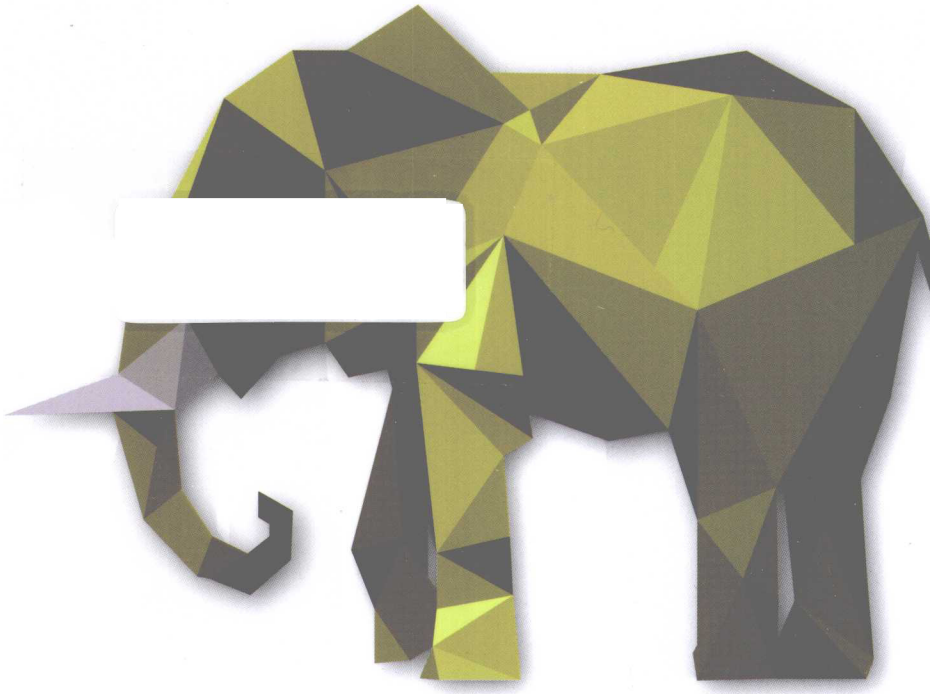
YARN RPC MapReduce HDFS Storm on YARN

Hadoop 2.0-YARN
核心技术实践

周维 编著

- 资深Hadoop 2.0-YARN专家倾力撰写
- 涵盖Hadoop 2.0的架构、重要组件、主要计算模式、资源调度
- 通过示例分析的形式降低读者的学习难度
- 部分案例直接选自BAT 中的工程实例



清华大学出版社

YARN RPC MapReduce HDFS Storm on YARN

Hadoop 2.0-YARN 核心技术实践

周维 编著

清华大学出版社
北 京

内 容 简 介

本书基于长期的教学实践以及同国内外顶尖公司的交流合作编写完成，系统介绍了 Hadoop 2.0-YARN 的基本概念与运行模式。全书共分为 7 章。内容涵盖 Hadoop 2.0 的架构、重要组件、主要计算模式、资源调度等重要问题。第 1 章回顾了 YARN 的起源，并与 Hadoop 1.0 进行了对比分析。第 2 章介绍了 YARN 的基本框架，对 YARN 中最重要的几个组件，如资源管理、节点管理、应用程序管理等做了说明。第 3 章通过 Hadoop 2.0-YARN 的安装、编译，以及简单的 MapReduce 调试示例，让读者能够迅速掌握 YARN 的基本操作，使得读者有一个初步的实践体验。第 4 章对 YARN 的通信原理和过程进行讨论，通过 Protocol Buffer、YARN RPC 的实例分析让读者理解 YARN 的通信协调过程。之后转入对 YARN 状态机进行深入分析，详细介绍了 YARN 中 4 类状态机的转换过程，同时提供了 YARN 状态机监控软件设计案例。第 5 章对基于 YARN 的几种计算模式（MapReduce、Storm、Spark）进行了讨论，每一种模式都提供了相应的安装步骤、案例分析。第 6 章叙述了 YARN 调度器，详细分析了 YARN 资源调度负载模拟器——SLS 和 Google 第三代调度器 omega 的基本原理，并分别给出了两种调度器的运行实例。第 7 章通过 Tez 和显示工作流引擎设计，使得读者对 YARN 工作流运行情况有一定了解。

本书最大的特点是理论与实践结合，通过示例分析的形式降低了读者的学习难度，避免了理论学习的枯燥性，本书的部分案例直接选自 BAT 中的工程实例，这使得本书更具有实战性。广大本科和研究生同学，可以参照本书实例，为他们进行分布式、云计算平台学习，专业课项目设计或毕业论文提供参考。本书也可作为业界研发人员的工程实践参考。

图书在版编目（CIP）数据

Hadoop 2.0-YARN 核心技术实践 / 周维编著. —北京：清华大学出版社，2015
ISBN 978-7-302-41139-0

Ⅰ. ①H…　Ⅱ. ①周…　Ⅲ. ①数据处理软件　Ⅳ. ①TP274

中国版本图书馆 CIP 数据核字（2015）第 182263 号

责任编辑：冯志强　薛　阳
封面设计：铁海音
责任校对：胡伟民
责任印制：李红英

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张：**13.25　　**字　数：**330 千字
版　　次：2015 年 9 月第 1 版　　**印　次：**2015 年 9 月第 1 次印刷
印　　数：1～3000
定　　价：49.00 元

