

中国科学技术大学

硕士学位论文

云计算技术在web日志挖掘中的应用研究

姓名：程苗

申请学位级别：硕士

专业：商务智能

指导教师：陈华平

2011-05-15

摘 要

如何解决数据挖掘中海量数据处理的问题一直是数据挖掘领域一个非常重要的研究课题。尤其是随着网络技术的迅猛发展,web 上的数据正以指数级形式飞速增长,且 web 上的数据具有海量、多样、异构、动态变化等特点,这使得基于单一节点的集中式数据挖掘平台已经不能满足目前海量数据网络的分析任务了。如何实现快速地从 web 这个最大的数据集合中提取出有用的信息已成为数据挖掘领域一个备受国内外学者关注的课题。

云计算正是产生于这样的背景之下,它的出现给海量数据的处理和存储带来了曙光。同时,云计算只需要部署在普通的廉价计算机集群上即可运行,但是数据处理能力却很强,因此 Web 数据挖掘系统在云计算集群框架下的成功实行具有很重要的意义和应用价值。

本文在 Hadoop 平台上,结合 web 日志挖掘的特点,给出了一种基于云计算的 web 日志挖掘系统的设计方案,并对该系统的各个功能模块进行了详细的阐述。同时,针对目前从 web 日志中挖掘用户偏爱路径的算法注重客观访问频度,而忽略用户对这一频繁访问路径是否感兴趣的不足,结合网站拓扑结构图修正基于频度的用户偏爱路径的衡量标准,提出了有用偏爱度的概念,给出了一个挖掘用户浏览偏爱路径的方法,从而剔除了由于页面放置和链接等因素对挖掘的影响。

最后对本文给出的改进算法的有效性以及云计算平台的高效性进行了实验比较分析。实验结果表明,改进后的挖掘用户偏爱浏览路径的算法更能反映用户的浏览意图。同时,利用云计算平台,通过“云”中多个资源完成原先由一个节点承担的工作,无论是在数据处理还是任务执行上,其效率都高于基于单机集中式环境的 web 日志挖掘。

关键词: 云计算 web 日志挖掘 Hadoop 浏览偏爱路径

ABSTRACT

How to solve the problem of processing massive data in data-mining filed is always an important researching subject. Especially with the rapid development of network technology, the data on the web increase rapidly in the form of exponential and with many characteristics such as massive, diverse, heterogeneous and dynamic, this makes mining on a single node can not meet the need of current massive data analysis task. How to extract useful information from the world's largest data collection—web, has become a more concerned subject for scholars from all over the world.

Cloud Computing is produced under the background of the situation mentioned above, its emergence gives a bright future for massive data processing and storage. The platform of Cloud Computing can run only to be deployed in an ordinary cluster of inexpensive computers, but the data processing capability is strong. Therefore, whether web data mining system run successful under the framework of Cloud's cluster or not, has an important significance and application value.

Based on the Hadoop platform, combined with the characteristic of web log mining, we present a solution of web log mining system which based on Cloud Computing, and describe each module of the system in details. Meanwhile, the current mining algorithms are focus on users' browsing frequency, neglect an important problem of whether users are interested in the frequent path or not. Due to this problem, combined with web topology structure, revise the measures of users' preferred browsing paths which based on browsing frequency, and present a concept of useful preference and a method of mining user preferred browsing path, remove the bad impact of mining due to pages' place and links.

Finally, we make experiments to verify the effectiveness of the improved algorithm and the efficient of Cloud Computing. The result shows, the improved algorithm can dig out preferred browsing path which reflecting the users' preference more accurately. Meanwhile, according to using rich resource in Cloud to accomplish mining task can reach to a higher efficiency than which in a single node environment, both in data processing and task execution.

Key Words: Cloud Computing, Web log mining, Hadoop, Preferred browsing path

中国科学技术大学学位论文原创性声明

本人声明所呈交的学位论文,是本人在导师指导下进行研究工作所取得的成果。除已特别加以标注和致谢的地方外,论文中不包含任何他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的贡献均已在论文中作了明确的说明。

作者签名: _____

签字日期: _____

中国科学技术大学学位论文授权使用声明

作为申请学位的条件之一,学位论文著作权拥有者授权中国科学技术大学拥有学位论文的部分使用权,即:学校有权按有关规定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,可以将学位论文编入《中国学位论文全文数据库》等有关数据库进行检索,可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。本人提交的电子文档的内容和纸质论文的内容相一致。

保密的学位论文在解密后也遵守此规定。

公开 保密(____年)

作者签名: _____

导师签名: _____

签字日期: _____

签字日期: _____

第 1 章 绪论

近年来，随着 Internet 技术的快速普及和迅猛发展，使得 web 成为世界上最大的数据源。如何在这个数据集中发现有价值的信息，无疑将成为数据挖掘研究的热点。然而传统基于单一节点的集中式 web 挖掘系统的计算及存储能力已经达到了一个上限，利用云计算技术获取网络中强大的计算资源，将消耗大量计算资源的复杂计算通过网络分布到多节点进行计算，是当前一种行之有效的解决方案。

本章首先对 web 数据挖掘问题以及课题的研究目的和意义做了一个概述，然后简单地介绍了国内外对于提高数据挖掘系统处理能力问题的研究状况，最后简述了本文的研究内容和组织结构。

1.1 引言

1.1.1 Web 挖掘及其研究意义

近年来，随着电子商务的普及，商业网站面临的一个严峻问题是：如何保留老客户吸引新客户以提高效益。基于电子商务的业务交易比传统线下的业务交易更加激烈，因为客户只需点击手中的鼠标，就可能从一个网站转到其竞争对手那边的网站。如何安排网站的内容、结构以及分析、跟踪、获取用户的个性特征，是基于电子商务的业务交易企业在竞争中获得成功的关键。因此，任何与消费者及其行为有关的信息对企业来说都是非常宝贵的，但是这些数据资源未能充分的挖掘和利用。开发一个通用平台来有效地从“数据丰富，信息贫乏”的 web 上挖掘出对企业有价值的信息，是互联网企业发展的必然选择。

基于这样的背景，Web 数据挖掘作为一门集 web、数据挖掘、数据库技术、自然语言理解、人工智能等多学科于一体的交叉学科应运而生。Web 数据挖掘是利用数据挖掘方法，从 web 的超级链接结构、网页内容和使用日志中，提取隐藏的、有用的模式和知识。

其中 web 日志挖掘是 web 挖掘的一个研究方向，它是从大量访问者的访问记录（即日志文件）中，挖掘网站的频繁使用模式、用户访问模式、具有相似行为的用户群、用户偏爱访问路径等信息，使企业能够充分了解自己 web 站点的使用情况和使用 web 站点的用户的行为模式，从而更好地优化站点的组织结构，更好地为用户提供服务，提高站点的访问量和性能。

1.1.2 论文研究目的与意义

传统的 web 数据挖掘系统主要是在单机上运行，它在 Internet 这个数据分布存储的环境下，要求首先把数据重新收集集中到一个数据仓库，这需要高速的通信网络，并将牺牲大量的带宽来移动大量的数据。同时，随着网络用户的不断增多以及电子商务的蓬勃发展，互联网迅速发展成为世界上规模最大的公共数据源，其上的数据具有如下特征：

- (1) 数据量大，呈指数级不断增长，并且分布在整个 Internet 上。
- (2) 数据类型多样，如结构化的表格、半结构化的网页、无结构化的文本以及图片、视频等多媒体文件。
- (3) 由于制作网页者的个体差异，使得表示相同或相似内容的网页可能会使用完全不同的文字和格式，多个网页的信息整合将变得困难。
- (4) 互联网上的信息包含噪音，要进行细粒度的 web 信息分析和数据挖掘，这些噪音必须去除。
- (5) 互联网具有动态性，其上的信息不断发生变化，如何紧跟并监测这些变化对许多网上应用而言是非常重要的。

Web 上数据的所具有的这些特点给传统的 web 数据挖掘系统带来了两大难题：(1)Web 上的数据量非常巨大，并且随着网络技术的发展在不断增长，传统数据挖掘系统的数据存储系统——数据仓库已经不能满足日益增长的存储需求；(2)Web 上数据类型多种多样，如结构化的表格、半结构化的网页等，且包含了大量的噪音，处理这些数据的复杂度提高，传统基于单一节点的集中式数据挖掘系统提供的有限计算资源的计算能力很难满足用户需求。传统的 web 数据挖掘系统因资源限制而不能满足存储及计算方面的需求，因此需要借助分布式计算技术来处理 web 数据挖掘这个涉及海量数据计算及存储的复杂问题。

云计算是基于互联网的新一代计算方式，它是网格计算、并行计算、分布式计算、效用计算、网络存储、虚拟化、负载均衡等传统计算机技术和网络技术发展融合的产物。在云计算中软、硬件资源以分布式共享的形式存在，可以被动态的扩展和配置，最终以服务的形式提供给用户。用户按需使用云中的资源，不需要管理，只需按实际使用量付费。在采用云计算技术以后，基于互联网的服务在计算及存储能力上得到大幅度提高，且可靠性也得到大幅度提升，计算资源的共享程度和使用效率大幅提高，采购成本和维护成本大幅度降低，在绿色和节能方面也获得了很大的进展。这些特征使存储及计算能够实现商业化，无论从需求上还是外部特征来看，基于云计算的海量数据存储和挖掘是一个具有理论和应用价值的研究领域。

