基于 Hadoop 的电子商务网站访问日志处理与分析 潘权义 武汉理工大学

武汉理工大学

（申请理学硕士学位论文）

基于 Hadoop 的电子商务网站访问日志处理与分析

培养单位：计算机科学与技术学院

学科专业：计算机软件与理论

研 究 生：潘权义

指导教师：王舜燕 教授

2012 年 4 月

分类号_____ 密　　级_____

UDC _____ 学校代码 10497

武漢理工大學

学　位　论　文

题　目 基于 Hadoop 的电子商务网站访问日志处理与分析

英　文 Process and Analysis of Access Log of

题　目 E-Commerce Site Based on Hadoop

研究生姓名 潘权义

指导教师 姓名 王舜燕 职称 教授 学位 博士

单位名称 武汉理工大学 邮编 430070

副指导教师 姓名_____ 职称_____ 学位_____

单位名称_____ 邮编_____

申请学位级别 硕士 学科专业名称 计算机软件与理论

论文提交日期 2012 年 4 月 论文答辩日期 2012 年 5 月

学位授予单位 武汉理工大学 学位授予日期_____

答辩委员会主席_____ 评阅人_____

_____

2012 年 4 月

独 创 性 声 明

本人声明，所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得武汉理工大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

签 名：_____日 期：_____

学位论文使用授权书

本人完全了解武汉理工大学有关保留、使用学位论文的规定，即学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权武汉理工大学可以将本学位论文的全部内容编入有关数据库进行检索，可以采用影印、缩印或其他复制手段保存或汇编本学位论文。同时授权经武汉理工大学认可的国家有关机构或论文数据库使用或收录本学位论文，并向社会公众提供信息服务。

（保密的论文在解密后应遵守此规定）

研究生（签名）：　　导师（签名）：　　日期

摘要

用户在访问 Web 站点的过程中，服务器会记录这些访问形成访问日志。对访问日志进行必要的处理可以获取大量的决策数据。就电子商务网站而言，处理网站访问日志有助于为网站的管理者提供决策支持进而指导网站运营，如改善网站结构提升用户体验；进行关键词营销提升流量、促进转化、提升效益；分析用户行为进行个性化的推荐和营销来提高网站的核心竞争力，在激烈的市场竞争中保持优势。

日志的处理与分析通常分为四个阶段：数据采集，数据预处理，分析算法的实施与数据可视化。常见的在线网站分析工具都能在不同程度上提供从日志采集、预处理与分析，直至提供包含各项关键绩效指标的可视化报告解决方案，然而随着电子商务网站不断发展，用户越来越多，业务越来越复杂，访问日志的分析也会变得异常复杂，同时以用户为中心的网站分析也变得更为重要，此时由电子商务网站自身组建分析团队，搭建自主的日志收集和处理平台的就变得非常必要。

Hadoop 是 Apache 基金会开发的一套分布式系统架构，以分布式文件系统 HDFS 和并行计算模型 MapReduce 为核心，为用户提供了系统底层细节透明的分布式基础架构。基于 Hadoop 对电子商务网站的访问日志进行预处理和分析，可以利用集群优势并行处理与分析日志，快速及时的为网站运营团队提供决策数据。

本文提出部署专用的日志服务器，由电子商务网站自身组件团队来完成日志处理与分析各个阶段的工作。采用 JavaScript 标记方式采集日志，基于 Hadoop 搭建自主网站分析平台处理日志数据，并结合用户数据进行网站分析并以 Web 报表的形式展示分析结果。在日志处理的过程中，对其中包含的海量 URL 进行识别是非常重要的，本文提出并实现了一种高效可行 URL 识别的算法。访问路径匹配是分析用户行为重要一环，本文也给出了一种简易的匹配算法和实现。本文在最后提出了一种数据密集型与计算密集型混合的集群协作模型，并将每个阶段的数据处理视为云服务，服务之间通过简单并且低耦合的接口调用完成调用，同时结合 Duboo 分布式计算服务框架完成了该集群协作模型的实现。

关键词：电子商务，访问日志处理，Hadoop，集群协作模型

Abstract

Access logs record the detailed visit information of Web pages. The necessity for processing access logs can elicit a large number of decision-making data. In terms of E-commerce sites, dealing with access logs help providing decision support for site managers, guiding operation and maintaining competitive edge.

There are four steps for process and analysis of access logs: data collection, preprocessing, analysis and implementation of algorithm and visualization. Common online web analytics tools more or less can provide visual reports of the key performance indicators after their integrated solution applied, but with the continuous development of E-commerce website, more users and more complex business structure lead to more complexity of analysis of access logs. At the same time, User-oriented web analytics become more important, building a platform to collect access logs and then process it become much necessary.

Hadoop developed by the Apache Software Foundation provides user a distributed infrastructure of transparent details of low layer. Process and analysis of access logs of E-commerce site based on Hadoop can takes advantage of parallel processing of cluster in order to obtain decision-making data in time.

This thesis puts forward an idea of deploying a dedicated platform to collect access logs and processing it by team of the E-commerce website. Use JavaScript asynchronous access generate access logs, build their own web analytics platform to deal with log data based on Hadoop, combine user data to conduct site analysis and then display data in the form of Web report. It’s an important step to identify the mass URL and Path in data processing, this thesis presents a URL recognition algorithm and its implementation, a simple access path match algorithm and its implementation. Finally, This thesis present a data-intensive and compute-intensive mixed data processing model, each stage of data processing is regarded as a cloud service called by other service through simple and low-coupled interface,then give a realization of the collaboration model of cluster using Duboo distribution computing framework.

KeyWord: E-Commerce, Access log processing, Hadoop, Cluster processing model

目　录

第 1 章 绪论

1.1 研究背景及意义

随着现代信息社会的飞速发展，互联网已成为人们日常生活不可缺少的一部分。人们在网络上发消息、邮件和微博，上传下载图片、音乐以及视频，利用搜索引擎查找信息，浏览购物网站、购买产品并完成支付甚至点击广告等等。

中国互联网络信息中心（CNNIC）在 2012 年初发布的《第 29 次中国互联网发展状况统计报告》显示截至 2011 年 12 月底，中国网民规模突破 5 亿[1]。伴随着网民数量的急剧增长，中国网民的购物比例也随着上升，网络购物用户总规模达到 1.94 亿人，网民中使用网络进行购物的比率达到 37.8%，网络购物市场交易规模达 7735.6 亿元，占到社会消费品零售总额的 4.3%，互联网商业价值突出显现。

互联网成为大数据快速增长的巨大驱动力之一，用户访问网站时会在服务器端产生大量的访问日志，对保存在服务器端的海量日志数据进行处理、分析和挖掘可以获得更多的信息。对于竞争激烈的电子商务网站来说，对这些日志进行处理与分析，可以为电子商务网站带来巨大的商业价值，提高电子商务网站的核心竞争力和带来可观的利润。

随着访问日志采集技术的发展，传统的由 Web 服务器根据页面访问生成日志记录的方式逐渐被采用 JavaScript 标记生成的访问日志的方式所取代。这种变化使得在数据处理、分析与挖掘的过程中，一些相应的技术手段也发生了根本性的变化，如用户识别和会话识别的方式。日志服务器端收集的日志数据随着用户数的增加呈现海量增加的趋势，复杂多变的业务与分析需求的多样化更是要求日志格式多样化，如何存储与分析这些日志，从中找到有价值的信息变成一件非常复杂且具有极大挑战性的工作。

Hadoop 是 Apache 基金会开发的一套分布式系统架构，为用户提供了系统底层细节透明的分布式基础架构，使得编写可以运行在分布式集群上处理海量数据的应用程序较为容易，采用 Hadoop 可以快速搭建海量数据存储与分析平台以利用集群优势并行处理与分析数据。到目前为止，上万个节点组成的集群搭载 Hadoop 已经被证明可以承担 PB 级别的数据处理与分析任务。

Hadoop 项目发展了数年，但是基于它进行分布式计算的具体实施流程、改进与优化方面在国内外还处于边应用边摸索阶段，深入研究和详细分析已有分布式计算平台框架有助于深刻的理解分布式计算及其编程模式，也有助于利用已有资源快速构建分布式计算平台来处理海量数据。

本文主要研究 Hadoop 在电子商务网站访问日志处理的应用。采用 Hadoop 来处理电子商务网站访问日志有助于快速从中获取有用的信息，为电子商务网站的管理者提供决策支持进而指导运营。如改善网站结构，进行关键词营销提升流量、促进转化、提升效益，在激烈的市场竞争中保持优势。因此这是一个非常有价值的课题，主要体现在以下几个方面：

