时间去建立先验知识。

4.1.3大数据隐私问题

隐私问题由来已久，计算机的出现使得越来越多的数据以数字化的形式存储在电脑中，互联网的发展则使数据更加容易产生和传播，数据隐私问题越来越严重。

（1）隐性的数据暴露。很多时候人们有意识地将自己的行为隐藏起来，试图达到隐私保护的目的。但是互联网，尤其是社交网络的出现，使得人们在不同的地点产生越来越多的数据足迹。这种数据具有累积性和关联性，单个地点的信息可能不会暴露用户的隐私，但是如果有办法将某个人的很多行为从不同的独立地点聚集在一起时，他的隐私就很可能会暴露，因为有关他的信息已经足够多了，这种隐性的数据暴露往往是个人无法预知和控制的。从技术层面来说，可以通过数据抽取和集成来实现用户隐私的获取。而在现实中通过所谓的“人肉搜索”的方式往往能更快速、准确的得到结果，这种人肉搜索的方式实质就是众包(Crowd sourcing)。大数据时代的隐私保护面临着技术和人力层面的双重考验。

（2）数据公开与隐私保护的矛盾。如果仅仅为了保护隐私就将所有的数据都加以隐藏，那么数据的价值根本无法体现。数据公开是非常有必要的，政府可以从公开的数据中来了解整个国民经济社会的运行，以便更好地指导社会的运转。企业则可以从公开的数据中了解客户的行为，从而推出针对性的产品和服务，最大化其利益。研究者则可以利用公开的数据，从社会、经济、技术等不同的角度来进行研究。因此大数据时代的隐私性主要体现在不暴露用户敏感信息的前提下进行有效的数据挖掘，这有别于传统的信息安全领域更加关注文件的私密性等安全属性。统计数据库数据研究中最早开展数据隐私性技术方面的研究，近年来逐渐成为相关领域的研究热点。保护隐私的数据挖掘(privacy preserving data mining)这一概念，很多学者开始致力于这方面的研究。主要集中于研究新型的数据发布技术，尝试在尽可能少

损失数据信息的同时最大化的隐藏用户隐私。但是数据信息量和隐私之间是有矛盾的，因此尚未出现非常好的解决办法。Dwork 在 2006 年提出了新的差分隐私(Differential Privacy)方法。差分隐私保护技术可能是解决大数据中隐私保护问题的一个方向，但是这项技术离实际应用还很远。

（3）数据动态性。大数据时代数据的快速变化除了要求有新的数据处理技术应对之外，也给隐私保护带来了新的挑战。现有隐私保护技术主要基于静态数据集，而在现实中数据模式和数据内容时刻都在发生着变化。因此在这种更加复杂的环境下实现对动态数据的利用和隐私保护将更具挑战。

4.1.4大数据能耗问题

在能源价格上涨、数据中心存储规模不断扩大的今天，高能耗已逐渐成为制约大数据快速发展的一个主要瓶颈。从小型集群到大规模数据中心都面临着降低能耗的问题，但是尚未引起足够多的重视，相关的研究成果也较少。在大数据管理系统中，能耗主要由两大部分组成：硬件能耗和软件能耗，二者之中又以硬件能耗为主。理想状态下，整个大数据管理系统的能耗应该和系统利用率成正比。但是实际情况并不像预期情况，系统利用率为 0 的时候仍然有能量消耗。针对这个问题，《纽约时报》和麦肯锡经过一年的联合调查，最终在《纽约时报》上发表文章《Power, Pollution and the Internet》。调查显示 Google 数据中心年耗电量约为 300 万瓦，而 Facebook 则在 60 万瓦左右。最令人惊讶的是在这些巨大的能耗中，只有 6%-12%的能量被用来响应用户的查询并进行计算。绝大部分的电能用以确保服务器处于闲置状态，以应对突如其来的网络流量高峰，这种类型的功耗最高可以占到数据中心所有能耗的 80%。从已有的一些研究成果来看，可以考虑以下两个方面来改善大数据能耗问题：

（1）采用新型低功耗硬件。从纽约时报的调查中可以知道绝大部分的能量都耗费在磁盘上。在空闲的状态下，传统的磁盘仍然具有很高的能耗，并且随着系统利用率的提高，能耗也在逐渐升高。新型非易失存储器件的出现，给大数据管理系统带来的新的希望。闪存、PCM 等新型存储硬件具有低能耗的特性。虽然随着系统利用率的提高，闪存、PCM 等的能耗也有所升高，但是其总体能耗仍远远低于传统磁盘。