产品编号：063111-01

前　言

随着计算机、互联网技术的发展，很多以前只能在单机上运行的程序现在越来越呈现出分布化、网络化的特点，近几年来，云计算、大数据更成为炙手可热的社会关注重心。在当前信息爆炸的时代，每天都在产生大量的信息数据，而如何高效地对这些信息进行处理成为计算机研发人员必须面对的挑战。虽然陆续提出过并行计算、网格计算等方案，但是在面临大规模，高效应用需求时都不是很理想。Hadoop 的诞生，很好地契合了当前全球计算机技术发展的潮流，由于其稳定性、可扩展性、开源性，Hadoop 成为国内外公司在云计算时代的首选支撑平台。

作者从事分布式教学与研究十余年，在同国内外顶尖公司的交流中，我们发现 Hadoop 很适合作为一个分布式课程教学实践与研究平台，因为 Hadoop 既包含了独立组件的运行，也包含了分布化的不同组件之间的通信，还包含了分布式系统的架构设计等，Hadoop 事实上成为一个集大成的分布式系统。最为难能可贵的是，Hadoop 是完全开源的系统，这使得我们有机会深入其中进行分析、研究。因此，近年来，作者逐步在分布式教学与研究中引入 Hadoop 系统，取得了明显的效果。在对 Hadoop 进行分析调研的基础上，我们认识到当前 Hadoop 书籍还存在一些不满意的地方，例如：① 由于 Hadoop 2.0-YARN 在 2013 年 11 月才发布稳定版，因此，对 YARN 的公开研究资料还不多，目前可查询到的 YARN 中文版书籍还比较少，这些书籍虽然也不错，但是更适合一个 Hadoop 从业人员作为技术手册，对于广大普通读者来说，入门门槛过高，而且看过后基本还是不会编写 YARN 程序。② 虽然网络上也有不少这方面的资料，但是由于网络写作的随意性，资料很零散且存在很多错误，因此让初学者和普通开发人员很难快速入手。

基于此，作者着手编著一本针对 Hadoop 2.0-YARN 的书籍。本书在写作过程中注重实践教学，因此配备有很多实际例程，这样读者可以边看书、边安装、边调试，因此降低学习难度，加快学习进度，同时，本书对 YARN 中的一些核心内容的剖析也很有价值，如① 出租车 Storm on YARN 实时处理实例；② YARN 状态机信息捕捉；③ YARN 调度模拟器——SLS 分析，并与 Google 第三代调度器 Omega 进行对比分析。这些资料都既有文字说明，又有实际代码。基于我们长期的实践，有些内容是首次披露，网络上也没有的，所以对读者会有很大的吸引力。本书在写作工程中，也力求和公司的工程项目结合起来，因此，部分案例直接选自 BAT 中的工程实例，这使得本书更具有实战性。

作为广大的本科和研究生同学，可以参照本书实例进行研究或修改，为他们进行分布式、云计算平台学习，专业课项目设计或毕业论文提供参考。本书也可作为业界研发人员的工程实践提供参考。

本书第 1~6 章由周维老师主要负责编写，第 7 章由薛岗老师主要负责编写。另外来自阿里的杨辉先生，袁硕同学，以及在百度实习过的刘笠熙同学、周可人同学都提出了宝贵

意见并设计了部分案例。此外，还要特别感谢很多参与代码调试的研究生同学，这其中包括麦超、刘建坤、刘长春、范航凯、傅央、张浩、向文坤、魏征、孙淋川、罗洁等。没有大家的帮助，这本书也不可能这么快写完，在此对所有支持本书编著的人表示衷心的感谢。

由于时间仓促，本书难免存在不妥之处，请读者批评指正。

编者

2015 年 4 月

目　录

第 1 章　YARN 的前世今生

1.1　Hadoop 基本情况回顾

当 Doug Cuting 决定发起 Hadoop 项目的时候，他肯定没有想到，不到十年的时间里，Hadoop 会成为一股席卷全球的铺天盖地的洪流。由于 Hadoop 的稳定性、可扩展性，最主要是其开源性，如今，它已成为众多公司，如 Yahoo、Twitter、FaceBook、IBM，还有国内的百度、腾讯、阿里等的业务平台。在当今移动互联网、云计算的大背景下，Hadoop 似乎成为不可或缺的一部分。那么为什么 Hadoop 会受到如此重视，首先让我们从 Hadoop 的起源讲起。