（1）为电子商务网站提高流量，促进转化。

通过分析流量来源与转化的关系，可以找到最佳流量来源，对于流量来源为搜索引擎的访问可以分析搜索关键词优化（SEO）与营销（SEM）的效果，找到最佳的关键词集合。根据这些分析结果调整网站运营，如对转化率较高的合作站点追加投资可能会带来更大的流量和转化，对转化率较高的关键词进一步调整可改善 SEO 与点击付费广告（PPC）的效果，同样可以提高电子商务网站的流量与转化。

（2）优化与改善网站的结构

对访问日志进行处理可以获得页面点击分布热点图，频繁访问路径，页面停留时间，页面跳出率，登陆页蹦失率等各项数据，对其进行分析以便指导网站建设，提供高效的导航与站内搜索，降低登录页的崩失率，提升通向转化过程的诱导力，降低转化过程中关键页面的退出率来促进转化。

（3）分类网站的目标用户群，通过个性化推荐，提升可用性并促进转化。

通过分析访问日志并整合站内注册用户数据提供以用户为中心的网站分析数据，如对用户进行分类并获得各类用户群的购买特点，当用户回访网站的时候可以向其进行个性化推荐或者对其进行个性化的邮件营销，提升个性化推荐与营销效果。

（4）为进一步日志数据挖掘提供可靠的基础数据

解读日志数据无论从深度到广度都是一个复杂的流程，在获取数据到做出正确的决策钱将会有很长的路要走，各种需求会不断的出现，对基础的日志数据进行预处理后，在未来扩展需求的情况下，可以直接采用处理过的数据进行进一步的数据挖掘与分析。

1.2 研究现状

1.2.1 Hadoop 研究现状

数据的指数级增长首先向 Google、Yahoo！、Amazon 和 Microsoft 等处于市场领导者地位的公司提出了挑战，它们需要遍历 TB 级甚至 PB 级的数据来发现有用的信息。Google 率先推出了 MapReduce 算法用来应对其数据处理需求，而 Apache Hadoop 项目对 MapReduce 算法的开源实现。

Hadoop 从 2006 年具有雏形到 2011 年底发布了 1.0 的正式版，发展至今仍然是较为年轻的计算平台，在很多方面仍需要改进与完善，抑或针对业务进行功能定制。国内外大量 Hadoop 爱好者们都已积极参与到 Hadoop 的研究中，在这种背景下对 Hadoop 的研究是具有非常重要的意义

在学术方面，Hadoop 得到了广泛关注，多所著名大学以对 Hadoop 展开研究，其中包括斯坦福大学，加州大学伯克利分校，康奈尔大学，卡耐基梅隆大学等。一些国内高校和科研院所如中科院计算所，清华大学，中国人民大学等也开始对 Hadoop 展开相关研究，研究内容涉及 Hadoop 的数据存储，资源管理，作业调度，性能优化，系统可用性和安全性等多个方面。

在商业方面，Hadoop 技术已经在互联网领域得到了广泛的研究与应用。如 Facebook 使用 Hadoop 存储内部的日志拷贝以及数据挖掘和日志统计。Yahoo！利用 Hadoop 支持广告系统并处理网页搜索。在国内 Hadoop 同样也得到了许多公司的研究和应用，如百度使用 Hadoop 进行搜索日志分析和网页数据库的数据挖掘，阿里巴巴使用 Hadoop 进行商业数据的排序和搜索引擎的优化等，淘宝使用 Hadoop 存储和处理电子商务交易的相关数据，中国移动研究院基于 Hadoop 对数据进行分析并对外提供服务等等。

1.2.2 网站访问日志处理与分析研究现状

访问日志处理、分析以及基于访问日志的数据挖掘，主要目的是为了对网站进行分析。近年来网站分析行业内涌现了多样化的日志分析工具，本地日志分析工具如 Webalizer，AWStats，Splunk，在线日志分析工具如 Google Analytics，Omniture SiteCatalyst，WebTrends，Coremetrics 等等。无论是本地分析的工具还是在线的分析工具都提供丰富的分析功能与统计功能，并可以做一定程度的配置以提供各种各样的分析报表对网站进行分析。

然而多数本地分析工具只能用于单机分析，一旦日志数据的存储与分析超过单台服务器的处理能力，利用集群进行存储与分析将会面临诸多挑战，如管理和协调资源、调度任务与处理失效等等。而那些在线分析工具由于无法获取部分用户数据，所以提供的功能仍然有限，免费的在线分析工具通常对单个站点最大流量也有限制。

对电子商务网站来说，除了常见的网站分析的各种需求，对登录用户进行多维度的分析具有非常重要的意义，这些分析需要结合 CRM 管理系统中的数据进行，这些数据对于电子商务网站来说是属于最高级安全级别的隐私数据，不宜向第三方网站分析机构提供以防信息泄露，这使得采用第三方的分析工具难以整合这些数据。要进行更为复杂的业务分析、挖掘工作，依然需要依靠电子商务网站自身开发分析工具来完成这些自定义的处理与分析。

1.3 研究内容与论文的组织方式

本论文主要的研究内容按照以下章节进行组织，本章为第 1 章绪论，概述本文研究背景与现状，讨论本文的研究背景、意义与内容。

第 2 章：阐述 Hadoop 基本理论，深入分析其核心子项目 HDFS 分布式文件系统与 MapReduce 并行计算框架的实现原理，为研究海量日志数据存储与分析提供理论支撑与技术手段。

第 3 章：电子商务网站访问日志的处理与分析。分析网站访问日志采集、预处理、分析以及数据可视化的过程，进而结合电子商务基本理论与网站访问日志分析的目的，探讨采用 Hadoop 自主搭建日志处理平台的原因。

第 4 章：运用 Hadoop 技术对访问日志数据进行预处理的设计与实现。结合日志的具体采集方式对日志数据进行合并、清洗与预处理。日志预处理过程中 URL 的识别是非常重要的，本为对 URL 识别提出了一种算法并使用 java 语言进行了实现；最后本文集合 URL 识别对用户行为分析与路径分析提出一个集群协作模型，并结合 Alibaba Dubbo 开源框架对模型进行实现。

第 5 章：采用 Hadoop 对某电子商务网站访问日志进行处理，将处理结果以 Web 报表的形式进行展现，结合电子商务领域业务逻辑与关键绩效指标（KPI），对几组处理结果数据进行网站分析。

在论文的结束部分对采用 Hadoop 进行访问日志处理和分析进行了总结，并展望日后的研究方向等。

第 2 章 Hadoop 理论基础

2.1 Hadoop 简介

2.1.1 什么是 Hadoop

Hadoop 是 Apache 基金会采用 Java 语言开发的一个分布式计算框架，它以 Hadoop 分布式文件系统（HDFS）[2]和 MapReduce[3]为核心为用户提供了系统底层细节透明的分布式基础架构，使用户更加容易编写可以运行在分布式集群上的应用程序以充分利用集群的能力处理海量数据。Hadoop 项目的总体结构图如图 2-1 所示。

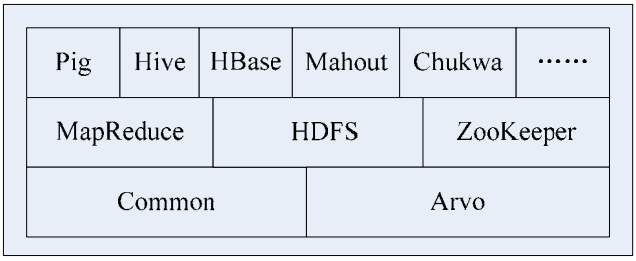

图 2-1 Hadoop 项目的总体结构图

在这个结构中，Common 为 Hadoop 其它子项目提供了通用工具，如 FileSystem API，RPC 等库。Avro 提供跨语言且高效的数据序列化的功能。

HDFS 与 MapReduce 是 Hadoop 项目的核心子项目。HDFS 是一个分布式文件系统的实现，可运行在商用计算机集群上提供高容错，高可用性、高吞吐量的数据访问，非常适合海量数据集上的应用。MapReduce 是一种编程模型，用于海量数据集的处理，可以解释为“ 任务的分解和结果的汇总 ”，Map 把输入分解成中间值 Key/Value 对，Reduce 把 Key/Value 合并成最终的输出。Zookeeper 是一个分布式的协调服务，提供分布式锁之类的服务以构建分布式应用。