从需求来讲，互联网上的数据快速增长，数据挖掘任务日益复杂，如果还采用高性能机来处理海量数据，在成本上人们是无法承受的，因此在挖掘过程中就需要有很好的开发环境 and 应用环境。而云计算只需部署在大量廉价的计算机上，可以大大节约实施分布式系统的成本，这种云计算的方式用于海量数据挖掘是比较合适的；从外部特征来看，由于云计算是部署在大量廉价计算机上的，因此可以大大降低企业的数据处理成本。同时基于云计算的数据挖掘平台的扩展性和容错性都比较高。

总之，用云计算的方式来挖掘海量数据的优势有：

(1) 互联网上数据的快速增长，使得数据挖掘任务远比搜索任务复杂得多，导致数据挖掘需要很好的开发环境 and 应用环境。在这样的背景下，基于云计算的方式是比较合适的。

(2) 云计算可以为企业提供低成本分布式并行计算的海量数据处理方案，使得企业的数据处理成本大大降低，同时也不再依赖于高性能计算机。

(3) 基于云计算的数据挖掘系统屏蔽了底层，在并行环境中，能利用原有的设备提高大规模数据的处理能力和速度。

1.2 国内外研究现状

1.2.1 对 web 日志挖掘的研究

目前，国内外基于 web 日志挖掘的研究工作大致有 3 类(周斌等, 1999)。

(1) 以分析 web 站点性能为目标：主要从统计学的角度，对日志数据项进行简单的统计，得到用户频繁访问页、单位时间访问数、访问数量随时间分布图等。绝大多数商用及免费的 web 日志分析工具均属此类。

(2) 以理解用户意图为目标：主要是通过算法从 web 日志文件中找出频繁用户浏览路径。Chen MS 等提出的路径遍历模式的发现算法，以及 Zaiane 等使用的数据立方体方法，便是此类的典型代表。

(3) 以改善 web 站点设计为目标：通过挖掘用户的频繁访问路径和用户聚类，重构站点的页面之间的链接关系，以更适应用户的访问习惯，同时为用户提供个性化的信息服务。

Chen MS 等人首先将数据挖掘技术应用于 web 服务器日志挖掘，发现用户的浏览模式。提出最大向前引用系列的概念。将用户会话分割成一系列的事务，然后采用与关联规则相似的方法挖掘频繁浏览路径。

目前也有一些 web 日志分析工具，但这些工具都只是对日志进行简单的统

计,如页面访问频率、访问时间等,很少分析数据间隐含的关系,不能发现用户的访问模式。

1.2.2 对分布式并行挖掘系统的研究

目前,已有大量国内外学者为了提高数据挖掘系统的处理能力,提出了各种分布式并行数据挖掘系统。这些系统有基于集群的、有基于 Agent 的(刘业政, 2004)、基于 CORBA 的(唐卫宁, 2002),但实现却相对复杂且只能针对特殊应用。之后, M Cannataro(2001)等人基于 Globus Toolkit 设计了一种分布式并行知识发现平台,该平台利用 Globus Toolkit 所提供的网格计算能力,解决了传统数据挖掘计算能力不足的问题。近几年集中在基于 Globus Toolkit 平台上并行数据挖掘算法的实现与改进方面的研究(郑晶, 2010)(齐玉成等, 2010)。但网格计算一般用于科学计算,为专用领域的问题而设计,缺少商业化实现,且 Globus Toolkit 是基于中间件技术,需要通过编程或安装设置来搭建底层架构,增加了系统实现的难度(纪俊, 2009)。另一方面,部署一个网格环境需要耗费大量服务器等硬件,对中小企业、个人用户来说,很难承受如此大的开销来满足处理能力较高的系统硬件需求。即便是能够购入如此多的硬件设备,在系统不工作或非满负荷工作时,会造成很多资源浪费。而在云计算中,软、硬件资源以分布式共享的形式存在,可以被动态的扩展和配置,最终以服务的形式提供给用户。用户按需使用云中的资源,不需要管理,只需按实际使用量付费,这些特征使得用云计算来搭建 web 数据挖掘系统,不仅可以解决传统数据挖掘系统计算及存储能力不够的问题,同时也解决了基于网格的数据挖掘系统的成本问题。

因此,基于云计算的海量数据处理方案已经出现。2008 年底,中科院计算所开发出中国第一个基于云计算平台的并行数据挖掘平台 PDMiner,用于中国移动 TB 级数据挖掘,它具有如下特点:提供大量并行挖掘算法和 ETL 操作组件;可以实现 TB 级海量数据处理并在此基础上进行并行挖掘分析处理;ETL 操作达到了线性加速比;多个任务可在云环境中的任意节点同时启动;具有很高的容错能力,算法组件也可方便地加载到平台中。目前,他们正在研究面向 web 基于云计算数据挖掘服务系统 WPDMiner。

除此之外,国内也有一些大学学者提出了一些基于云计算的海量数据挖掘平台及挖掘算法:纪俊(2009)用扩展的 Google App Engine 开发平台设计与实现了一数据挖掘系统,通过 ID3 决策树与 K-means 算法与相关实验验证了基于云计算的数据挖掘的高效性;李雪峰(2010)主要是针对 web 数据挖掘中 Graph 的算法在云计算环境中进行研究,介绍了基于 MapReduce 的分布式 Graph 直径求解

算法，并利用 Hadoop 对该算法进行了仿真实验；高勋(2010)在云计算环境下研究 pagerank 算法，对基于 MapReduce 的一般并行 pagerank 算法、分块并行 pagerank 算法和底迭代 pagerank 算法进行测试和比较。

1.2.3 基于云计算的应用研究

云计算的应用领域非常广，包括科学应用和商业应用。在生物领域，主要通过 MapReduce 这种通用的并行技术来实现海量数据管理和大规模计算的能力，如马里兰大学的 Michael C(2009)使用开源的 MapReduce 搭建了一个在多个计算节点上并行执行的 DNA 信息分析平台——CloudBurst；在高校教育平台搭建领域，华盛顿大学的 Justin Cappos(2009)等人搭建了一个由分布在世界各地的 1000 个计算机组成的自由、教育研究平台——Seattle；在地理信息领域，Qichang Chen 等提出了高性能工作流系统——MRGIS，这是一个高效率执行 GIS 应用程序的基于 MapReduce 集群的并行和分布式计算平台；在科学计算领域威特沃特斯兰德大学的 Scott Hazelhurst 使用亚马逊的弹性计算云(EC2)进行了虚拟高性能计算的研究；在商业应用方面，IBM 开发了一个企业文本分析平台，应用 Hadoop MapReduce 框架来支持分类工作。

总体来看，如何基于 MapReduce 框架开发面向大规模数据和计算的应用，是未来一个十分重要的研究方向。

1.3 研究内容与论文框架

本文在分析了传统集中式 web 日志挖掘系统不足的基础上，利用云计算技术的优势，设计了一个基于云计算的分布式并行 web 日志挖掘系统。借助 MapReduce 的编程思想，将 web 日志挖掘任务在两个方面进行分布式并行处理：一是将海量数据集划分成若干子块，每个子块分别发送到网络中不同的处理机上并行处理；一种是将传统的挖掘算法改进使其能够在分布式网络环境中并行处理，从而减少单台处理机的工作负荷。本文还针对目前基于 web 日志挖掘用户浏览偏爱路径的算法，注重客观访问频度而忽略了用户对这以频繁访问路径是否感兴趣的问题，提出了有用偏爱度的概念，并结合 web 拓扑结构图修正基于频度的用户浏览偏爱路径的衡量标准，剔除了由于页面放置和链接等因素对挖掘的影响。最后对改进的算法做并行处理，在 Hadoop 平台上设计实验，验证了改进算法相对于传统算法性能的提高，证明了云计算带来的强大计算能力。

第一章为绪论，对课题的研究背景和研究目标进行了简要的说明，同时对

web 日志挖掘以及分布式并行挖掘的国内外研究现状做了一个概述,最后介绍了论文的主要内容和主要工作。

第二章介绍了云计算及 web 日志挖掘领域的关键技术,并简单介绍了 Hadoop 的基本理论以及两个核心技术——MapReduce 编程模式和 Hadoop 分布式文件系统 HDFS。

第三章针对传统基于单机的集中式 web 日志挖掘系统计算能力不够的问题,给出了基于 Hadoop 的 MapReduce 并行计算模型的 web 日志挖掘平台的详细设计策略与方案,并对并行系统的各个功能模块进行了详细的设计。

第四章首先针对目前挖掘用户浏览偏爱路径算法无法反映用户真实兴趣度的问题,结合网络拓扑图,给出了有用偏爱度的概念。并基于该概念对用户浏览偏爱路径算法进行了改进。同时,针对传统 web 数据挖掘算法不能被分布式执行的缺陷,将改进的求解用户偏爱路径的算法进行了并行设计,使算法能够在 Hadoop 集群架构下分布式并行执行。

第五章首先介绍系统开发的平台以及 Hadoop 环境的两种搭建方法,然后使用 eclipse 搭建了仿真运行的云计算环境。最后设计了两组实验分别用于测试改进算法的有效性和基于云计算的分布式并行算法的高效性。