随着人类社会信息技术的进步，特别是互联网的发展，每天都在产生大量的信息数据。然而如何高效地对这些信息进行处理一直困扰着计算机研发人员。虽然已经提出了并行计算、分布式计算、网格计算等方案，但是效果都不是很理想。这种情况对那些站在科技发展尖端的企业，如 Google，表现得特别明显。作为全球最大的搜索引擎企业，Google 公司需要对全球的海量网页进行及时的处理，然而当时（2000 年）的技术手段却显得有点力不从心。因此，这种业务需求促使 Google 的工程师设想开发一种全新的系统。2003、2004 年，Google 相继发表了两篇文章论述了他们为应对海量数据而开发的两项新技术 GFS 和 MapReduce，这在当时引起了广泛热议，使得许多正为数据急剧膨胀而头痛不已的企业看到了希望。与此同时，作为开源项目 Nutch（Web 搜索引擎）的创始人，Doug Cutting 正在为 Nutch 架构无法扩展到支撑拥有数十亿网页的网络感到头疼。他敏锐地意识到，他和他的团队可以利用 GFS 和 MapReduce 的原理来开发一个新的框架，可以极大地提高 Nutch 的承载能力。2006 年 2 月，他们从 Nutch 转移出来成为一个独立的 Lucene 子项目，被正式命名为 Hadoop。在 Yahoo 的大力支持下， Hadoop 项目日益活跃并受到广泛关注。2008 年 1 月，Hadoop 成为 Apache 下的顶级项目。在 2008 年 2 月，雅虎宣布其搜索引擎产品部署在一个拥有 1 万个节点的 Hadoop 集群上。随着 Hadoop 项目取得的巨大成功，Doug Cutting 在 2010 年 9 月被选为 Apache 软件基金会（Apache Software Foundation）主席。

Hadoop 最初设计的时候是为了搜索引擎业务服务的（如 Yahoo、Google 等公司）。然而始料不及的是，由于 Hadoop 的良好特性，Hadoop 被应用到了各行各业。那么，Hadoop 真的能解决这些业务需求吗？随着 Hadoop 应用的深入，人们发现 Hadoop 在一些特定业务领域并不能很好发挥，学术界和工业界都活跃起来，希望通过各种改进来提高 Hadoop 的性能。因此下一代 Hadoop 的研发被提上日程，在 Hadoop 0.23 的版本中，引入了革命性的、

全面的更新，这个版本被正式命名为 YARN（或 MRv2）。

下面我们用一个直观的图来展示 Hadoop 的发展历程，如图 1-1 所示。

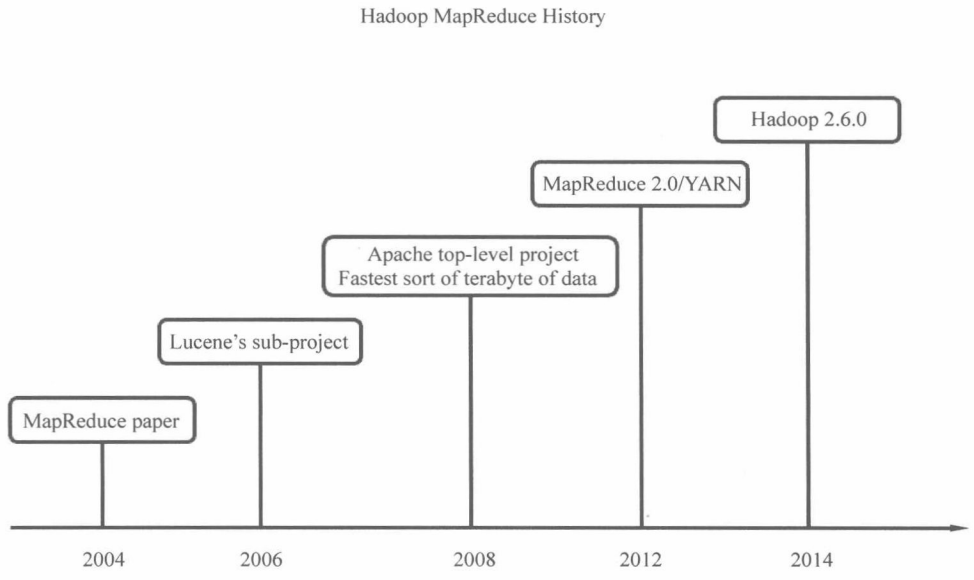


图 1-1 Hadoop 的发展历程

1.2 为什么我们需要 YARN

由于 Hadoop 1.0 的良好特性，Hadoop 1.0 被应用到了各行各业。各大著名公司都相继采用，如 Facebook、Yahoo、Twitter、百度、腾讯、阿里等。随着 Hadoop 1.0 应用的日益丰富，原先 Hadoop 1.0 设计中存在的一些问题显露出来，主要表现在以下几方面。

（1）存在单点故障、影响可扩展性、稳定性。

Hadoop 1.0 中 Namenode 和 JobTracker 设计成单一节点，一旦该节点出现故障，对整个系统有致命性影响。这个节点是整个系统的单点故障源（SPOF），严重制约了整个 Hadoop 的可扩展性和可靠性。Datanode 定期向 Namenode 发送 Block Report，这些数据是占用内存空间的，随着 Hadoop 集群存储空间的增多，这些 Block Report 也会越来越多，因此，Namenode 的内存容量成为制约 Hadoop 集群规模的一个重要因素。JobTracker 也存在同样的问题，由于随着 JobTracker 业务的增长，处理的任务也随之增长，从而造成内存消耗过大，同时任务失败的概率也随之增加。