Pig、Hive、HBase、Mahout、Chukwa 和 Sqop 等项目建立在 HDFS 和 MapReduce 基础上提供更高层面的应用，为 Hadoop 项目提供了强有力的补充。目前这些项目均已升级为 Hadoop 的顶级项目，Pig 用于检索海量数据集；Hive 适合在数据仓库上以类 SQL 的查询语言查询数据；HBase 是一个基于列存储分布式数据库；Mahout 提供一些可扩展的机器学习领域经典算法的实现；Sqoop 在数据库与 HDFS 之间提供高效的数据传输。

2.1.2 Hadoop 的历史

Hadoop 项目开始于 Nutch 项目，Nutch 项目又开始于 Lucene 项目。Nutch 是一个开源的 Web 搜索引擎项目，用于抓取数十亿的网页并维护索引和提供高质量的搜索。Lucene 项目是一个开源的全文检索引擎工具包。这三个项目均有 Doug Cutting 所创立。

2003 年至 2004 年期间，Google 先后发表论文介绍其分布式文件系统与运行其上的分布式计算模型，即“ The Google File System ”（GFS）[4]和 “ MapReduce：Simplified Data Processing on Large Clusters ” [5]的论文。此后 Doug Cutting 开始尝试实现 MapReduce 计算框架并对 Nutch 项目上的主要算法进行移植，这种新的实现提升了 Nutch 可扩展性。

2006 年 2 月，Nutch 分布式文件系统（NDFS，用于支撑采用 Nutch 抓取的海量网页数据）与 MapReduce 从 Nutch 项目中分离出来形成 Lucene 项目一套完整而独立的子项目，起名为 Hadoop。

2008 年，Hadoop 升级成为 Apache 的顶级项目，它的数个子项目后来也陆续升级为顶级项目，Hadoop 逐渐成为 Hadoop 项目和其一系列子项目的统称。

Hadoop 团队在 0.20 版本时进行了大量的 API 调整以改进扩展性以及为正式版本的发布做好准备。2011 年底，Apache 发布了 Hadoop 的 1.0 正式版。

2.1.3 Hadoop 的特点

用户可以将 Hadoop 部署在多台商用计算机上形成一个集群。利用 Hadoop 轻松的组织计算资源完成海量数据的存储与处理[6]。Hadoop 的与众不同之处在于以下几点：

（1）高扩展性：Hadoop 在集群中可用的计算机上分配数据并完成任务计算，集群的规模可以方便地扩展到成千上万的节点。

（2）经济性：Hadoop 的设计能够在商用计算机集群中实现向外扩展。向上扩展的代价是非常昂贵的，性能十倍于标准 PC 的机器其成本将大大超过同样数目 PC 的集群计算能力，向外扩展显然是更加经济的方式。

（3）高容错性：Hadoop 能够自动保存数据的多个副本，并且能够自动将失败的任务重新分配。能够在节点之间动态地移动数据并保持各节点的动态平衡。

（4）高效性：分布式文件系统的高效数据交互以及 MapReduce 结合本地数据处理的模式，为高效处理海量的信息作了基础准备。

2.1.4 Hadoop 的功能与作用

Hadoop 功能与作用体现在提升存储容量，保证数据安全，降低存储与分析成本，并行高效地处理数据，提高开发效率方面[7]。

Hadoop 在应对海量数据存储问题时采用了分布式存储方式，并提供了一个分布式文件系统的旗舰实现。这使得储存容量显著增长而不再受限于单台服务器的存储容量，存储容量达 PB 级。

Hadoop 分布式文件系统将文件划分为块并采用存储冗余数据块的方式保证了数据的安全性。存储数据的节点采用普通的商用服务器，可以显著降低实施分布式系统集群的硬件成本，这与传统的 SAN、NAS 等分布式文件系统相比有着巨大的优势。

Hadoop 的设计理念是将计算过程移动到数据所在节点而不是移动数据到计算节点，因为将 CPU 计算移动到数据的代价更小[8]。对于可分割可并行处理的数据集来说，分布式的数据存储与采用 MapReduce 对数据并行处理使得数据读取速度与数据处理吞吐量大大提高。Hadoop 会尽力在存储有数据的节点上发起 MapReduce 任务来保证数据的“ 本地化 ”，这种方式可以加快处理速度，避免耗尽网络带宽。

在分布式计算环境下处理与恢复系统的部分失效是一个非常大的挑战，MapReduce 采用了无共享的方式使得各个任务之间相互独立，并实现了系统失效检测与恢复，这使得开发人员可以不必充分了解这些内容进而有更多的精力关心应用逻辑，因此大大提高了开发效率。

Hadoop 是采用 Java 语言编写的并提供了相应的 API 接口，使用 Java 语言开发 MapReduce 程序非常方便[9]。Hadoop 同时提供了 Streaming API，使用 Unix 标准输入输出流作为 Hadoop 和应用程序之间的接口，这种架构使得 Map 和 Reduce 函数可以采用其它语言编写，如 Ruby，Python 等等，这对于那些不熟悉 java 语言的开发者同样可以采用 Hadoop 来处理海量数据集。

Hadoop 的分布式文件系统 HDFS、并行计算模型 MapReduce 与分布式数据库 HBase 技术等覆盖云计算领域关键支撑技术的几个重要方向，同时拥有丰富的子项目，这使得众多企业使用 Hadoop 搭建云计算平台，Hadoop 成为云计算体系中的一种 PaaS 技术[10]。如今，Hadoop 已成为云计算系统上进行程序开发和任务调度的主流基础软件平台，广泛应用在海量数据存储，并行处理，搜索引擎，数据挖掘等各种领域。

2.2 Hadoop 分布式文件系统

2.2.1 HDFS 简介

用于管理在计算机网络中跨多台机器存储文件的系统为分布式文件系统。Hadoop 整合了众多的文件系统，并以接口的形式提供了一个综合性的文件系统抽象，同时提供了这个抽象的分布式文件系统一个旗舰式的实现，被称为 HDFS。

HDFS 可以部署在商用计算机集群上提供高容错、高吞吐量、高可用海量数据访问，它的主要特点包括[11]：

超大文件：HDFS 可以提供 TB 甚至 PB 级的大文件存储而不适合对大量的小文件存储。目前，HDFS 已被扩展到上万个节点上以提供 PB 级的数据存储。

数据以流式访问：HDFS 的设计思路是数据集大多情况下是“一次写入，多次读取”并在此数据集上长时间进行各种分析任务。每次分析都会读取整个数据集或者数据集的很大一部分，所以读取整个数据集的时间延迟比读取一条记录的时间延迟更加重要。整个流式读取的过程中以批处理方式不与用户交互以极大的提升吞吐量。

商用计算机集群：HDFS 并不需要部署和运行在昂贵但可靠性较高的计算机上，而是运行在商用计算机上，这就意味着集群中出现节点故障的情况概率非常高。HDFS 能够自动恢复这种故障并且不让用户察觉到明显的中断，即被设计成用户透明的。

以上的几点使得 HDFS 在处理特定问题时会有局限性，如低延迟的数据访问以及存放大量的小文件，多用户写入及修改文件等等。

2.2.2 HDFS 的重要概念

HDFS 中数据块的设计、NameNode 与 DataNode 的主从架构是非常重要的概念，是理解 HDFS 的关键[12]。文件数据块与 NameNode、DataNode 的关系如图 2-2 所示：

数据块：块是文件系统中至关重要的概念，块是磁盘读写数据的最小单位。HDFS 作为一个分布式文件系统同样有块的概念，但 HDFS 的块是一个抽象的概念。HDFS 中的大文件被划分为多个分块，每个块作为一个独立的存储单元存储在 DataNode 的本地文件系统中，多个分块将分散到集群中进行存储，由 NameNode 统一管理文件的元数据信息。HDFS 中块大小默认为 64M，不足一个

块大小的文件并不会占用一个块的空间，HDFS 中的块巨大的原因是为了减少寻址时间相对于磁盘传输数据的时间的比率。在更快的硬盘传输速率的情况下，可以将块大小配置增大到 128M。

HDFS 中的块抽象可以带来诸多好处[13]。首先，文件的大小可以大于集群中任一磁盘的容量，块可以存放在集群中的任何一个磁盘上。极端情况下整个 HDFS 集群可以仅存储一个文件。二是可以简化存储系统的设计，因为计算一个磁盘能存储多少块相对容易，对文件的一些元数据信息如存取控制信息可以单独存储与管理。Hadoop 通过在块层面上对数据进行备份以提供高容错能力和高可用性。当数据块所在的磁盘损坏或者块校验和不正确，系统会自动从集群中其它机器读取另外一个副本并且实现块的复制使得块副本数量恢复到正常值。