第六章总结了目前所做的主要工作,同时提出了以后还需要进行的研究工作和研究方向。

第 2 章 云计算技术及 web 日志挖掘概述

在过去的 20 年中，Internet 的迅速发展使其成为世界上规模最大的公共数据源。它具有数据量大、数据类型多、异构、分布等许多特点，这使得在 web 上挖掘有用信息和知识的任务变得十分有趣，并且富有挑战。云计算的出现，加速了 web 数据挖掘的发展。

本章首先对云计算的定义及特点做了一个简要的概述，介绍了云计算所涉及的关键技术及其研究现状，以及国内外学者在云计算领域的研究热点及应用领域。然后介绍了云计算开源系统 Hadoop，重点介绍了它的两个关键技术——MapReduce 编程模型和 HDFS 分布式文件系统。最后对 web 日志挖掘做了一个概述，介绍了 web 日志挖掘的过程，web 日志挖掘主要用到的方法。

2.1 云计算介绍

2.1.1 云计算定义及特点

云计算是在 2007 年产生的新名词，到目前为止至少可以找到 100 种解释，还没有公认的定义。中国云计算专家刘鹏将云计算定义为：云计算(Cloud Computing)(刘鹏, 2010)是一种商业计算模型，它将计算任务分布在由大量计算机机构成的资源池上，使用户能够按需获取计算能力、存储空间和信息服务。它是基于互联网的超级计算模式，它主要关注如何充分地利用 Internet 上硬件、软件和数据的能力，以及如何更好地使网络中各个廉价的 PC 机协同工作发挥最大效用的能力。它的基本思想是，通过开发的技术和标准把硬件和软件抽象为分布的、可全球访问的、动态可扩展、可配置的资源结构，并对外以服务的形式提供给用户。用户通过互联网访问这些服务，不需要管理，只需按实际使用量付费。从研究现状上看，云计算具有一下特征：

(1) 大规模。“云”具有相当大的规模，Google 云计算已经拥有 100 多万台服务器，AMAZON、IBM、微软、Yahoo 等的“云”均拥有几十万台服务器。企业私有云一般拥有数百上千台服务器。“云”能赋予用户前所未有的计算能力。

(2) 虚拟化。云计算支持用户在任意位置、使用各种终端获取应用服务。所请求的资源来自“云”，而不是固定的有形的实体。应用在“云”中某处运行，但实际上用户无需了解，也不用担心应用运行的具体位置。只需要一台笔记本

或者一个手机，就可以通过网络服务来实现我们需要的一切，甚至包括超级计算这样的任务。

(3) 高可靠性。“云”使用了数据多副本容错、计算节点同构可互换等措施来保障服务的高可靠性，使用云计算比使用本地计算机可靠。

(4) 通用性。云计算不针对特定的应用，在“云”的支撑下可以构造出千变万化的应用，同一个“云”可以同时支撑不同的应用运行。

(5) 高可扩展性。“云”的规模可以动态伸缩，满足应用和用户规模增长的需要。

(6) 高度自治性。通过自动化配置管理服务，用户可以按需自动调配任务，以及根据应用环境的变化自动增加或减少服务的数量。

(7) 极其廉价。由于“云”的特殊容错措施可以采用极其廉价的节点来构成云，“云”的自动化集中式管理使大量企业无需负担日益高昂的数据中心管理成本，“云”的通用性使资源的利用率较之传统系统大幅提升，因此用户可以充分享受“云”的低成本优势。

2.1.2 云计算的关键技术

云计算的发展和应用离不开一系列创新技术支持，这些技术包括(王庆波, 2010)：

(1).虚拟化技术。它是云计算中最关键的技术之一，通过虚拟化技术，单个服务器可以支持多个虚拟机运行多个操作系统和应用，给云计算带来诸多优势：一方面可以提升基础设施利用率，实现运营开销成本最小化；另一方面可以通过整合应用栈和即时应用镜像来实现业务管理的高效敏捷。目前在云计算中普遍使用的三种虚拟机技术是：VMware Infrastructure、Xen 和 KVM。

(2).海量数据处理。在互联网时代，互联网数据的统计和分析很多是海量数据级的，单台计算机不可能满足海量数据处理的性能和可靠性等方面的要求。作为以互联网为计算平台的云计算，将海量数据分割后调度到互联网中的多个计算节点上批处理计算任务和海量数据，并建立一个可扩展的可靠的计算环境。当今世界最流行的海量数据处理编程模型是由 Google 公司的 Jeffrey Dean 等人所设计的 MapReduce 编程模型。

(3).大规模分布式存储。随着过去几十年互联网技术的发展，越来越多的互联网应用具有存储海量数据的要求，比如搜索引擎，这种需求催生了诸如分布式文件系统这类大规模分布式存储技术。云计算的出现给分布式存储带来了新的需求和挑战。在云计算环境中，数据的存储和操作都是以服务的形式提供的；数据的类型多种多样，包括了普通文件、虚拟机镜像文件这样的二进制大

文件、类似 XML 的格式化数据，甚至数据库的关系型数据等。目前在云计算环境下的大规模分布式存储方向已经有了一些研究成果，如 Google 公司的 BigTable，Amazon 公司的 S3 等。

(4).资源调度。云计算的海量规模为资源调度带来了挑战，目前的技术已经实现了在几秒钟内将一个操作系统进程从一台机器迁移到另一台机器。

2.1.3 云计算技术的研究现状

云计算所具有的优势使它引起了业界的广泛重视，受到许多大公司的推动，发展极为迅速。Google、亚马逊、IBM、微软、Yahoo 等大型 IT 公司是云计算的先行者。

(1) Google 云计算

Google 的搜索引擎、Google Maps、Google Earth、Gmail 等各种业务都需要面向全球用户提供实时服务，并都涉及到大量的数据，因此 Google 必须要解决海量数据的存储和处理问题，这促使了 Google 云计算的诞生。Google 云计算技术包括：Google 文件系统 (GFS)、分布式结构化数据存储系统 BigTable 以及分布式计算编程模型 MapReduce。

GFS 是一个大型的分布式文件系统，提供了海量数据的存储和访问能力，处于所有核心技术的最底层。它使用廉价的商用机器构建分布式文件系统，将容错的任务交由文件系统来完成，利用软件的方法解决系统可靠性问题，使得存储的成本成倍的下降。GFS 最大的特点是它采用了多种方法，从多个角度使用不同的容错措施来保证整个系统的可靠性。

MapReduce 是 Google 提出的一个软件架构，是一种处理海量数据的并行编程模式，用于大规模集的并行运算，它使得海量数据的并行处理变得简单易行。

BigTable 是 Google 开发的机遇 GFS 的分布式存储系统，它存储了 Google 的很多数据，如 web 索引、卫星图像数据等结构化和半结构化数据。

(2) 亚马逊云计算

亚马逊很早就进入了云计算领域，并在云计算、云存储等方面一直处于领先地位。目前亚马逊的云计算服务主要包括：弹性计算云 EC2、简单数据存储 S3、简单数据库服务 Simple DB。

EC2 是亚马逊提供的云计算环境的基本平台。通过这个平台，用户可以按照自己的需求随时创建、增加或删除实例。通过配置实例数量可以保证计算能力随着通信量的变化而变化。这样在提高访问者用户体验的同时也降低了成本，对中小企业来说是非常有利的。

S3 是亚马逊推出的简单存储服务，通过亚马逊提供的服务接口，用户可以将任意类型的文件临时或永久地存储在 S3 服务器上。

SimpleDB 与 S3 不同，主要用于存储结构化的数据，并为这些数据提供查找、删除等基本的数据库功能。一些新兴的中小企业有选择地将 EC2、S3、SimpleDB 组合使用，可以节省大量的前期基础设施投资。

(3) 微软云计算

微软在 2008 年推出了自己的云计算平台——Windows Azure Service Platform，它主要包括四个组件：Windows Azure、Microsoft.NET、Microsoft SQL 服务、Live 服务。

Windows Azure 是微软云计算操作系统，位于云计算平台的最底层，它只是一个服务平台，用户可以用它在通过互联网访问的微软数据中心运行 Windows 应用程序和存储应用程序数据。

Microsoft.NET 服务为云端和本地的应用程序提供常用的基础功能模块，包括三个构件：访问控制服务、服务总线服务、工作流服务。

Microsoft SQL 服务为云端和本地的应用程序提供基于 SQL Server 的数据服务。

Live 服务将 Windows Live 应用程序功能进行封装，扩展到云端，使现有的应用程序可以通过 Live 服务来存储和管理自己的数据信息。

2.1.4 云计算当前研究热点及应用领域

学术界对云计算的研究主要集中在以下方面(刘鹏, 2010)：

(1) 体系结构研究。

目前对云计算体系结构的划分有两种，一种是 Youseff 等依据可组合性将云系统划分成云应用层、云软件环境层、云软件基础设施层、软件内核、固件和硬件五层。他重点介绍了云的层次栈，阐述了各层次内涵、作用、架构，并深入分析了它们之间的相互依存的关系，有利于加强对云系统分类、联系和内部依存关系的理解。