（2）计算模式比较单一，只支持 MapReduce 模式。

Hadoop 1.0 中对计算模式的支持比较单一，只支持 MapReduce 模式。然而，实现业务处理却存在多种需求。例如需要对实时业务进行及时处理，于是产生了 Storm，Spark 系统；需要对具有工作流性质的业务进行处理，于是产生了 Tez 项目，能够提供更底层的 DAG

编程接口；需要基于图业务性质进行处理，于是产生了 Giraph 图计算项目。这些系统的产生是对 MapReduce 模式的有效补充，然而，如何将这些系统和 Hadoop 有效整合起来成为一个亟待解决的问题。

（3）Map 和 Reduce 模式绑定太死，不灵活。

在 Hadoop 1.0 最初设计的时候，Map 和 Reduce 是作为一个整体来提供给用户使用的。然后，并不是每个业务都同时需要这两个操作，有的时候只需要其中之一。然而，现有机制中 TaskTracker 会把任务分成 Map Task Slot 和 Reduce Task Slot，如果当前节点仅存在 Map Task 或者 Reduce Task，则会造成资源浪费。

（4）资源管理方案不灵活。

Hadoop 1.0 中采用静态 slot 资源分配策略，在节点启动前为每个节点配置好 slot 总数，一旦启动后，就不能动态更改。此外，Hadoop 1.0 中将 slot 分为 Map slot 和 Reduce slot 两种，且不允许交换共享。而节点在真实运行情况下往往是 Map slot 紧张而 Reduce slot 空闲，或者反之。在任务紧张的时候很大地阻碍了 Hadoop 的性能。再者，Hadoop 默认情况下是按照 2GB 内存和 1 个 CPU 来为每个 slot 分配资源，如果 task 使用内存过多（如 5G），则很可能把这个节点撑爆。如果 task 使用内存过小（如 1G），又可能造成资源浪费。而且对资源的描述比较单一，只是从 CPU 和内存数量进行描述，实际上影响任务的资源还有很多，例如带宽、传输速度、存储空间等。

上述几点只是列举了 Hadoop 1.0 在实际运行过程中存在的一些主要问题，虽然科学家和软件工程人员已经从各方面着手来解决这些问题，但是始终缺乏一个完整的，能够和 Hadoop 1.0 紧密整合在一起的解决方案，这样说来，大家也就能够理解为什么新一代 Hadoop-YARN 的研发工作呼之欲出了。

1.3　YARN 和 Hadoop 1.0 对比分析

随着 Hadoop 1.0 应用的日益丰富，原先 Hadoop 1.0 设计中存在的一些问题显露出来，针对这些问题，Hadoop 开发团队的研究与工程技术人员深入讨论，在充分收集各方提出的意见，经过长时间准备和开发后，设计出了 YARN，Hadoop 1.0 中存在的一些主要问题在 YARN 中都得到了很好地回应，下面我们来进行对比分析。

1.3.1　体系结构对比

图 1-2 是 Hadoop 1.0 的体系结构图，在 Hadoop 1.0 中 Namenode 和 JobTracker 设计成单一节点，一旦出现故障，对整个系统有致命性影响。该节点是整个系统的单点故障源（SPOF），严重制约了整个 Hadoop 的可扩展性和可靠性。随着 JobTracker 业务的增长，处理的任务也随之增长，从而造成内存消耗过大，同时任务失败的概率也随之增加。举一个直观的例子，假设 Hadoop 1.0 同时运行有 10 个 job，每个 job 又有 1000 个 task，假设这些

task 都运行在不同的机器上，那么，我们有 10 000 个节点（Datanode）在同时运行，如果每个 Datanode 每隔 5 分钟定期向 JobTracker 发送信息，那么 JobTracker 的工作压力会很大，所以正常情况下 Hadoop 1.0 的集群管理规模只能达到 4000 台左右。这也是造成 Hadoop 1.0 存在单点故障的直接原因，影响了 Hadoop 1.0 的可扩展性、稳定性。