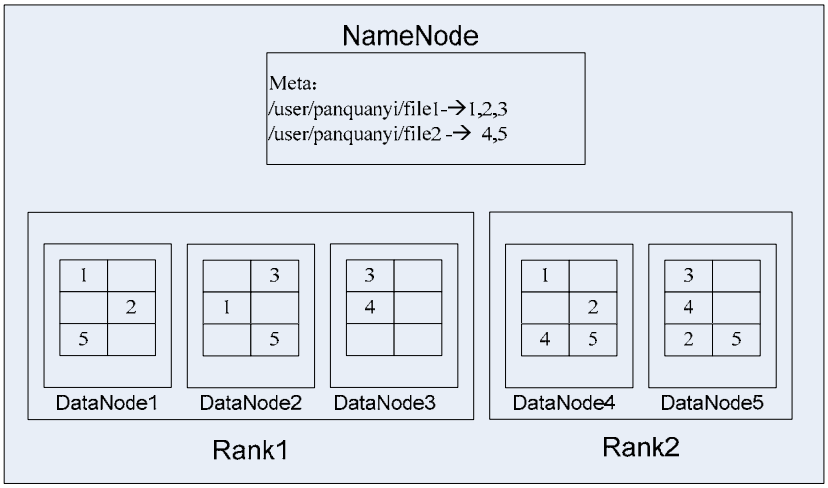

图 2-2 文件数据块与 NameNode、DataNode 的关系图

NameNode 与 DataNode：HDFS 采用 Master/Slave 架构，一个 HDFS 集群包含一个 NameNode 和多个 DataNode，即一个名称节点与多个数据节点。

NameNode 管理整个文件系统的命名空间，并处理客户端诸如打开、关闭、重命名和删除文件和目录等操作，决定块与 DataNode 的映射，维护整个文件系统的文件目录树以及索引这些目录等等。除此之外还记录着每个文件各个数据块所在的数据节点信息但并不永久保存而是在系统启动时交互 DataNode 重建。在进行文件读取的过程中，由 NameNode 想 Client 提供元数据信息。

DataNode 是 HDFS 中的工作节点。DataNode 在本地存储文件的数据块，响应客户端对数据库的读写请求。数据块的存储会随着 Hadoop 系统运行做出调整，如磁盘故障，节点故障或系统负载均衡等。DataNode 与会周期性的向 NameNode 报告数据块信息。

2.2.3 HDFS 中文件读取剖析

客户端以代理方式来与 HDFS 文件系统交互，整个过程由 Client、NameNode 和 DataNode 共同交互完成[14]。当客户端对 NameNode 发出读取请求，NameNode 返回存储文件的各个 DataNode 节点的信息，客户端直接向 DataNode 请求读取文件数据。由于数据块分散在整个集群上，NameNode 只需要向客户端响应元数据信息而无需响应数据内容，这种设计使得 NameNode 节点不会成为性能瓶颈，整个 HDFS 非常容易扩展以支持大量的并发客户端。但这种设计也使得 NameNode 的单点故障变的突出，目前对单点故障的解决方案多采用备份 NameNode，并且定时同步的方案将 NameNode 节点故障的风险降到最低。

文件读取的详细过程如图 2-3 所示，详细步骤为：

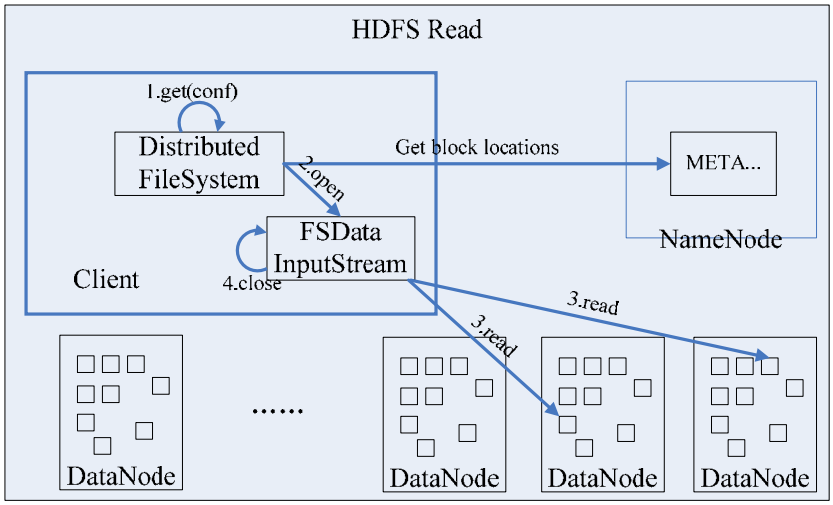

图 2-3 HDFS 读取过程图示

（1）客户端调用 FileSystem.get(conf)静态方法获取一个分布式文件系统的实例，通常情况为一个 HDFS 实例。

（2）客户端接着调用此实例的 open(path)方法打开欲读取的文件。此调用返回一个输入流对象，客户端将使用这个对象进行数据读取。

FileSystem 实例通过 RPC 调用从 NameNode 获取文件数据块和副本所在的 DataNode 位置。open(path)方法返回的是一个封装了 DFSInputStream 的 FSDataInputStream 输入流对象，它继承自 java 中的 DataInputStream 类并支持随机访问。从文件的随机位置访问数据对 MapReduce 过程中的数据分片处理是非常之重要的。

（3）客户端使用输入流对象读取数据。在读取数据的过程中由

DFSInputStream 顺序连接存储了所需数据块且距离最近的 DataNode 来获取数据，距离由网络的拓扑结构得出。当一个数据块读取完成后，输入流对象会建立与另一个数据块之间的连接并继续读取数据，对客户端来说多个块的数据流是一个连续的数据流，数据块之前的切换对客户端来说是透明的。

如果客户端本身就是一个 DataNode 并保存块的一个副本，那么数据将从本地读取。如果客户端与 DataNode 的通信发生故障或者块校验和不正确，客户端会自动尝试去读取数据块的其它副本，并向 NameNode 发送报告标记读取出错避免后续反复读取该 DataNode，这对客户端来说也是透明的。

（4）客户端读取完成后关闭输入流，至此文件读取完成。

2.2.4 HDFS 中的文件写入剖析

HDFS 中文件的写入也在客户端，NameNode 和 DataNode 共同交互完成[15]。文件写入的详细过程如图 2-4 所示，详细步骤为：

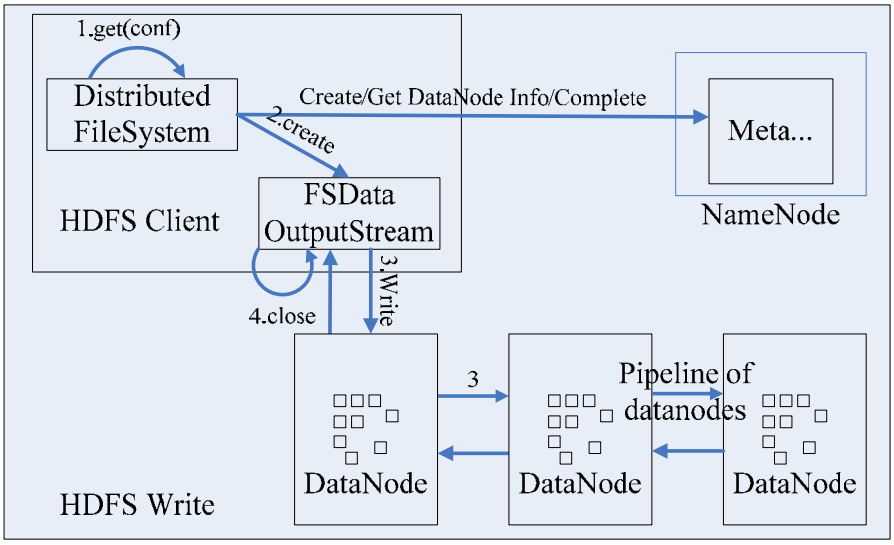

图 2-4 HDFS 写入过程图示

（1）同读取过程一样，首先获得一个分布式文件系统实例。

（2）通过调用 FileSystem 实例的 create(path)方法来创建文件。FileSystem 实例通过 RPC 调用 NameNode 在其命名空间创建一个新文件并返回一个输出流对象，客户端使用这个对象进行数据写出。