另一种是 Lenk 提出的一个通用的云计算栈，将云技术和服务分为不同的层次，并通过例子对每个层次进行阐述，说明该模型是如何从总体上涵盖现有的云计算计算的，进一步阐述了该栈在云计算环境建模中的应用。Lenk 将云计算划分成了基础设施即服务层 (IaaS)、平台即服务层 (PaaS)、软件即服务层 (SaaS) 和人员即服务层 (HaaS)。

目前，国内外学者主要都是以 Lenk 提出的云架构服务层次来对云计算进行研究的。

(2) 关键技术研究

云计算的发展和应用离不开一系列创新技术支持，这些技术包括：虚拟化技术、安全管理、云监测、能耗管理等。

云监测。在云计算中，对海量数据进行处理、对虚拟机进行监测是非常重要的和关键的，研究人员也对这方面进行了大量的研究工作，目前已经出现了一些典型的案例。例如 Jerome Boulon 等设计实现的建立在 Hadoop 上的数据收集系统——Chukwa，用以监测和分析大规模分布式系统。

能耗管理。如何在包括数以万计计算资源的云计算基础设施中有效地整合资源、降低运行成本、节省运行计算机所需要的能源是一个备受关注的热点问题。Shekhar Srikantaiah 等研究了云计算中能源消耗、资源利用率以及整合后的工作性能之间的内在关系，对云平台中能源优化问题作出了时间和探索。

资源调度。云计算的海量规模为资源调度带来了挑战，目前的技术已经实现了在几秒钟内将一个操作系统进程从一台机器迁移到另一台机器。

(3) 支撑平台研究

随着云计算的不断发展与成熟，研究机构和科研人员在云平台设计与实现方面做了大量的研究实践工作，满足不同应用需求的云计算平台不断出现并投入使用，比较典型的有：Lizhe Wang 等通过采用新的云技术整合现有的网络基础设施来为数据中心建立一个科学云——Cumulus；IBM、SAP 等科研机构联合开发的一个云服务融合平台——RESERVOIR；Peng Bo 等设计实现了一个基于 Hadoop 变种的云计算平台——TPlatform；Yahoo 构建了一系列可扩展、高度可用的数据存储和处理服务。

(4) 应用研究

云计算在数据挖掘、生物、网络安全、地理信息等方面已经有了一些研究成果。

在数据挖掘方面，Robert Grossman 等人开发了一个类似于 Hadoop 的广泛应用与数据挖掘领域的云计算平台：Sector and Sphere，该系统已经广泛应用于数据挖掘领域(Robert Grossman, 2008a)(Robert Grossman, 2008b)，其中 Sector 是部署在广域网上的类似于 Hadoop DFS 的分布式存储系统，已经用于数字巡天系统——Sloan。Sphere 是建立在 Sector 之上的计算，为用户编写分布式数据密集型任务提供了编程接口。我国在 2008 年底，中科院计算所开发出中国第一个基于云计算平台的并行数据挖掘平台 PDMiner，用于中国移动 TB 级数据挖掘，目前，他们正在研究面向 web 基于云计算数据挖掘服务系统 WPDMiner。

在生物方面，Massimo(2008)等用 Hadoop 实现了两个算法：一个是用来比对生物序列的一级结构的算法——BLAST，一个是用于分析 DNA 微点阵的算法

——GSEA，这两个算法可以很好的用于解决生物信息计算的问题；马里兰大学的 Michael C(2009)使用开源的 MapReduce 搭建了一个在多个计算节点上并行执行的 DNA 信息分析平台——CloudBurst。

在网络安全方面，中国云计算专家刘鹏在 2003 年提出的反垃圾邮件网格是云安全技术的前身；金山、瑞星等厂商提出了“安全云”计划，将病毒资料库放在“云”端，与客户端通过网络连接，当“云”端发现不安全链接时，直接阻止其进入用户机器，保证用户机器安全。

在地理信息空间分析方面，Qichang Chen 等提出了一个高效率执行 GIS 应用程序的基于 MapReduce 集群的并行和分布式计算平台——MRGIS。

在高校教育平台搭建领域，华盛顿大学的 Justin Capps(2009)等人搭建了一个由分布在世界各地的 1000 个计算机组成的自由、教育研究平台——Seattle。

在科学计算领域威特沃特斯兰德大学的 Scott Hazelhurst 使用亚马逊的弹性计算云(EC2)进行了虚拟高性能计算的研究；在商业应用方面，IBM 开发了一个企业文本分析平台，应用 Hadoop MapReduce 框架来支持分类工作。

2.2 开源云计算系统 Hadoop 介绍

2.2.1 Hadoop 平台概述

随着 Internet 数据的爆炸性增长，传统的技术架构已经越来越不适应当前海量数据处理的要求，Hadoop 就是在这样的环境下出现的。

Hadoop 是 Apache 开源组织的一个分布式计算框架，它可以运行在大型集群的廉价硬件设备上，让程序自动分布到集群中的普通机器上并发执行。它包括两个核心组件：MapReduce 和 HDFS。

2.2.2 MapReduce 编程模型

MapReduce 是现在普遍所采用的一种编程模型，是一种分布式计算模型。由于它是开源的，任何人都可以使用这个框架来进行并行编程。

MapReduce 将一个复杂的运行在大规模集群上的并行计算过程抽象为两个函数：Map（映射）和 Reduce（化简）。Map 步骤负责根据输入的键值对 (key/value) 生成中间结果，中间结果同样采用 key/value 对的形式。Reduce 步骤则将所有中间结果根据 key 进行合并，然后生成最终结果。由于 MapReduce 所具有的这个两个过程，适合用它处理的任务有一个基本要求：待处理的数据集可以分解成许多小的数据集，而且每一个小数据集都可以完全并行地进行处

理。

MapReduce 的运行环境由两种不同类型的节点组成：Master 和 Worker。Worker 负责数据处理，Master 负责任务调度及不同节点之间的数据共享。基本工作流程如图 2.1 所示(刘鹏, 2010)。

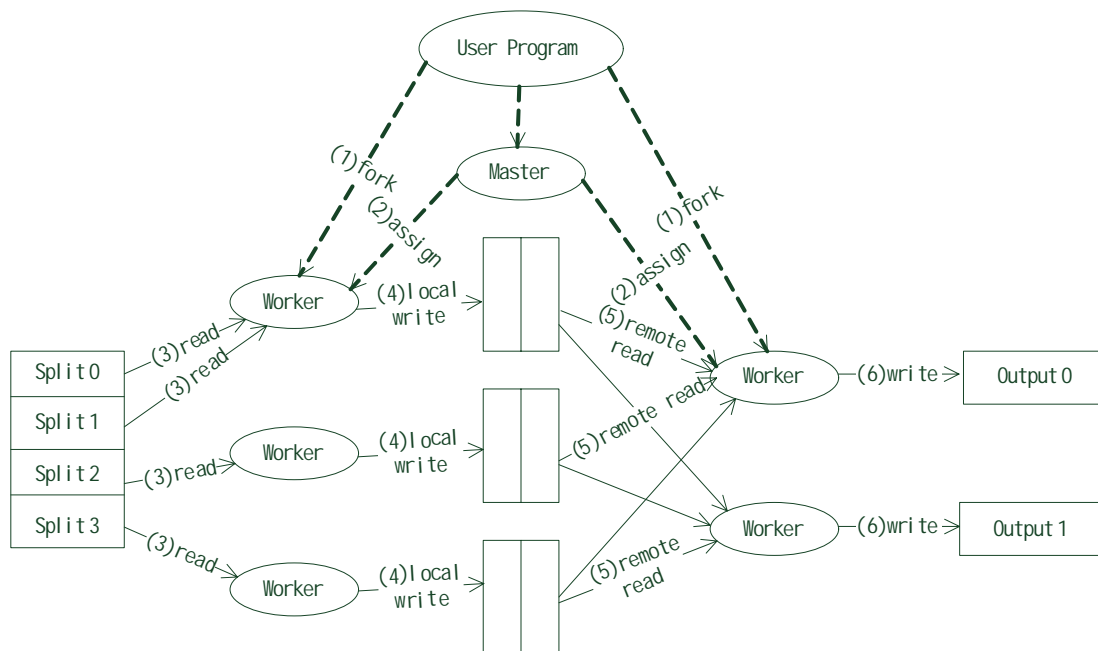


图 2.1 MapReduce 基本工作流程图

(1) Fork。利用 MapReduce 提供的库将输入数据切分成大小不等的 M 份，然后在计算机集群中启动程序。

(2) Assign map/reduce。Master 负责找出集群中所有空闲的 Worker 节点，并为它们分配子任务，包括 M 个 Map 子任务和 R 个 Reduce 子任务。

(3) Read。获得 Map 子任务的 Worker 读入对应的输入数据，从输入数据中解析出 key/value 对，并调用用户编写的 Map 函数。

(4) Local write。缓存在内存中的 Map 函数的中间结果，被周期性地写入本地磁盘。写入磁盘的数据根据用户指定的划分函数被分为 R 个数据区。这些中间结果的位置被发送给 Master。

(5) Remote read。负责 Reduce 任务的 Worker 获取子任务后，使用远程调用的方式执行 Map 任务的 Worker 节点的本地磁盘读取数据到缓存，然后遍历所有中间结果，按关键字进行排序，把具有相同关键字的数据分为一类。