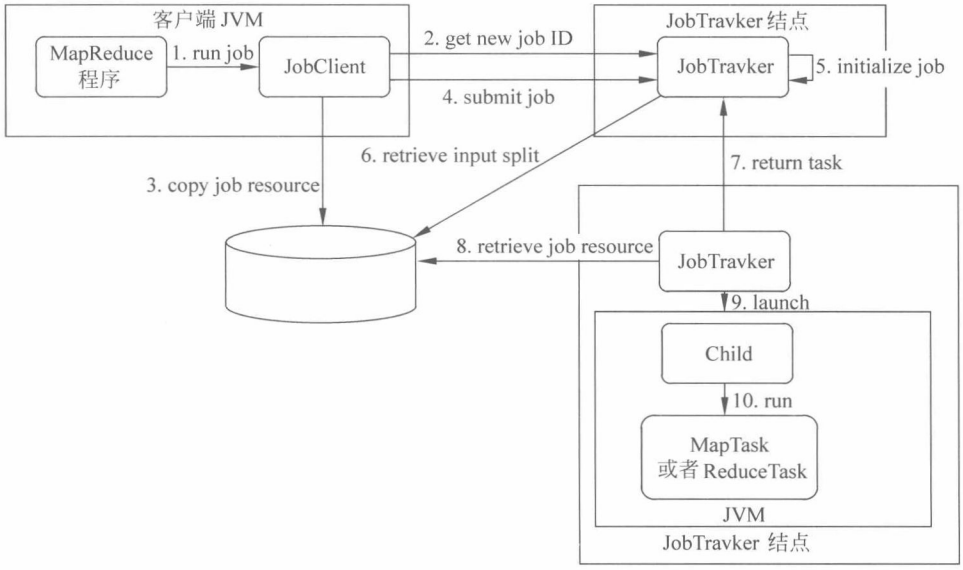


图 1-2 Hadoop 1.0 的体系结构

JobTracker 任务过重是造成 Hadoop 1.0 单点故障和可扩展性差的主要原因，在 Hadoop 1.0 中 JobTracker 主要完成两大任务：资源的管理和作业控制。由于需要对集群中几千台机器的资源进行管理，并对运行在集群上的各种作业进行调度控制，JobTracker 显得有些穷于应付。因此，在 YARN 的设计中，资源的管理和作业控制是被分离开的。如图 1-3 所示，资源的管理由 ResourceManager 来处理，而作业控制则由 ApplicationMaster 来处理。为了便于说明，我们还是举 Hadoop 1.0 的那个例子来说明。假设 YARN 同时运行有 10 个 job，每个 job 又有 1000 个 task，假设这些 task 都运行在不同的机器上，那么，我们有 10 000 个节点（Datanode）在同时运行，同时每个 Datanode 每隔 5 分钟定期发送信息。那么，在 YARN 中会给每个 job 分配一个 ApplicationMaster 来负责管理其下运行的 1000 个 task。这 1000 个节点只会向对应的 ApplicationMaster 汇报情况，只有一些精简的汇总信息才会通过这 10 个 ApplicationMaster 和 ResourceManager 通信。因此，对 ResourceManager 来说，它只会收到 10 个 ApplicationMaster 的信息，相对于 Hadoop 1.0 中 JobTracker 需要处理 10 000 个节点（Datanode）的信息而言，ResourceManager 可以更专注于整个集群的资源管理。需要注意的是，每个 job 分配的 ApplicationMaster 可以在同一台机器上，也可以在不同机器上，这样整个集群的作业控制负载相对均衡，不会集中在一个中心点。将作业控制分离出来的结果是 ResourceManager 的负担减轻，可以专注于整个集群的资源管理。这种体系结构上的变更解决了 Hadoop 1.0 的单点故障问题，整个集群也可以轻松扩展到上万台机器。

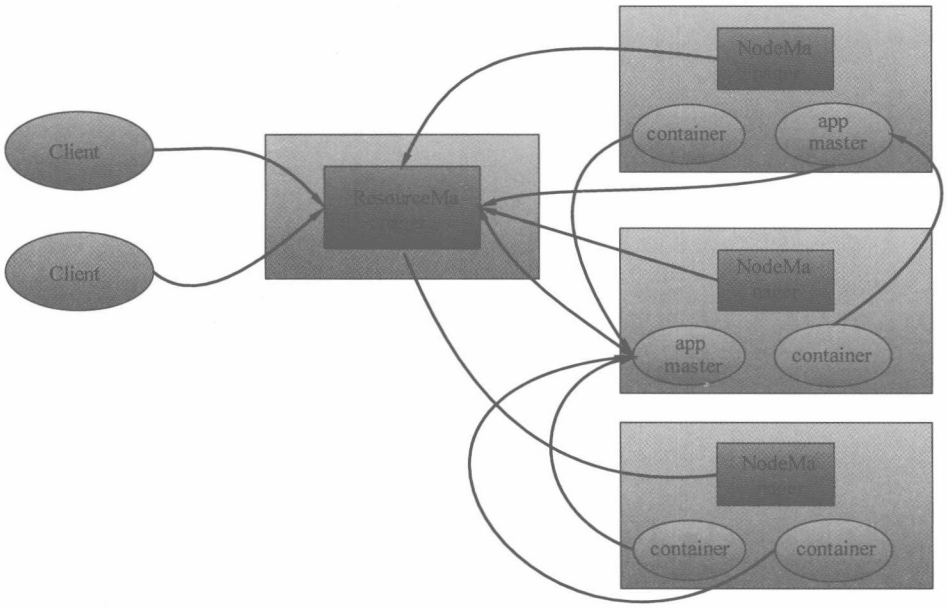


图 1-3　YARN 的体系结构

1.3.2　运算框架对比

Hadoop 1.0 设计的时候并没有预料到它会应用得如此广泛。Hadoop 1.0 中对计算模式的支持比较单一，只支持 MapReduce 模式。然而，各类公司的业务处理却存在多种需求。例如需要对实时业务进行及时处理，于是产生了 Storm、Spark 系统；需要对具有工作流性质的业务进行处理，于是产生了 Tez 项目，能够提供更底层的 DAG 编程接口；需要基于图业务性质进行处理，于是产生了 Giraph 图计算项目。这些系统的产生是对 MapReduce 模式的有效补充，然而，如何将这些系统和 Hadoop 有效整合起来成为一个亟待解决的问题。