返回的输出流对象是一个封装了 DFSOutputStream 的 FSDataOutoutSream 对象，它继承自 java 中的 DataOutputStream 类。

（3）写出数据。写出的数据不会被立即写入到 DataNode 节点，而是由客户

端创建一个本地临时文件缓存这些数据（Staging），当数据超过一个块大小的时候，客户端才会交互 NameNode 获得用来保存此块的 DataNode 列表。

数据块在写入 DataNode 的过程中，多个副本的写入是同时进行的，DataNode 之间形成一个管道。以默认副本数为例，输出流中的数据首先写入管道中的第一个 DataNode，该 DataNode 将数据写入本地文件文件的同时将数据发送至管道中的第二个 DataNode，第二个 DataNode 以同样的方式写入本地文件的同时再发给第三个 DataNode，当收到管道中所有数据块写入都成功后，才进行下一个数据块的写入。

（4）当所有的数据块写入成功后，调用数据流对象的结束方法，通知 NameNode 节点写入完成，文件创建成功。

2.3 Hadoop 的计算模型

2.3.1 MapReduce 简介

MapReduce 是 Google 提出的一种简化分布式编程的软件架构，采用这种架构将海量数据分散在集群中的多个节点并行处理可以显著的缩短数据的处理时间，这使得 MapReduce 在海量数据处理上很有优势并在行业中得到了广泛的应用[16]。MapReduce 的架构方式似乎是一种暴力方法，但这也正是它的能力，它使得对数据进行快速的处理以挖掘有用的信息，其每次处理都涉及整个数据集。

MapReduce 主要思想是从函数式编程和分布式计算中借鉴来的，它计算包括 Map 和 Reduce 两个阶段，这也是 MapReduce 命名的来源。数据流图 2-5 如下：

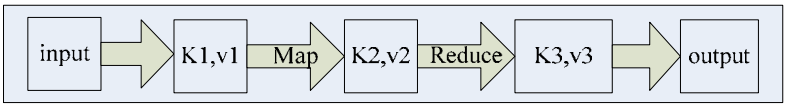

图 2-5 MapReduce 数据流图

在 Map 阶段，海量数据集将被分成多个子集并行数理，数据子集的输入被组织为一个键值对<k1,v1>列表，经过 Map 阶段的处理后输出另外一个键值对<k2,v2>列表。<k2,v2>列表将被按照 k2 排序，以 k2 为键聚合 v2 并进行拆分后作为 Reduce 阶段的输入。

在 Reduce 阶段，所有 Map 阶段的输出被聚合到一个<k2,v2>概念列表中，相同 k2 的键值对被组织成一个新的键值对<k2,list<v2>>并成为 Reduce 阶段的输入，经过 Reduce 阶段的处理后，输出为<k3,v3>的过程。

2.3.2 Hadoop MapReduce 的整体架构

Hadoop 对 MapReduce 进行了开源实现，MapReduce 和 HDFS 运行在同一个集群上以便使得计算所需的数据本地化[17]。

Map 任务通常一次处理一个输入分片中的数据，过小的输入分片会使得数据块寻址开销较大，同时每次 Map 处理少量数据使得 Map 任务数过多，启动与维护 Map 进程的额外开销过大。过大的输入分片会使得 Map 任务数太少，每个 TaskTracker 节点的 Map 任务处理时间太长而失去并行计算的优势。通常情况下，一个输入分片的大小等于 HDFS 块的大小。

Hadoop 对 MapReduce 实现采用 Master/Slave 架构，框架由一个 JobTracker 节点和多个 TaskTracker 节点组成。处理过程中数据流如图所示：

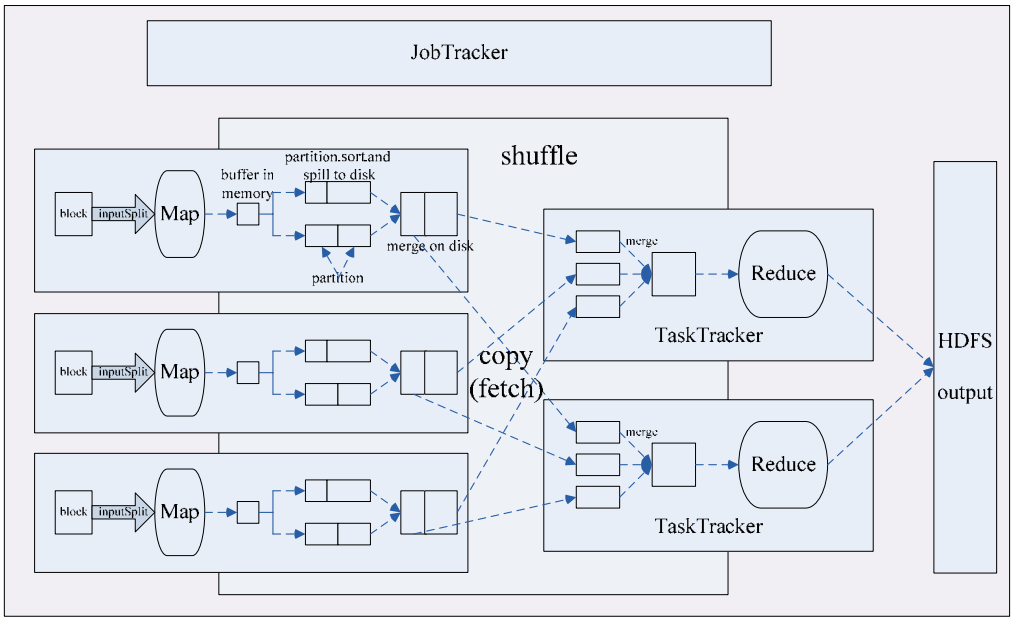

图 2-6 MapReduce 处理过程数据流图示

JobTracker 充当 master 的角色负责调度构成一个 MapReduce 作业的所有任务，这些任务分布在不同的 TaskTracker 上，JobTracker 监控这些任务的执行并重新运行执行失败的任务。TaskTracker 主要负责执行 JobTracker 指派的任务，它通常从 HDFS 中获取数据，而这些数据大多数情况下就在本地文件系统中存有副本可以进行本地计算。这种方式使得计算可以发生在存储数据的节点上而不至于使集群中的带宽成为性能瓶颈。通过 Map 任务处理，生成中间结果，这些中间结果通过 Reduce 任务进行收集合并后输出最终结果。

MapReduce 的 runtime 系统会处理输入数据的分布细节，在集群中调度任务

执行，处理和恢复失效的任务，管理集群中节点的通讯请求等等。这些工作对程序员来说都是透明的。开发人员在一个集群上采用 MapReduce 处理海量数据所需要做的主要工作就是编写 Map 和 Reduce 函数。

2.3.3 MapReduce 一些重要概念

Hadoop 为采用 java 语言编写的 MapReduce 程序的用户提供了 API，在编写 MapReduce 程序时利用这些 API 来处理数据[18]。新版的 API 中与 Map 和 Reduce 过程相对应的抽象类分别是 Mapper 和 Reducer。数据经由 Mapper 处理后的中间结果的通过 Partitioner 进行分区，为了避免 Mapper 与 Reducer 之间的数据传输，还允许用户指定一个本地合并函数 Combiner 来完成本地合并。

Mapper：如果一个类要作为 Mapper，需要继承自 org.apache.hadoop.mapreduce 包下抽象类 Mapper，它的泛型形式为 Mapper<k1,v1,k2,v2>，然后实现这个抽象类的抽象方法 map(k1 key,v1 value,Context context)，这个函数处理一个给定的键值对(k1,v1)，通过自定义的数据处理后生成键值对(k2,v2)，采用 context.write()方法将此键值对输出。

Reducer：如果一个类要作为 Reducer，需要继承自 org.apache.hadoop.mapreduce 包下的抽象类 Reducer，它的泛型形式为 Reducer<k2,v2,k3,v3>，然后实现这个抽象类的抽象方法 reduce(k2 key,Iterable<v2> values,Context context)，这个函数接受 Mapper 函数的输出，并将 k2 相同的值聚合到一个可迭代的容器中传给 reduce 函数，然后执行自定义的逻辑后生成键值对(k3,v3)，同样采用 context.write()方法将键值对输出。

Partitioner：当 MapReduce 作业有多个 Reducer 时，需要采用一个办法来确定 Mapper 应该把键值对输出给哪个 Reducer，默认的是采用 HashPartitioner 来确定 Reducer。在自定义的输出 key 的情况下，需要定制 Partitioner 以便能按照 key 发给正确的 Reducer。