(6) Write。Reduce 函数的结果被写入到一个最终的输出文件。

当所有的 Map 子任务和 Reduce 子任务完成后，Master 将 R 份 Reduce 结果返回给用户程序。用户程序可以将这些执行 Reduce 子任务的 Worker 生成的结果数据合并得到最终结果。

2.2.3 分布式文件存储系统 HDFS

随着互联网技术的发展，基于互联网的应用对存储海量数据的需求越来越高，比如搜索引擎。这也催生了云计算中分布式文件系统。云计算的数据存储技术主要有 Google 非开源的 GFS 和 Hadoop 开发的 GFS 的开源实现 HDFS。由于 HDFS 的开源性，大部分 IT 厂商都采用 HDFS 数据存储技术。

HDFS 是一个运行在普通廉价的计算机上的分布式文件系统，它是为以流式数据访问模式存储超大文件而设计的文件系统，是一个主从结构的体系。集群中有一个 NameNode 和很多个 DataNode。NameNode 管理文件系统的元数据，DataNode 存储实际的数据。用户联系 NameNode 以获取文件的元数据，而真正的文件 I/O 操作是直接和 DataNode 进行交互的。其结构示意图如图 2.2 所示(刘鹏, 2010)。

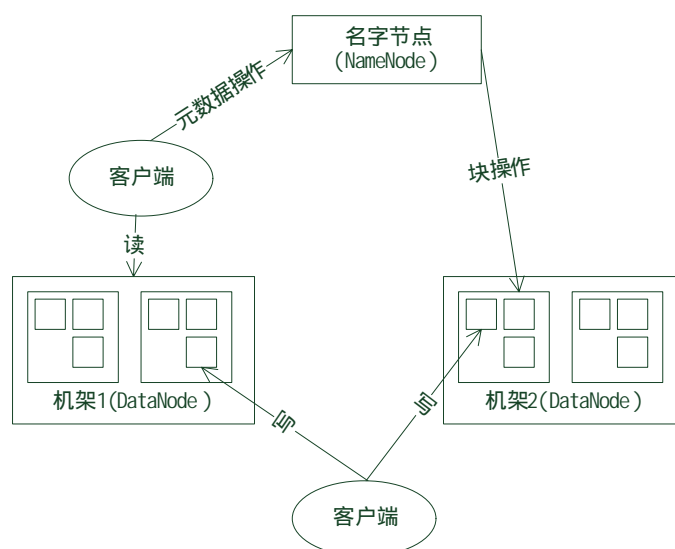


图 2.2 HDFS 结构示意图

由图 2.2 可知，NameNode 和 DataNode 之间相互操作如下：当客户端向 NameNode 发起文件写入的请求，NameNode 根据文件大小和文件块配置情况，向客户端返回 DataNode 的信息(如存储位置、存储空间大小等)。客户端根据 DataNode 的 IP 地址，将划分成多个块的文件按顺序写入到每一个 DataNode 中；当客户端向 NameNode 发起文件读取请求时，NameNode 返回文件在 DataNode 的存储信息(如文件被分成多少块，每一块的存储位置、大小等)。客户端根据返回的信息直接读取 DataNode 上的文件信息；为了保证在故障情况下数据存储也能正常进行，HDFS 还提供了冗余备份、副本存放、心跳检测等措施来保证数据存储的可靠性。

在云计算分布式计算环境中，网络带宽是一个相对紧缺的资源。为了有效

地减少网络数据流量，HDFS 采取将输入数据保存在构成集群机器的本地硬盘上的方式来减少网络带宽的开销，提出了“计算向存储迁移”的思想，提供了让程序将自己移动到离数据存储的位置更近的一系列接口。当在一个足够大的集群上运行 MapReduce 操作的时候，大部分输入数据都是在本地机器读取的，这已过并不占用网络带宽。这种思想在涉及到大量数据的 web 日志挖掘中具有很好的应用价值。

2.3 web 日志挖掘概述

近年来，随着 Web 挖掘是利用数据挖掘方法，从 web 的超级链接结构、网页内容和使用日志中，提取隐藏的、有用的模式和知识。根据在挖掘过程中使用的数据类别，web 挖掘任务可以被划分为三种主要类型(Bing Liu, 2009)：web 结构挖掘、web 内容挖掘和 web 使用挖掘。Web 结构挖掘就是从表征 web 组织结构的超链接中推导出知识；web 内容挖掘就是从网页内容中抽取有用的信息；web 使用挖掘就是从服务器端记录每位用户点击情况的使用日志中挖掘用户的访问模式。

对一个网站而言，网页浏览量、点击数、用户访问路径等是反映网站用户访问情况的重要指标。而 web 日志文件记录了用户的一系列点击流信息，因此对 web 日志文件进行挖掘，依靠数据挖掘技术从 web 日志文件中海量的、有噪音的、随机的点击流数据中提取有用的信息，从而帮助企业有效地提高网站的服务，例如通过分析访问者的来源，可以使一个网站有针对性地提供内容；通过分析用户偏爱访问路径，可以调整网站的内容和结构。

2.3.1 Web 日志挖掘的过程

根据标准的数据挖掘过程，web 日志挖掘过程也分为三个相互依赖的阶段：数据收集和预处理、模式发现、模式分析。图 2.3 描绘了 web 日志挖掘的整个过程。

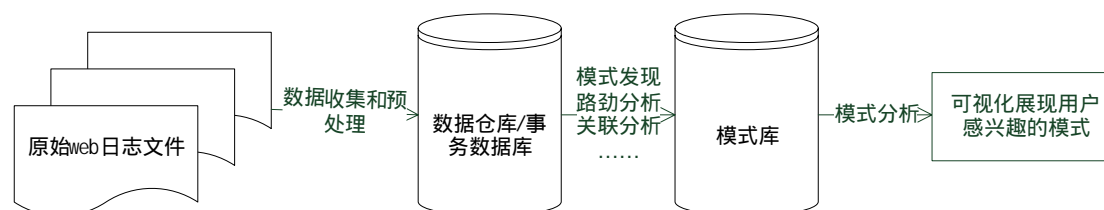


图 2.3 web 日志挖掘过程

(1) 数据收集和预处理。

由于 web 自身所具有的分布广泛、用户众多等特性，使其上所产生的数据是海量的、地理上分布的、异构的、动态的，并且存在着大量的冗余与噪声。同时，这些数据不同于传统数据库中的结构化数据，大多是一种半结构化甚至无结构的数据。如果不对这些数据预先进行处理，则很难从中找出有用的信息。可见，数据的准备过程是 web 挖掘中很重要的一步，也是最花时间、计算量最大的步骤，涉及到对原数据的预处理，整合来自不同地方的数据并且将整合好的数据转换成合适的形式以作为特定的数据挖掘方法的输入。

Web 日志挖掘的挖掘对象就是 web 日志文件，用户对网站的访问信息基本都记录在 web 日志文件中，常见的 web 日志一般有两种格式：CLF 和 ECLM，常见的 web 日志一般采用 ECLM 日志模式。它的结构如表 2.1 所示。

表 2.1 ELCM 日志模式结构

IP Address		Time/Data	Method/URL/Protocol	Status	Size	Referer	Agent
222.195.77.79	-cm	[15/Mar/2011:12:45:25-0600]	GET/~en/HTTP/1.0	304	0	http://www.clear-ep.net	Mozilla/2.0Gold B/ (WinXP; I)

其他的数据来源如网站文件、业务数据库等对于数据准备和模式发现来说也是必须的，通过对它们的深度挖掘，可以帮助企业决定客户的生命周期、产品和服务的交叉营销策略、评估促销活动的效果、优化 web 应用程序的功能、为访问者提供更个性化的内容以及为自己的 web 空间找到最有效的逻辑结构。

在对原始数据进行预处理时，有以下几个步骤：

数据的融合和清理：并不是 web 日志中所有的数据对分析是必要的，如用户页面中包含的图片、音频、视频、广告等都会被自动下载保存在日志文件中，这些都是要删除的与挖掘无关的数据，并将来自多个 web 和应用程序服务器的日志文件进行合并。

页面访问识别：每个页面访问记录了大量的属性，和大量面向产品的事件相关，如产品视图、注册、改变购物车等。通过它们对用户行为分类，这在电子商务网站的个性化推荐系统中有很好的应用价值。

用户识别：对于注册用户直接记下 ID 即可；而对于非注册用户，单单 IP 地址并不足以识别出用户，因为不同的用户可以通过同一台电脑访问网站，它们的 IP 地址却是相同的。一般最常用的 web 日志挖掘工具使用的技术是基于日志/站点的方法(姚洪波, 2006)，基本思想是：IP 优先考虑，即 IP 不同代表不同的用户；IP 相同时，考虑浏览器和操作系统类型，默认不同的浏览器和操作系统为不同的用户；将访问日志、引用日志和站点拓扑结构结合，构造每一位用户的浏览路径，如果用户请求的页面同已浏览过的页面之间没有超级链接关

系,则认为是具有相同 IP 地址的多个用户。

会话识别:将每个用户的用户活动记录分成一个一个会话的过程,每个会话代表了一次对站点的访问。通过会话识别可以让我们获得一个用户一次访问站点的真实行为序列。会话识别常用的方法有三种(任晓霞,2008):Cooley 等人提出的一种事务识别会话的方法——序列长度法;最大向前序列法;Timeout 方法。