在 Hadoop 2.0 中，将计算框架 MapReduce 和 HDFS 分离开来，用 YARN 进行集群的资源管理，这样，在 YARN 上可以运行不同的计算框架，而 MapReduce 只是其中一个选项。其他的如 Storm、Spark、Tez 等都可以在 YARN 上运行，这极大地丰富了 YARN 的应用场景。

总地来说，YARN 还是 Master/Slave 的架构，不同于 Hadoop 1.0 的是，在整个资源管理框架中，ResourceManager 充当了 Master 的角色，NodeManeger 则充当了 Slave 的角色。ResourceManager 的职责是对多个 NodeManager 的资源进行统一管理和调度。

如图 1-4 所示为 Hadoop 2.0 的框架结构，YARN 是这种双层架构中的第二层，这种将资源管理相关的功能与应用程序相关的功能分开的做法，使得 Hadoop 能够轻易支持多种计算框架。它使得 Hadoop 不再仅限于支持 MapReduce 这种单一的计算模型，而是极大地扩充了其生态圈。

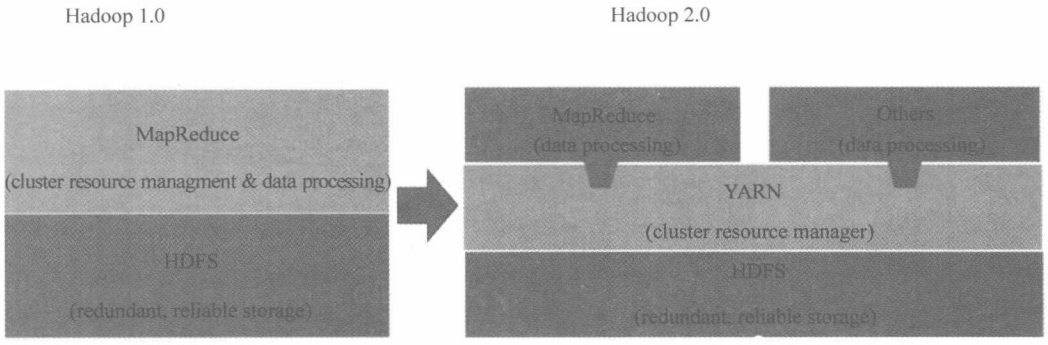


图 1-4 Hadoop 2.0 的框架结构

1.4 Hadoop 生态系统

由于其稳定性、可扩展性，开源性，Hadoop 已成为众多公司，如 Yahoo、Twitter、FaceBook、IBM，还有国内的百度、腾讯、阿里等的业务平台。从底层到上层，围绕 Hadoop 衍生出众多的服务商，提供不同种类的技术支持。

图 1-5 描述了整个 Hadoop 完善的生态系统。“Hadoop 即服务”为基础架构和不同的应用分析场景提供了强大而广泛的支撑；同时，在大规模数据分析和处理上，Hadoop 也能直接作为应用平台，通过基础数据库和各种编程语言为其开发相应的应用程序。Hadoop 具有很强的灵活性和可扩展性。Hadoop 2.0 之后的版本引入了 YARN 作为其资源管理与调度系统，但也可采用其他的第三方管理平台（如 Mesos）作为 Hadoop 的资源管理与调度系统。

1.5 小　　结

总结下来，YARN 框架相对于老的 Hadoop 1.0 框架具有很多优势，具体表现在以下几方面。

（1）将 Hadoop 1.0 中 JobTracker 的两大任务（资源的管理和作业控制）分离开，分别由 YARN 中的 ResourceManager 和 ApplicationMaster 来进行管理。这样使得 YARN 能够承载更多的业务，同时稳定性也大大提升。由于 JobTracker 和 Namenode 的限制，Hadoop 1.0 一般只能管理 4000~8000 台机器，不能对上万台机器服务进行有效管理。为了突破这个瓶颈，一些顶尖的大业务公司（例如百度等）通过修改 Hadoop 1.0，用三个 Namenode 来代替原来的一个 Namenode，这样提升整个 Hadoop 1.0 的业务承载能力。在 YARN 中，上述问题通过 ResourceManager 和 ApplicationMaster 的分离得到很好的解决，因此 YARN 对集群的管理能力得到极大提升。

（2）支持多计算框架。YARN 改变了 Hadoop 1.0 中只支持 MapReduce 计算框架的问题，将计算框架和底层存储和调度分离开，不但支持离线的 batch MapReduce 计算，同时对其他计算框架 Storm、Spark、Giraph 等都能很好地支持，丰富了应用场景。