（2）引入可再生的新能源。数据中心所使用的电能绝大部分都是从不可再生的能源中产生的。如果能够在大数据存储和处理中引入诸如太阳能、风能之类的可再生能源，将在很大程度上缓解能耗问题。这方面的工作很少，有研究人员探讨了如何利用太阳能构建一个绿

色环保的数据库。

4.1.5大数据处理与硬件的协同

硬件的快速升级换代有力的促进了大数据的发展，但是这也在一定程度上造成了大量不同架构硬件共存的局面。日益复杂的硬件环境给大数据管理带来的主要挑战有：

（1）硬件异构性带来的大数据处理难题。整个数据中心（集群）内部不同机器之间的性能会存在着明显的差别，因为不同时期购入的不同厂商的服务器在 IOPS、CPU 处理速度等性能方面会有很大的差异。这就导致了硬件环境的异构性(Heterogeneous)，而这种异构性会给大数据的处理带来诸多问题。一个典型的例子就是 MapReduce 任务过程中，其总的处理时间很大程度上取决于 Map 过程中处理时间最长的节点。如果集群中硬件的性能差异过大，则会导致大量的计算时间浪费在性能较好的服务器等待性能较差的服务器上。这种情况下服务器的线性增长并不一定会带来计算能力的线性增长，因为“木桶效应”制约了整个集群的性能。一般的解决方案是考虑硬件异构的环境下将不同计算强度的任务智能的分配给计算能力不同的服务器，但是当这种异构环境的规模扩展到数以万计的集群时问题将变得极为复杂。

（2）新硬件给大数据处理带来的变革。所有的软件系统都是构建在传统的计算机体系结构之上，即 CPU-内存-硬盘三级结构。CPU 的发展一直遵循着摩尔定律，且其架构已经从单核转入多核。因此需要深入研究如何让软件更好的利用 CPU 多核心之间的并发机制。由于机械特性的限制，基于磁性介质的硬盘（Hard Disk Drive, HDD）的读写速率在过去几十年中提升不大，而且未来也不太可能出现革命性的提升。基于闪存的固态硬盘(Solid State Disk,SSD)的出现从硬件层为存储系统结构的革新提供了支持，为计算机存储技术的发展和存储能效的提高带来了新的契机。SSD 具有很多优良特性，主要包括极高的读写性能、抗震性、低功耗、体积小等，因此正得到越来越广泛的应用。但是直接将 SSD 应用到现有的软件上并不一定会带来软件性能的大幅提升。Sang-Won Lee 等人的研究表明虽然 SSD 的读写速率是 HDD 的 60~150 倍，基于 SSD 的数据库系统的查询时间却仅仅提升了不到 10 倍。二者之间的巨大差距主要是由 SSD 的一些特性造成的，这些特性包括：SSD 写前擦除特性导致的读写操作代价不对称、SSD 存储芯片的擦除次数有限等。软件设计之时必须仔细考虑这些特性才能够充分利用 SSD 的优良特性。与大容量磁盘和磁盘阵列相比，固态硬盘的存储容量相对较低，单位容量的价格远高于磁盘。且不同类型的固态硬盘产品性能差异较大，将固态硬盘直

接替换磁盘应用到现有的存储体系中难以充分发挥其性能。因此现阶段可以考虑通过构建HDD 和 SSD 的混合存储系统来解决大数据处理问题。当前混合存储系统的实现主要有三种思路：HDD 作为内存的扩展充当 SSD 写缓冲；HDD 和 SSD 同做二级存储；SSD 用作内存的扩展充当 HDD 读写缓冲。国外的 Google、Facebook，国内的百度、淘宝等公司已经开始在实际运营环境中大规模的使用混合存储系统来提升整体性能。在这三级结构之中，内存的发展处于一个相对缓慢的阶段，一直没有出现革命性的变化。构建任何一个软件系统都会假设内存是一个容量有限的易失结构体。随着以 PCM 为代表的 SCM 的出现，未来的内存极有可能会兼具现在内存和磁盘的双重特性，即处理速度极快且非易失。虽然 PCM 尚未有可以大规模量产的产品推出，但是各大主流厂商都对其非常重视，三星电子在 2012 年国际固态电路会议(ISSCC 2012)上发表了采用 20nm 工艺制程的容量为 8G 的 PCM 元件。一旦 PCM 能够大规模的投入使用，必将给现有的大数据处理带来一场根本性的变革。譬如前面提到的流处理模式就可以不再将内存的大小限制作为算法设计过程中的一个主要考虑因素。