路径完善:客户端或代理端的缓存功能经常会导致对那些被缓存的页面和对象的访问引用的丢失,这需要依靠服务器日志上的站点结构和引用信息来完善路径。

数据整合:是数据预处理的最后一个步骤,这一步将经过上述处理 web 日志得到的点击流信息与用户数据(用户统计信息、用户方位呢频率和购物历史记录)整合,存放在数据仓库或事务数据库中,作为模式发现的数据来源。

(2) 模式发现。

主要是对数据预处理所形成的文件,运用数据挖掘的一些有效方法,例如关联规则、聚类与分类分析、序列模式分析等,来发现反映用户特定行为的隐藏模式以及 web 资源、会话和用户的简要统计。

(3) 模式分析

对上一步挖掘出的模式、规则进行分析,找出用户感兴趣的模式,提供可视化的输出结果。

2.3.2 Web 日志挖掘的主要方法

Web 日志挖掘的目的是为了捕获网页间的访问关联以及客户的行为模式并建立模型,给访问网站的客户建立档案。这些模式能被用来更好地理解不同页面间的访问关系以及客户的各种行为,进而提高网站组织和结构的设计,并且通过向客户提供动态的产品和服务推荐带来个性化体验。因此日志挖掘的主要方法也是围绕此展开。目前常用的 web 日志挖掘方法主要有:关联及相关度分析、序列模式和访问路径分析、聚类分析和访问路径分析以及用户分类和预测分析。这些常用数据挖掘方法获取的用户行为模式,解决了页面自动导航、页面重要性评价以及改进网站设计,提高网站运营效益等问题。

(1).关联及相关度分析

关联规则发现和统计相关度分析可以获取用户页面访问行为间的关系,找到普遍被一起访问的页面,用于有效地改进站点内容,提供有效的交叉交易产品推荐。目前已有大量的基于关联规则的推荐系统,例如 Sarwar(2000)等人在协同过滤的背景下,在电子商务的 top-N 推荐系统中使用了关联规则。

(2).序列模式和访问路径分析

序列模式挖掘是用来发现会话间的模式，获取用户访问习惯，如在一套按时间顺序排列的会话或事务中一个项目的存在跟在另一个项目后面。它可以从 web 日志中寻找用户普遍访问的规律，捕捉用户路径之中常用的导航路径，可以很好的帮助企业安排网页内容和结构。

(3).聚类分析和访问者分割

聚类是对符合某一访问规律特征的用户进行用户特征挖掘，从而从 web 访问信息中分割出具有相似特征的用户群。得到的用户分组可以在电子商务应用中进行市场划分或给具有相似兴趣的用户提供个性化的 web 内容。它还可以被用于建立基于 web 的“用户社区”以反映具有相似兴趣的用户，并且还用于学习可以在个性化应用中提供动态建议的用户模型。

(4).用户分类和预测分析

分类主要用于对用户行为特征进行归类识别，根据各种可挖掘出的某些共同特性来对新的用户行为进行分类。例如根据用户的统计属性以及导航活动统计出每个用户特定期间的购物记录综合，将用户分成高购买倾向和低购买倾向，使企业有针对性的进行营销活动。

第3章 基于云计算的 web 日志挖掘系统设计

数据挖掘作为挖掘知识的一个有效手段，受到了学术界以及商业界的广泛关注，它无论是在算法理论还是应用上都得到了很好的发展。总的来说，数据挖掘软件的发展经历了以下几个阶段(徐超, 2010)。

表 3.1 数据挖掘软件的发展历程

发展阶段	特征	数据挖掘算法	集成	分布式计算模型	数据模型
第一阶段	作为一个独立的应用	支持一个或多个算法	独立的系统	单个机器	向量数据
第二阶段	和数据库以及数据仓库集成	多个算法；能够挖掘一次不能放进内存的数据	数据管理系统，包括数据库和数据仓库	同质、局部区域的计算机群集	有些系统支持对象、文本和连续的媒体数据
第三阶段	和预测模型系统集成	多个算法	数据管理和预测模型系统	Intranet/extranet 网络计算	支持半结构化数据和 web 数据
第四阶段	分布式数据挖掘	多个算法分布在多个节点	算法、调度系统	网格计算	普遍存在的数据模型
第五阶段	基于云计算的并行数据挖掘与服务	同一个算法分布在多个节点；多个算法之间也可以并行	计算资源按需分配	云计算	BigTable DFS

从表 3.1 可以看出，数据挖掘已经进入了基于云计算的并行数据挖掘与服务的时代。

本章首先对现有的 web 日志挖掘系统的不足进行了分析，借助云计算的优势，提出了一种基于云计算的 web 日志挖掘思路。然后对基于云计算的 web 日志挖掘系统的总计架构进行设计，最后分别对该系统的主要功能模块进行了详细的设计。

3.1 当前 web 日志挖掘系统分析

3.1.1. 基于单一节点的集中式 web 日志挖掘系统

传统基于单一节点的集中式 web 日志挖掘系统是将所有挖掘任务和数据存储任务都在一台服务器上完成。这种挖掘系统在数据挖掘产生初期，任务复杂度较低时具有很好的挖掘效率，但随着数据的不断增多，以及 web 挖掘任务的不断

断复杂化，基于单一节点的挖掘系统也渐渐地暴露出它的一些弊端：(1) 随着网络技术的发展，人们对网络服务应用的需求变得更加复杂，集中式挖掘系统已经不能满足这种海量数据处理的要求；(2) 随着 web 上数据的不断增多，集中式挖掘系统的存储空间已经达到饱和，不能再保存日益增长的海量数据。

虽然已有很多大型 IT 公司开发了一系列高性能计算机，但是这些硬件设施所带来的成本是中小型不能承受的。同时，web 上的数据是呈指数级增长的，其硬件设施的更新程度远远赶不上数据的增长速度。

3.1.2. 并行数据挖掘系统

为了解决计算能力和存储能力不足的问题，国内外学者们相继提出了各种分布式并行数据挖掘系统。这些系统有基于集群的、有基于 Agent 的(刘业政, 2004)、基于 CORBA 的(唐卫宁, 2002)，但实现却相对复杂且只能针对特殊应用。之后，M Cannataro(2001)等人基于 Globus Toolkit 设计了一种分布式并行知识发现平台，该平台利用 Globus Toolkit 所提供的网格计算能力，解决了传统数据挖掘计算能力不足的问题。近几年集中在基于 Globus Toolkit 平台上并行数据挖掘算法的实现与改进方面的研究(郑晶, 2010)(齐玉成, 2010)。但网格计算一般用于科学计算，为专用领域的问题而设计，缺少商业化实现，且 Globus Toolkit 是基于中间件技术，需要通过编程或安装设置来搭建底层架构，增加了系统实现的难度(纪俊, 2009)。另一方面，部署一个网格环境需要耗费大量服务器等硬件，对中小企业、个人用户来说，很难承受如此大的开销来满足处理能力较高的系统硬件需求。即便是能够购入如此多的硬件设备，在系统不工作或非满负荷工作时，会造成很多资源浪费。

3.1.3. 基于云计算的 web 日志挖掘系统

针对上述问题，我们需要开发这样一个 web 日志挖掘框架：它可以在短时间内进行海量数据的处理；它应当为海量数据提供一个安全可靠的存储平台；它在可接受的时间内，通过有效的调度和负载均衡，实现海量数据的并行处理；它在扩展集群规模时，购置成本应该是在企业能够承受范围之内的。

云计算以其独特的优势，给 web 日志挖掘提供了一个具有较强应用价值的解决方案。其原因有：

(1) 低成本分布式并行计算环境。不同规模的企业都适合采用云计算技术来部署自身的计算、存储环境，为中小企业的海量数据处理提供低成本计算环境。同时，大型企业采用大量廉价的 pc 机部署云计算平台，不再完全依赖大型

高性能计算机。

(2) 开发方便，使用者只需专注于问题本身，而不用关心底层如何实现。使用者不需要考虑数据的划分、数据的分配、计算任务的调度等工作。

(3) 由于分布式并行处理各个任务，因此数据处理规模大幅度提高。

(4) 在分布式环境中，可以很方便的增加和删除节点，扩展性好。

(5) 它可以自动处理失效节点，当节点瘫痪时，仍能完成计算任务，具有很高的容错能力。

通过以上分析，开发一个基于云计算的 web 日志挖掘系统是很有应用价值的。目前还没有人提出将云计算技术用于 web 日志挖掘中。因此，本文设计了一个基于云计算的 web 日志挖掘系统。