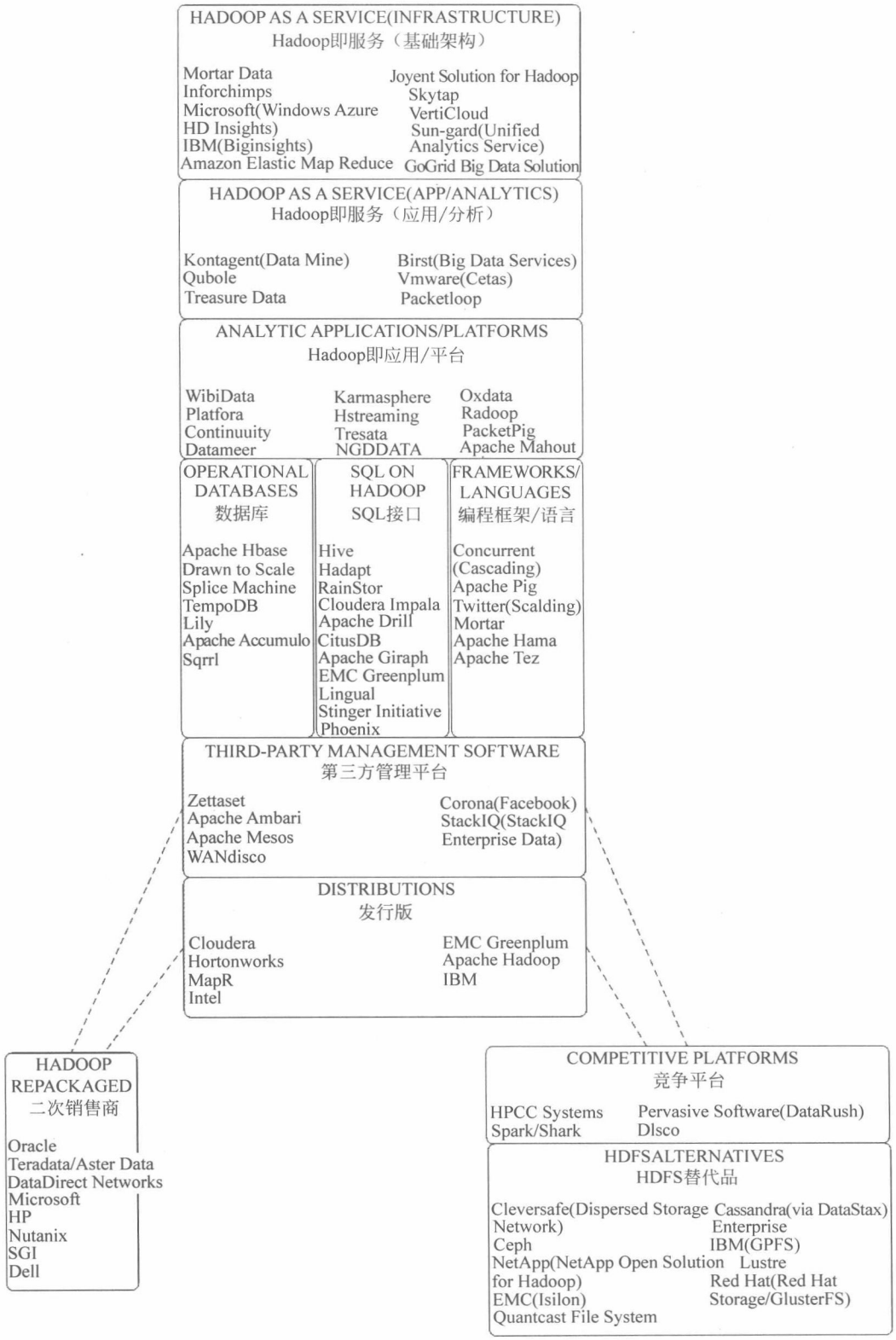


图 1-5　Hadoop 生态系统

第 2 章　YARN 基本框架

2.1　YARN 基本框架

在 1.2 节中，我们列举了 Hadoop 1.0 设计中存在的主要不足，包括存在单点故障、计算模式比较单一等。这些问题都成为 YARN 设计中重点需要解决的问题。在 Hadoop 2.0-YARN 中，将计算框架 MapReduce 和 HDFS 分离开来，用 YARN 进行集群的资源管理，这极大地丰富了 YARN 的应用场景。

如图 2-1 所示，YARN 上可以运行不同的计算框架，而 MapReduce 只是其中一个选项。其他的如 Storm、Spark、Tez 等都可以在 YARN 上运行。