4.1.6大数据管理易用性问题

从数据集成到数据分析，直到最后的数据解释，易用性应当贯穿整个大数据的流程。易用性的挑战突出体现在两个方面：首先大数据时代的数据量大，分析更复杂，得到的结果形式更加的多样化，其复杂程度已经远远超出传统的关系数据库。其次大数据已经广泛渗透到人们生活的各个方面，很多行业都开始有了大数据分析的需求。但是这些行业的绝大部分从业者都不是数据分析的专家，在复杂的大数据工具面前，他们只是初级的使用者(Naïve Users)。复杂的分析过程和难以理解的分析结果限制了他们从大数据中获取知识的能力。这两个原因导致易用性成为大数据时代软件工具设计的一个巨大挑战。关于大数据易用性的研究仍处于一个起步阶段。从设计学的角度来看易用性表现为易见(Easy to discover)、易学(Easyto learn)和易用 (Easy to use)。要想达到易用性，需要关注以下三个基本原则：

（1）可视化原则(Visibility)。可视性要求用户在见到产品时就能够大致了解其初步的使用方法，最终的结果也要能够清晰的展现出来。未来如何实现更多大数据处理方法和工具的简易化和自动化将是一个很大的挑战。除了功能设计之外，最终结果的展示也要充分体现可视化的原则。可视化技术是最佳的结果展示方式之一，通过清晰的图形图像展示直观地反映出最终结果。但是超大规模的可视化却面临着诸多挑战，主要有：原位分析、用户界面与交互设计、大数据可视化、数据库与存储、算法、数据移动、传输和网络架构、不确定性的量

化、并行化、面向领域与开发的库、框架以及工具、社会、社区以及政府参与等。

（2）匹配原则(Mapping)。人的认知中会利用现有的经验来考虑新的工具的使用。譬如一提到数据库，了解的人都会想到使用 SQL 语言来执行数据查询。在新工具的设计过程中尽可能将人们已有的经验知识考虑进去，会使得新工具非常便于使用，这就是所谓的匹配原则。MapReduce 模型虽然将复杂的大数据处理过程简化为 Map 和 Reduce 的过程，但是具体的 Map 和 Reduce 函数仍需要用户自己编写，这对于绝大部分没有编程经验的用户而言仍过于复杂。如何将新的大数据处理技术和人们已经习惯的处理技术和方法进行匹配将是未来大数据易用性的一个巨大挑战。这方面现在已经有了些初步的研究工作。针对 MapReduce 技术缺乏类似 SQL 标准语言的弱点，研究人员开发出更高层的语言和系统。典型代表有 Hadoop 的 HiveQL 和 Pig Latin、Google 的 Sawzall、微软的 SCOPE 和 DryadLINQ 以及 MRQL 等。

（3）反馈原则(Feedback)。带有反馈的设计使得人们能够随时掌握自己的操作进程。进度条就是一个体现反馈原则的经典例子。大数据领域关于这方面的工作较少，有部分学者开始关注 MapReduce 程序执行进程的估计。传统的软件工程领域，程序出现问题之后有比较成熟的调试工具可以对错误的程序进行交互式的调试，相对容易找到错误的根源。但是大数据时代很多工具其内部结构复杂，对于普通用户而言这些工具近似于黑盒(black box)，调试过程复杂，缺少反馈性。如果未来能够在大数据的处理中大范围的引入人机交互技术，使得人们能够较完整的参与整个分析过程，会有效的提高用户的反馈感，在很大程度上提高易用性。

满足三个基本原则的设计就能够达到良好的易用性。从技术层面来看，可视化、人机交互以及数据起源技术都可以有效的提升易用性。而在这些技术的背后，海量元数据管理的问题是需要我们特别关注的一个问题。元数据是关于数据的数据，数据之间的关联关系以及数据本身的一些属性大都是靠元数据来表示的。可视化技术离不开元数据的支持，因为如果无法准确的表征出数据之间的关系，就无法对数据进行可视化的展示。数据起源技术更是离不开元数据管理技术。因为数据起源需要利用元数据来记录数据之间包括因果关系在内的各种复杂关系，并通过这些信息来进行相关的推断。如何在大规模存储系统中实现海量元数据的高效管理将会对大数据的易用性产生重要影响。