Combiner：在大多数情况下，Mapper 的输出结果在分发给 Reducer 之前，可以考虑本地合并，本地合并的输出将作为 reduce 函数的输入。本地合并会降低 Map 任务与 Reduce 任务之间传输的数据量，并使 shuttle 洗牌更加高效，从而显著提升 MapReduce 的性能。

Mapper 阶段的数据输出，分区，本地合并已经 Reducer 获取 Mapper 的输出结果并进行排序的过程成为洗牌（Shuttle）[19]。

2.3.4 MapReduce 任务的执行过程剖析

用户在提交一个 MapReduce 作业后，整个作业的执行过程如图 2-7 所示，执行过程如下[20]：

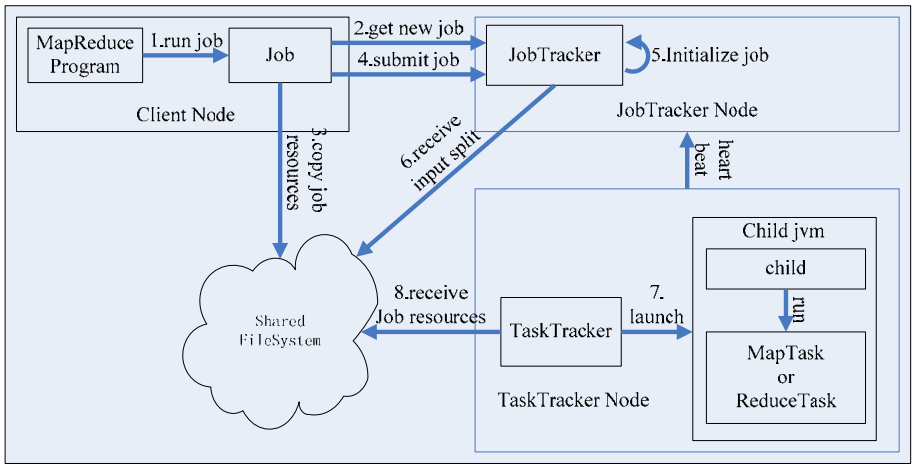

图 2-7 MapReduce 作业提交图示

（1）客户端提交一个 MapReduce 作业：新建一个 Job 实例，设置 Job 实例的各种属性，包括输入、输出在 HDFS 中的路径，用户编写的 Mapper、Reducer 类全限定名，输出的 key、value 的格式。Job 将通过交互 JobTracker 获得作业 ID，拷贝作业资源到共享文件系统，并提交作业。调用 waitForCompletion()启动作业并等待作业完成。见图中的 1，2，3，4 步骤。

（2）作业控制类将对作业的输入进行划分，将作业运行需要的资源复制到 JobTracker 所在的文件系统中，并初始化作业。见图中的 5，6 步骤。

（3）创建运行任务列表，JobTracker 通过获取输入划分信息，为每一个分片创建一个 Map 任务，同时根据程序设置的 Reduce 任务数创建 Reduce 任务。JobTracker 为各个 TaskTracker 节点分配 Map 和 Reduce 任务。理想情况下，Map 任务的输入数据是本地化的。

（4）TaskTracker 执行 JobTracker 指定的 Map 和 Reduce 任务。在任务的运行期间，任务定期向 TaskTracker 反馈任务的执行进度，同时 TaskTracker 以心跳方式定期和 JobTracker 报告节点的 MapReduce 任务执行的情况。

（5）JobTracker 等待所有任务执行成功后，将作业状态标志为成功并清空作业的工作状态信息，同时通知 TaskTracker 节点清空作业状态信息。

整个任务在执行的过程中，可能会有各种各样的任务失败产生[21]。如 MapReduce 任务失败，任务所在 java 虚拟机意外退出，TaskTracker 失败或与

JobTracker 通讯失败甚至 JobTracker 失败等情况。对于 MapReduce 失败，TaskTracker 任务失败的情况，JobTracker 会重新调度任务的执行，使得系统从故障中恢复，但是 JobTracker 发生失败将是无法恢复的。

2.4 本章小结

本章主要介绍了 Hadoop 的基本理论，包括 Hadoop 的概念、发展历史，功能与特点等和 Hadoop 的核心：HDFS 分布式文件系统和 MapReduce 计算模型。详细分析了 HDFS 解决海量数据存储的原理与读写过程，MapReduce 在 HDFS 上并行数理数据的原理与过程，从中可以看出 Hadoop 在处理海量数据问题上的优势。访问日志数据的可分割与可并行处理的性质使得采用 Hadoop 处理是合适的。将访问日志数据使用 Hadoop 进行预处理，充分利用集群的计算资源，显著提高处理数据的速度，为快速获取分析结果提供了保证。

第 3 章 电子商务网站访问日志处理与分析

3.1 访问日志概述

访问日志是在用户与服务器交互的过程中产生，并由服务器对网站的访问进行记录而生成的文件[25]。从广义上来说包括 Web 服务器，代理服务器和专用日志服务器等生成的日志文件，这些日志文件记录用户访问网站的详细信息，日志数据反应了用户对网站的访问行为，是网站分析的主要数据来源。

访问日志的特点如下：

（1）日志数据量巨大。在访问量较大网站中，日志每天可能会以数十 GB 的速度增长。

（2）产生日志的服务器采用集群的情况下日志会分散到多台服务器上。

（3）日志准确地记录了用户访问服务器的数据流，这些数据能密切反应用户与服务器的交互过程。

（4）Web 服务器生成的 Web 日志中会包含大量的噪声，采用 JavaScript 标记方式生成的日志噪声要少的多。

3.1.1 访问日志的格式

访问日志可以用不同的格式存储，了解日志的格式将有利于更好地进行数据收集、处理和分析进而对网站进行优化[26]。目前常见的数据格式有两种，通用日志格式 CLF(common log format)和扩展日志格式 ECLF(extended common log format)，一个最常见的基于 NCSA 扩展的 Apache Web 日志 (ECLF)示例如下：

202.114.93.195 -- [27/Aug/2012:14:59:59 +0800]

"GET /detail.example.com/item.htm?id=123456789 HTTP/1.1" 200 1234

"http://www.google.com.hk/search?q=hregion"

"Mozilla/5.0 (Windows NT 6.1; WOW64; rv:11.0) Gecko/20100101 Firefox/11.0"

"202.114.93.195.1312275036153.0 202.114.93.195.90720536175243.0 "

这个日志可以解读为：一个 IP 为 202.114.93.195 的用户在 27/Aug/2012:14:59:59 +0800 使用 Mozilla Firefox/11.0 浏览器通过搜索引擎 Google 查找 hregion 并在搜索结果页中点击进入域名为 detail.example.com 的网

站，访问了/item.htm?id=123456789 页面并获得成功，访问协议为 HTTP/1.1 协议，访问方式 GET，得到 1234 字节数据。其中还包括一些用来记录客户端信息的 Cookie 数据，这些 Cookie 数据将在网站分析过程中发挥重要的作用。

近年来，网站分析的过程中所使用的传统 Web 日志的数据逐渐被采用 JavaScript 标记方式生成的访问日志数据取代[27]。采用 JavaScript 标记方式生成在日志与 Web 服务器生成的日志非常相似，但由 JavaScript 代码收集的客户端 Cookie 数据通常以参数的方式拼接在访问日志服务器的 URL 中，如下是 Google Analytics 的 JavaScript 代码在生成日志时，访问 URL 参数中拼接的字符串示例。

utmac=UA-29465542-1;utmcc=__utma%3D191089395.1565089535.1330129486.1330302791.1330326659.14%3B%2B__utmz%3D191089395.1330129486.1.1.utmcsr%3D(direct)%7Cutmccn%3D(direct)%7Cutmcmd%3D(none)%3B;utmhn=www.hregion.com;utmp=%2Fota%2Fuser%2FtoWelcome.dox;utmdt=%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F;utmr=0;utmn=725889875;utmsr=1600x900;utmcs=UTF-8;utmsc=32-bit;utmul=zh-cn; utmwv=5.2.5

这些数据包含了大量的客户端信息，包括用户标识信息，Session 标识信息，Refer 页面，客户端操作系统与浏览器的一些参数等等。

3.1.2 日志处理与分析过程中的相关术语

页面浏览(Page View)：页面浏览是评价站点常用的指标，简称 PV。PV 中的 page 通常指来自浏览器的一次 html 网页请求，除此之外网站内部跳转，AJAX 请求，FLASH 异步请求，文件下载，RSS 订阅操作，弹出式广告以及各种恶意访问通常也被当作一次 PV。将这些 PV 累加成为站点或子站点或者某个应用的总的 PV。监测网站 PV 的变化趋势和分析其变化的原因是一项非常重要的工作。