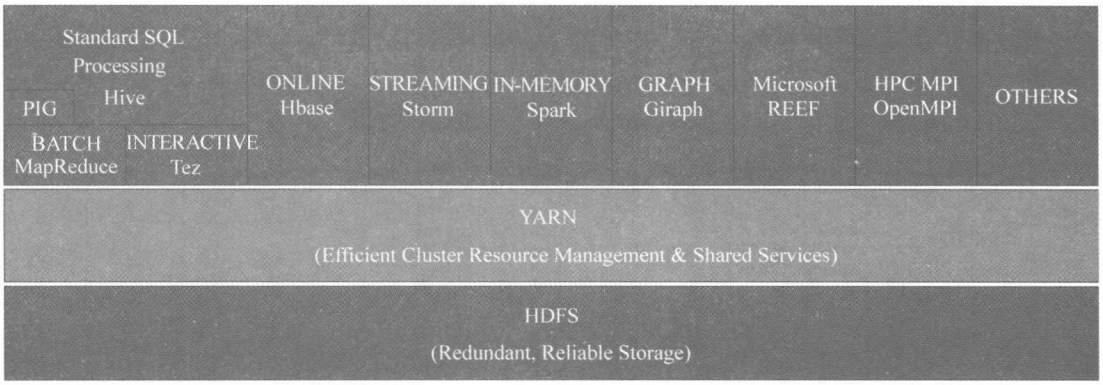


图 2-1　YARN 框架

YARN 作为资源管理、任务调度的一个框架，主要包含三大模块，即 ResourceManager（RM）、NodeManager、ApplicationMaster（AM）。这其中，ResourceManager 全局负责所有资源的监控、分配和管理；ApplicationMaster 负责每一个具体应用程序的调度和协调；NodeManager 负责每一个节点的维护。几个模块之间的关系如图 2-2 所示。

相对于 Hadoop 1.0，YARN 最重要的变化是将资源的管理和作业控制分离开（即将 Hadoop 1.0 中 JobTracker 两个最重要的功能，即资源管理和作业调度与监控拆分开），分别由全局的 ResourceManager 负责资源管理、ApplicationMaster 负责作业调度和监控。对于所有的 applications，RM 拥有绝对的控制权和对资源的分配权。而每个 AM 则会和 RM 协商资源，同时和 NodeManager 通信来执行和监控 task。

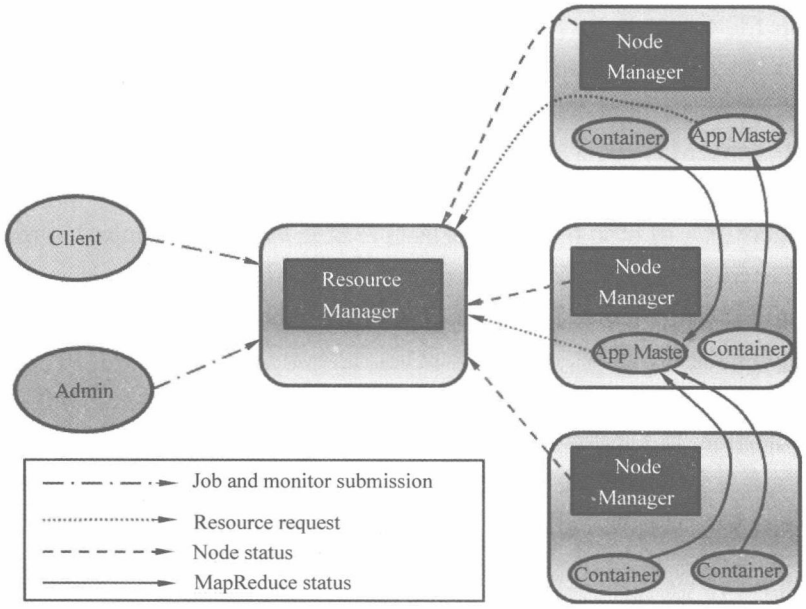


图 2-2　三个模块间的关系

2.2　ResourceManager

ResourceManager 在 Hadoop 第二代框架 YARN 中负责整个集群的资源管理和分配，是一个全局的资源管理系统。它包含多个组件，其中包括 ApplicationMasterService、NodeListManager、ContainerAllocatioin 等，它们分别负责 Application 管理、状态机管理等，如图 2-3 所示。

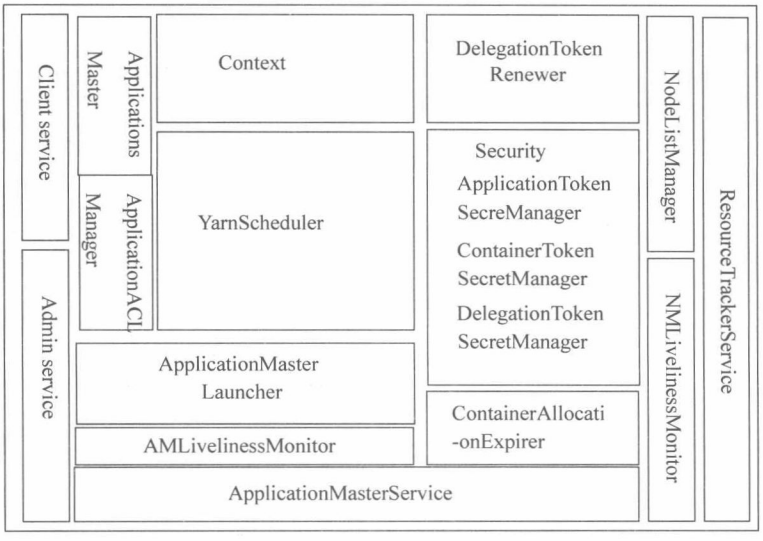


图 2-3　ResourceManager 框架图