3.2 云计算平台总体架构设计

由于 web 日志文件是分布存储在网络中的服务器上的，如果采取传统的数据收集与存储模式，即先把数据集中到一个数据仓库或事务数据库中，然后在采用数据挖掘技术对其进行挖掘处理。这要求有高速的通信网络，并且大数据量的传输将会占用大量网络带宽。另外，由于数据的私有性、保密性、异构性以及系统的不兼容性，要把所有数据都集中到一个平台上也是不现实的。即便是完成了数据收集工作，在数据仓库这个大数据集上进行所有的挖掘任务，势必会增加系统的工作负荷，影响挖掘效率。本文设计的基于云计算的 web 日志挖掘系统则充分利用网络中大量的廉价资源，将一个大型、复杂的计算任务分派到多个节点进行分布式并行处理，从而降低单一节点的工作负荷。基于云计算的 web 日志挖掘系统在两个层面进行分布式并行处理：一是在对 web 日志文件进行数据预处理时；一是在进行挖掘任务时，算法的并行执行。与传统的 web 数据挖掘系统大致相同，只是在数据预处理时，不再将预处理后的数据集中到一个数据仓库或事务数据库，而是就地存储在本地磁盘中；挖掘任务则交由集群中的大量计算机分布式并行处理。因此，基于云计算的 web 挖掘系统少了一个数据仓库，多了一个用于并行挖掘和存储 web 日志文件的集群系统。其总体架构如图 3.1 所示。

从图 3.1 可以知道，基于云计算的 web 日志挖掘系统可以分为三层，它们分别是可视化交互层、功能层和硬件资源层。可视化交互层用以实现用户与系统的可视化交互，接收用户的挖掘请求，并对挖掘结果进行形象展示，使用户能方便的与系统进行交互，同时根据用户不同的需求提供了各种可视化挖掘模块，如客户行为分析、市场分析等；功能层是整个框架的核心，它集成了整个 web

日志挖掘的所有处理过程，包括数据的并行预处理、算法的并行执行以及日志文件的分布式存储等，在这一层，我们实现了一个基于 MapReduce 的并行数据预处理管道和一个基于 MapReduce 的并行挖掘算法；硬件资源层即是集群系统中位于最底层的实际物理硬件资源，它包括大量低成本的计算机、网络设施等，该层通过一个集群管理服务器来管理命名空间、访问控制、数据检索以及调度集群中的物理资源，如计算资源、存储资源等。

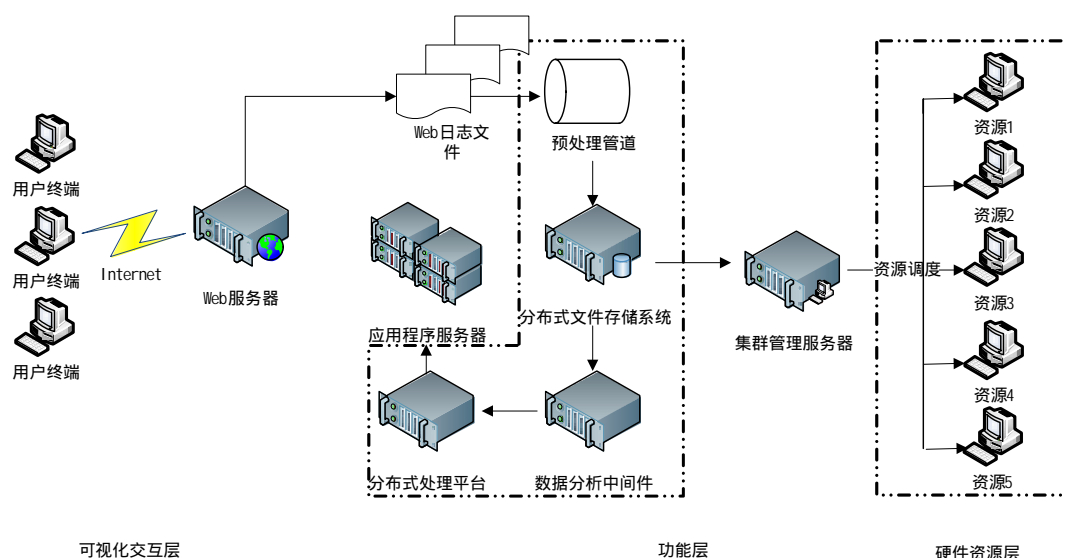


图 3.1 云计算平台总体架构

按云计算平台中所有物理节点的功能不同，我们将节点分为三类：一类是主控节点（Master），在云中，主控节点只有一个，是整个系统的管理节点，负责调度与协调服务节点之间的工作进程，处理节点失效、负载均衡等功能；一类节点是算法存储节点，负责存储数据挖掘以及数据清理所需的算法；还有一类节点是服务节点，它是任务实际执行的地方，负责存储经过预处理管道处理后的 web 日志文件以及执行由主控节点分配的任务，并把计算结果返回给主控节点。整个云计算平台数据处理流程如图 3.2 所示。

服务节点每隔一段时间向主控节点发送一个信号，以证明该服务节点工作正常。主控节点将工作正常的服务节点放入空闲节点列表。当主控节点接收到用户任务请求时，查找空闲节点列表，看挖掘任务所需数据的节点是否空闲或是否正常工作。如果该节点存在于空闲节点列表中，则将子任务分配到数据所在节点；如果不在空闲节点列表中，则将任务分派到该数据块备份的节点执行子任务。

从图 3.2 可知，计算节点和数据分块存储节点位于同一台机器上，这是因为网络带宽是基于云计算的挖掘系统的瓶颈，减少通过网络发送的数据量，能够大大改善基于云计算平台的系统的执行效率。因此我们从本地磁盘读取数据后直接在本地进行计算任务，并且 MapReduce 产生的中间数据也写到本地磁盘，

以保留网络带宽。

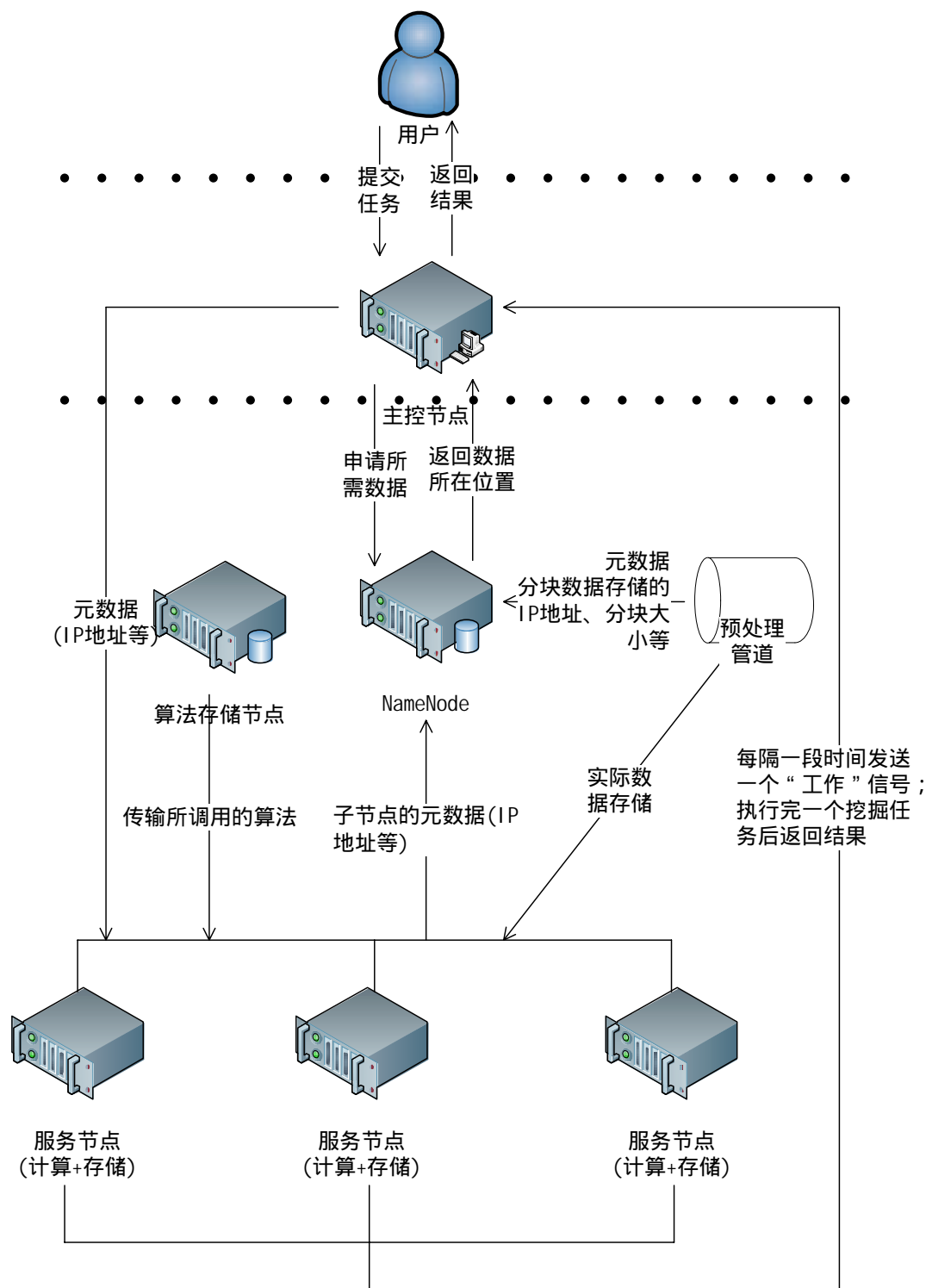


图 3.2 云计算平台数据流程图

3.3 可视化交互平台设计

可视化交互层是用户和系统交互的通道。用户首先向系统提出挖掘请求，

然后系统根据用户请求执行相应的挖掘任务，并最终将结果形象地展示给用户。该层主要包括四个功能模块：算法管理、日志文件管理、日志挖掘以及结果展示。其整体架构如图 3.3 所示。