独立访客(Unique Visitor)：简称 UV，是指可以唯一确定的网站访问者，通常用来统计在一段时间内有多少唯一身份的人来过网站。从理论上讲，访问者的用户帐号识别是非常准确的方式，但浏览绝大多数网站时通常并不需要用户登录使得帐号识别难以应用。常见的 UV 识别方法有两种：根据 JavaScript 读取 Cookie 判断和根据 IP+UserAgent 判断。

访问/会话(session)：指访问者与服务器之间的一次交互过程。当一个用户来到网站，在一段时间内连续点击并浏览了一些页面后离开网站，这样一次交互过程称为会话。会话通常会有超时设定，常见的会话超时时间为 30 分钟，如果连续两个请求之间的时间超过 30 分钟，这两个请求会被认为是两次不同的访问。为了便于统计，通常可以认为跨天的访问属于不同的 Session，这与服务器端保

存的 Session 并不完全一致。

进入页(Entry Page)，退出页(Exit Page)与登录页(Landing Page)：进入页是指访问网站时所浏览的第一个网页，即每个访问的起始网页。退出页是指结束网站访问前的最后一个网页，退出页常用来确定网站访问的瓶颈。登录页是指用户通过搜索引擎或者广告后进入网站看到的第一个引导页面。由于一次访问期间用户可能会多次进入网站，所以可能有多个登录页。与这些概念紧密结合概念的还有退出率和崩失率。退出率指某个网页作为退出页与这个网页的 PV 的比值。崩失率有两个层面，一是指进入网站后只浏览了当前页面就离开与这个网页 PV 的比值。二是整站的跳出访问数与访问数的比值[28]。

访问来源(Referer)：是指一次访问或者一个网页浏览的流量来源，又称为推荐来源。访问的浏览来源按照来源网站的性质可以分为以下几类：搜索引擎 PPC 与 SEO，门户网站的广告，合作站点的反向链接以及直接来源。按照定义的方式流量来源分为页面的流量来源、访问的流量来源与访问者的流量来源。

初访者(New Visitor)，回访者(Return Visitor)和重复访问者(Repeat Visitor)：初访者指在统计时间内首次浏览网站的访问者。回访者是指在统计期间之前浏览过网站，并在统计期内再次访问网站的访问者。重复访问者是指在统计期内多次访问网站的访问者，初访者和回访者都可能是重复访问者。

转化与转化率：转化是指网站访问达成了某个重要的商业目标，有时也指代那些未来能够产生受益的行为。商业目标的不同会使得网站转化的定义不同。如访问者访问某个目标网页，或是成为注册用户，或是浏览广告抑或是电子商务网站能促成购买行为等等。转化数与总数的比值为转化率，如通过广告进入网站的转化数与广告进入网站的总数的比值就是广告的转化率。

订单平均金额与订单平均成本：平均订单金额是由总销售额/订单数计算得出，为平均每笔订单的金额数，这个指标用来衡量电子商务网站的质量。高订单数可能是高转化的结果，但电子商务网站和成交额度哥紧密联系要求网站在高转化数的情况下，更有高质量的转化。订单平均成本值获得一个订单需要支付的平均营销费用。它能反应网站的营销投资情况，直接影响着 ROI，减小营销开支会导致网站流浪的降低，订单数减少。所以增加订单数是可以使得订单的平均成本下降。

购物车投入率与支付完成率：购物车投入率用来衡量访问者在访问网站的过程中将商品加入购物车的比率。这个指标和支付完成率通常一起衡量网站的健康度。只有支付完成，才意味着商品销售成功。

3.1.3 访问日志分析与网站分析

访问日志分析着重于从技术层面的来解读数据并为网站分析提供基础数据，网站分析偏重于从业务层面来关注整个网站的表现。可以通过日志分析工具来分析用户对网站的浏览情况，但在实际应用中应更多的是采用一些第三方的网站分析工具来监控访客对网站的访问情况[29]。

近几年行业内最大的变化是免费网站分析工具的出现，并从付费分析工具那里获取了部分市场份额，Google Analytics 就是其中的典型代表。这些网站分析工具无论免费与收费能够提供包含从日志采集到预处理直到以报告的形式给出网站各项关注指标与纬度可视化数据报告的解决方案。以下是几个常见的网站分析工具及特点：

（1）Google Analytics(GA)：GA 是一款功能强大并且免费的网站分析工具，提供了可以和收费工具相媲美的功能，并且功能相当完备。使用也非常简便，只要将 GA 的分析脚本加入到网页中就完成部署了，在更高级的配置方面 Google 提供了部分 API 来根据具体的站点进行调整[34]。

GA 的 JavaScript 跟踪代码将访客信息存储到 Cookie，然后将信息发送到 GA 的数据服务器上进行数据处理，这些信息包括唯一访客标识，Session 信息，网页标题，Refer，Google 的数据服务器记录的同时也记录了访客 IP，访问时间，以及浏览器类型、操作系统、系统语言、屏幕分辨率等等，处理后 GA 会根据网站的分析配置文件提供报告。网站的分析配置文件能够提供非常详细的过滤功能。

（2）百度统计是百度 2009 年推出的一款专业网站分析工具，提供了丰富的数据指标，功能强大但操作简易。通过百度统计的分析数据为网站建设与运营提供决策依据，如改善用户体验，提高流量，提升转化与投资回报率[35]。

百度统计提供了几十种图形化报告，全程跟踪用户的行为路径，可以监控各种网络媒介推广效果以便及时了解哪些关键词与哪些创意的效果较好好。同时，百度统计集成百度推广数据，可以及时了解百度推广效果并优化推广方案。

（3）WebTrends 是网站分析行业的开创者，其强大的分析能力，全面、精准以及可定制的数据分析报告让 WebTrends 吸引了大量的电信运营商和金融企业。WebTrends 优于其它分析工具的一面包括支持 PageTag 方式的数据收集。然而 WebTrends 需要网站支付高额的授权费，这让不少企业望而却步，也因此 WebTrends 的市场占有率甚至不如 GA[36]。

3.2 日志处理与分析的步骤

日志处理与分析的过程通常分为四个阶段 :数据采集阶段，数据预处理阶段，分析算法实施阶段，数据可视化阶段。数据预处理和日志挖掘分析算法是日志处理与分析过程中的关键技术。

3.2.1 日志的采集

日志的采集是进行分析的前提。目前应用比较广泛的数据采集方法分为以下两种：Web 服务器采集与 JavaScript 标记[37]。

Web 服务器端日志采集：当用户发起访问请求时，Web 服务器通过 HTTP 提供服务的同时，将 HTTP 中的一些信息写入访问日志文件。通过配置 Web 服务器就可以获取非常详细日志信息。

利用 Web 服务器记录用户访问网站的日志数据时需要处理一些问题，如代理服务器的缓存使得一部分 HTTP 访问请求不会到达 Web 服务器，访问日志中包含对 js、css 以及音频视频的访问请求，搜索引擎爬虫和其它自动代理对 Web 服务器的请求会生成大量的日志，部分请求出错的日志也会被记录。

JavaScript 标记采集：这种方式的特点是它独立于 Web 服务器，把访问日志数据从 Web 服务中地分离出来，使用专门的日志服务器来收集与存储访问日志数据。这种方式的好处在于网站开发人员专注于提供页面，而分析人员专注于对采集日志数据的分析，双方在各自的工作中有极大的灵活性。访问日志采集原理如图 3-1 所示：