4.1.7性能测试基准

关系数据库产品的成功离不开以 TPC 系列为代表的测试基准的产生。正是有了这些测

试基准，才能够准确的衡量不同数据库产品的性能，并对其存在的问题进行改进。目前尚未有针对大数据管理的测试基准，构建大数据测试基准面临的主要挑战有：

（1）系统复杂度高。大数据管理系统的类型非常多，很多公司针对自己的应用场景设计了相应的数据库产品。这些产品的功能模块各异，很难用一个统一的模型来对所有的大数据产品进行建模。

（2）用户案例的多样性。测试基准需要定义一系列具有代表性的用户行为，但是大数据的数据类型广泛，应用场景也不尽相同，很难从中提取出具有代表性的用户行为。

（3）数据规模庞大。这会带来了两方面的挑战。首先数据规模过大使得数据重现非常困难，代价很大。其次在传统的 TPC 系列测试中，测试系统的规模往往大于实际客户使用的数据集，因此测试的结果可以准确的代表系统的实际性能。但是在大数据时代，用户实际使用系统的数据规模往往大于测试系统的数据规模，因此能否用小规模数据的测试基准来代表实际产品的性能是目前面临的一个挑战。数据重现的问题可以尝试利用一定的方法来去产生测试样例，而不是选择下载某个实际的测试数据集。但是这又涉及到如何使产生的数据集能真实反映原始数据集的问题。

（4）系统的快速演变。传统的关系数据库其系统架构一般比较稳定，但是大数据时代的系统为了适应数据规模的不断增长和性能要求的不断提升，必须不断的进行升级，这使得测试基准得到的测试结果很快就不能反映系统当前的实际性能。

（5）重新构建还是复用现有的测试基准。如果能够在现有的测试基准中选择合适的进行扩展的话，那么将极大减少构建新的大数据测试基准的工作量。可能的候选测试标准有 SWIM(Statistical Workload Injector for MapReduce)、MRBS、Hadoop 自带的 GridMix、TPC-DS、YCSB++等。

现在已经开始有工作尝试构建大数据的测试基准，比如一些针对大数据测试基准的会议 WBDB 2012、TPCTC 2012 等。但是也有观点认为当前讨论大数据测试基准的构建为时尚早。Yanpei Chen 等通过对 7 个应用 MapReduce 技术的实际产品的负载进行了跟踪和分析，认为现在根本无法确定大数据时代的典型用户案例。因此从这个角度来看并不适合构建大数据的测试基准，还有很多基础性的问题亟待解决。

总的来说，构建大数据的测试基准是有必要的。但是面临的挑战非常多，要想构建一个类似 TPC 的公认的测试标准难度很大。

本章小结

本章内容首先介绍了大数据处理的基本流程和大数据处理模型，接着介绍了大数据的关键技术，其中，云计算是大数据的基础平台和支撑技术，本章以 Google 的相关技术为主线，详细介绍 Google 以及其他众多学者和研究机构在大数据技术方面已有的一些工作，包括文件系统、数据库系统、索引和查询技术、数据分析技术等；接下来，介绍了大数据处理平台和工具，就目前技术发展现状而言，Hadoop 已经成为了大数据处理工具事实上的标准。最后，介绍大数据时代面临的新挑战，包括大数据集成、大数据分析、大数据隐私问题、大数据能耗问题、大数据处理与硬件的协同、大数据管理易用性问题以及性能测试基准。

参考文献

[1]孟小峰, 慈祥. 大数据管理：概念、技术与挑战. 计算机学报, 2013 年第 8 期.

（注：本章绝大多数内容来自上面的参考文献）

附录 1:任课教师介绍

林子雨(1978－),男,博士,厦门大学计算机科学系助理教授,主要研究领域为数据库,数据仓库,数据挖掘.

主讲课程：《大数据技术基础》

办公地点：厦门大学海韵园科研 2 号楼

E-mail: ziyulin@xmu.edu.cn

个人网页：http://www.cs.xmu.edu.cn/linziyu