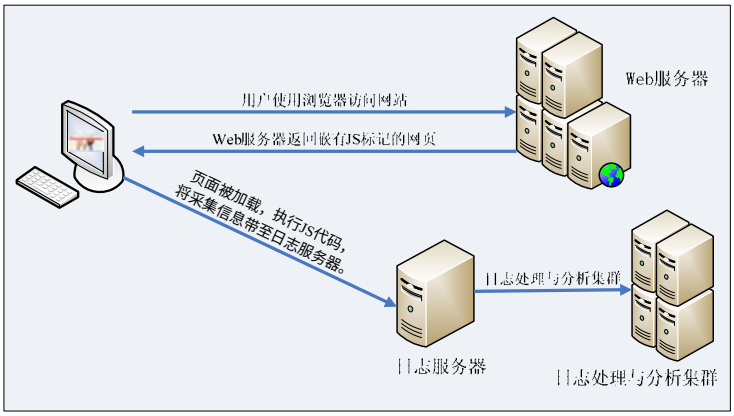

图 3-1 JavaScript 标记采集日志原理

采用此方式进行访问日志数据的收集时只需要在业务关注的网页中签入一

个标记即可，对于采用模板方式的网页来说是非常容易实现。

JavaScript 标记方式的数据收集处在用户行为源头的客户端，能够更加准确的捕捉用户行为、更加容易解决缓存和会话识别等问题，所以它比服务器端收集更具有优势，但是它的收集需要用户许可并可能涉及用户隐私。同样的优势还包括搜索引擎的爬虫程序不会执行 JavaScript，不会生成访问日志因而无需对爬虫日志进行数据清洗。目前在客户端部署 JavaScript 标记的方式更为常见。在使用 JavaScript 标记方式收集数据时要求客户端浏览器中没有禁止 Cookie。当网页载入过慢，访问者后续点击过快时，会丢失网页的浏览记录。

3.2.2 日志的预处理流程

需要对原始日志进行多次预处理后才适合做数据分析、挖掘等工作，这使得日志的预处理工作在整个日志处理过程中极为重要。

数据的预处理可以改进数据质量，加快数据挖掘与分析的进度，改进分析系统的性能等。整个预处理过程占数据挖掘与分析过程超过一半以上的工作量。数据预处理的步骤包括：数据合并与清洗，用户识别，会话识别，事务识别。如下图 3-2 所示：

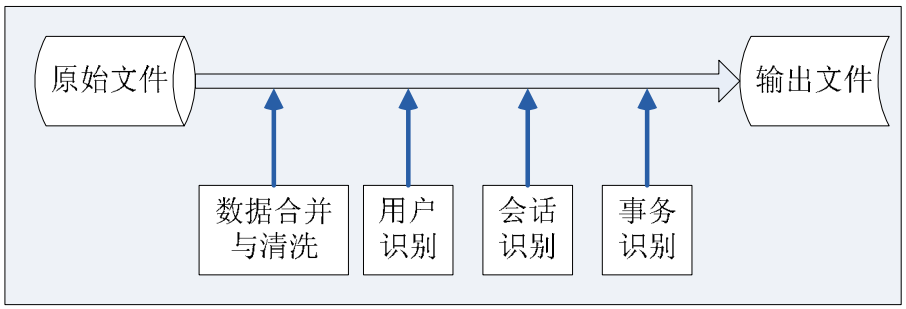

图 3-2 数据预处理的步骤

数据合并与清洗是对原始的日志文件进行处理，将集群中的多个日志文件合并成一致的数据存储，同时删除与后续的日志挖掘与分析过程中无关的日志记录的过程。这其中包括清洗非显式请求的样式文件，图片，JavaScript 脚本，视频，音频等文件的日志记录等等。数据清洗过程需要根据挖掘与分析的目的判断数据是否为脏数据。不同的关注角度会有不同的脏数据界定范围。例如一个以图片、视频或音频为主要业务的网站，需要根据实际情况选择性清洗部分图片、视频与音频数据访问记录。搜索引擎和其它自动代理对网站的爬取行为生成的日志记录由于不能反应用户交互行为，通常也需要被清洗掉。其它如错误的请求,公司内部开发测试阶段的内部访问日志等也要根据实际情况进行清洗。

用户识别对于进行聚类分析和访问模式的挖掘是非常关键的。用户识别就是识别有哪些不同的用户访问了网站。在具体的网站分析过程中，由于多个用户可能会使用同一计算机访问同一站点，多个用户也可能会通过同一代理服务器来访问同一站点。同一用户在多台计算机上访问同一站点，同一用户采用不同的操作系统和浏览器访问同一站点，同一用户在接入网络时由网络提供商动态提供 IP 地址等等。采用 JavaScript 标记方式可以在 Cookie 中的唯一标识来区分访问者，当用户禁用 Cookie 或者采用不同的浏览器时统计结果偏差较大。采用 IP+UserAgent 的方式对于同一计算机上不同用户的访问难以识别，对于一个公司内部采用统一 IP 出口和操作系统版本的情况来说 UV 的偏差也较大，但随着计算机软硬件方面条件的改善，这种识别处理误差并不明显。无论采用哪种方式，在实际工作中要尽量保持识别方式在较长时间内稳定，因为变化的趋势更加重要。

会话识别是指在较长一段时间内同一用户与网站的多次会话的所产生的日志依据某种规则进行区分。如果日志时间跨度较大，日志文件会包含用户会在此时间段内多次访问网站的记录，会话识别就是把属于同一用户日志记录识别到不同的会话中去。目前会话识别的主要方式包括整站停留时间上限的阀值算法，单个页面停留时间的阀值算法，以及是否通过引用页访问站点来识别用户会话。在采用 JavaScript 标记方式的情况下可以采用设置 Cookie 时效的方式。

事务识别是在会话识别后的基础上，对用户会话中的页面序列进行语义分组，寻找此次会话中有意义的访问路径。事务的大小取决于对事务所定义的粒度，通常一次会话过程会有若干事务首尾连接而成。事务识别对于关联规则，频繁路径的分析尤为重要。它采用的方法有时间窗口法、最大向前引用路径法、引用长度法、基于兴趣度的事务识别算法。

3.2.3 日志模式识别的主要方式

将数据挖掘技术应用于日志挖掘分析可以发现潜在用户，识别用户访问模式、发现频繁访问的路径和页面和挖掘用户的兴趣关联规则等，从而帮助人们理解、分析和预测用户行为，改进站点设计和服务并提供个性化信息服务决策支持。

日志经过预处理阶段处理以后就可以对具体的分析要求选择适当的算法进行数据分析。由于各个站点分析目标的不同，具体采用的技术也不尽相同。常

见的模式分析算法包括路径分析、分类与聚类，关联规则等等。

路径分析是一种常用的模式分析方法，它可以用来发现 Web 站点中最频繁的访问路径，以及业务关注点对应的点击流路径，根据这些分析结果可以调整站点结构，改进页面设计进而提高站点的转化。

聚类是指按照对象的关注属性将多个聚集对象分为若干类，类并非事先确定，而是通过聚类算法自动获得。聚类后同一类中对象之间具有较高的相似度而类之间的相似度较低。进行聚类分析时，需要把聚类依据的属性抽象出来，然后通过计算对象之间的距离进行聚类。常见的聚类分析算法包括划分方法，层次方法，基于密度的方法，基于网格的方法和基于模型的方法。K-Means 算法就是一种划分方法算法实现。

聚类作为一种有效的模式方法较为成功的实现了对低维数据的处理。然而随着信息技术的迅猛发展，需要处理的数据无论从维数还是数量上都到达了前所未有的高度，传统的基于距离的聚类方法已经无法来构建簇。分析其失效的原因，首先是存在大量的无关数据，其次是高维数据普遍存在分布稀疏，彼此之前的距离几乎相等。如何实现高维数据的聚类，已经成为聚类技术的重要研究方向和需要攻克的技术难题。

日志挖掘与分析的过程中的聚类主要包含用户聚类，Web 页面聚类与搜索结果聚类等。用户聚类将相似浏览行为的用户归类来提供个性化推荐。Web 页聚类有助于用户快速理解网页中的主要内容。通过搜索结果聚类可以使得网站分析人员更好的理解用户意图，提升站内搜索的准确率。

关联规则指运用数据挖掘技术分析给定的项目和记录集，发现项集中项目关联的程度。目的是发现数据间的隐藏联系。

在关联规则算法中，把项目的集合称为项集，包含 k 个项目的项集称为 k 项集，包含项集的事务数称为项集的出现频率或支持度。如果项集的出现频率大于或等于最小支持度与事务总数的乘积，则称该项集为频繁项集。用户设置最小置信度和最小支持度，项目之间的相关性，只要这两个值大于用户设置的值，那么就可以认为项集之间具有较高的相似度。

关联规则挖掘过程分为下面两步：首先找出所有的频繁项集，即找出支持度大于或等于给定的最小支持度阈值的所有项集。然后由频繁项集产生强关联规则，即找出满足最小支持度和最小置信度的关联规则。

序列模式分析是指给定一个由不同序列组成的集合，其中每个序列由不同的元素按顺序有序排列，同时给定一个用户指定的最小支持度阈值找出所有的频