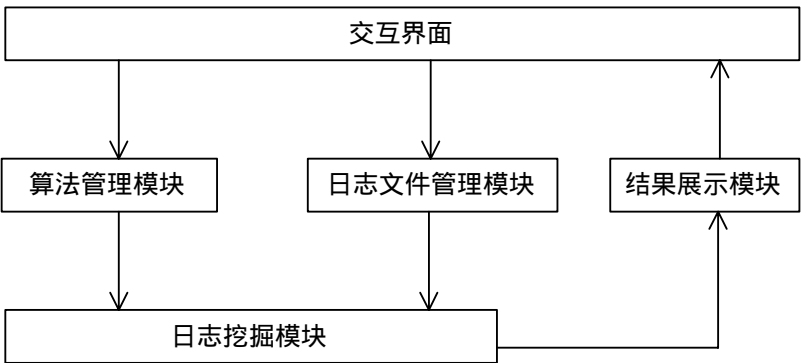


图 3.3 可视化平台整体架构

3.3.1 算法管理模块

算法管理模块主要负责对数据挖掘算法进行动态管理，包括新增新算法、修改旧算法以及删除旧算法。每个算法都以一个脚本文件的形式存储在 Web 日志挖掘算法库中。通过对这些脚本文件的管理，用户就可以动态对挖掘算法进行添加、编辑以及删除等操作，这种即插即用的方式可以大大降低算法维护的代价，增强整个云计算平台的通用性。所有的挖掘算法脚本文件都保存在算法存储库中，构成了一个数据分析中间件。其工作流程如图 3.4 所示。

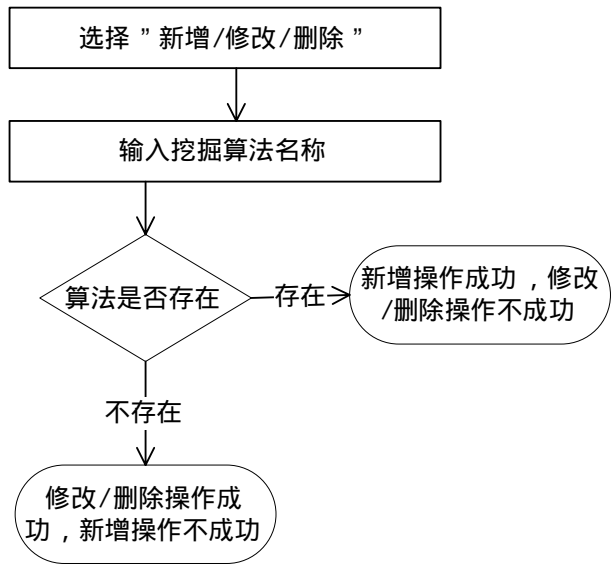


图 3.4 算法管理流程图

许多传统的挖掘算法只适用于小规模输入数据。随着数据量的增加，传统挖掘算法处理海量数据的速度会因计算量的增大而减慢甚至无法运行，这将消耗大量的计算时间。将传统挖掘算法进行并行化处理，使之能够部署到云计算平台的

分布式环境中并行执行,是解决这一问题的有效方法。数据分析中间件就是实现一些可用于分布式并行处理的算法,它依托功能层中的分布式计算和分布式文件存储系统对海量数据进行处理。

3.3.3.1 并行数据挖掘算法概述

目前,有不少经典的数据挖掘算法已经实现了并行处理。但并不是所有的算法都可以通过改进后适合在分布式并行环境中执行。例如,决策树就属于一种非易于并行的数据挖掘算法,因为它在运行的时候会产生很大的通信延迟和通信量,该延迟会抵消并行计算带来的高效率,最终的效果取决于谁的影响大。

(1).并行关联规则算法

目前所有的关联规则挖掘算法都是在 Apriori 基础上改进得到的,它们有着共同的缺陷:对数据库进行多次扫描,产生大量的候选集,占用大量的系统资源。针对这个缺陷,国内外学者已经提出了许多关联规则的并行算法,其并行思想是:将候选集分成几部分,分别交由不同的处理机处理,产生局部频繁项集,然后各个处理机之间进行通信来产生全局频繁项集。这种并行处理方法有一个缺陷,就是在由局部频繁项集生成全局频繁项集时,会造成节点间巨大的通信开销。同时,每个处理器生成频繁项集的时间也是一个瓶颈。

(2).并行聚类算法 k-means

此算法并行采取如下方法:首先将训练样本进行分割,每个子样本分配给一个处理机处理。处理机完成计算任务后,以最靠近的训练样本来更新簇中心(刘华元,2006)。

(3).并行分类算法 SPRINT

SPRINT 算法分为建树和剪枝两个阶段,其中最耗时的是建树阶段,因此它的并行处理集中在建树阶段。它首先将属性列表平均分配到 N 个处理机上,同时求解分割点,得到局部的对叶节点最佳的分割点,对于连续数据,接着就比较 N 个分割点求出最小;对于离散数据,接着就只需从各个处理机上收集计数矩阵的数据,相加求和就得到全局的计数矩阵。

(4).并行 web 结构挖掘算法

目前广泛应用的 web 结构挖掘算法是 Brin 等人提出的 PageRank 算法,Google 搜索引擎的核心算法便是基于此算法的。由于 pagerank 算法产生的邻接矩阵一般是一个稀疏矩阵,这会造成大量存储资源和计算资源的浪费;同时该算法需要多次迭代,因此当网页数量增加时,在执行效率上会产生瓶颈。

针对这个缺陷,高勋(2010)基于开源云计算基础架构 Hadoop 将 Pagerank 算法与 mapreduce 框架结合,给出了基于 MapReduce 的 Pagerank 算法。并对算法

进行改进从而加快 Mapreduce 迭代阶段执行速度。

3.3.3.2 本文给出的并行算法

云计算环境下的并行化跟一般的并行方法不同,它注重多个节点之间乃至多个集群之间的并行化。传统挖掘算法能否 MapReduce 化是海量数据处理能否在 hadoop 平台上成功执行的关键。因此,算法的并行化是一个很值得研究的问题。

本文重点研究基于 web 日志文件的用户浏览偏爱路径挖掘,因此,本文给出的并行算法是在传统浏览偏爱路径挖掘算法的基础上改进的。在并行改进之前,本文还对目前注重访问频度的用户浏览路径挖掘算法进行了修正,给出了一种基于有用偏爱度的用户浏览偏爱路径挖掘算法(Preferred Browsing Paths Mining Algorithm based Useful-preference, PBPU),实验表明改进后的算法更能真实的反映用户的浏览兴趣度。该算法将在第四章详细阐述。

3.3.2 日志文件管理模块

日志文件管理模块主要负责对 web 日志文件进行动态的分布式存储管理,包括新增 web 日志文件、删除过时的 web 日志文件、web 日志文件的备份等。该模块提供一个可视化界面对 web 日志文件进行管理,后台根据用户提交的指令直接对分布式文件系统中的日志文件进行管理。这些日志文件直接由其生成的日期命名。

3.3.3 日志挖掘模块

日志挖掘模块主要根据用户提交的参数(算法名称和需要挖掘的日志文件),执行用户请求的挖掘任务。首先,系统接收到用户提交的参数后,根据算法名称在算法库中查找算法,如果算法存在则将算法传输至 web 日志文件所在的节点,执行挖掘任务,否则返回异常。该模块处理流程图如图 3.5 所示。

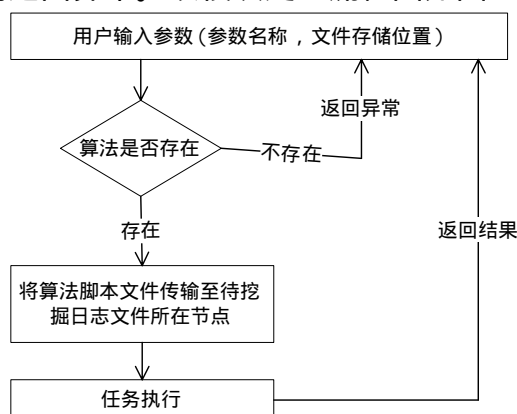


图 3.5 日志挖掘模块处理流程图

3.3.4 结果展示模块

结果展示模块主要给用户提供一个可视化的结果展示平台。它应包括如下几个版块：点击数分析、用户数分析、趋势分析、网页分析、访问行为分析以及流量分析。

(1).概要分析

概要分析主要是对网站的一个整体情况进行统计分析，给出了网站在任意时间段的基本访问情况，如最大访问数、最大访问时段、最大访问地区、点击数最大网页、最常进入网页、最常离开网页等。

(2). 频道分析

频道分析主要是对网站每个频道的访问情况进行分析，如每个频道的访问数、访问时长等。

(3). 趋势分析

趋势分析主要是对网站访问趋势进行分析，如点数趋势分析、用户数趋势分析、停留时间趋势分析等。

(4). 网页分析

网页分析主要是对网站中每个页面的访问情况进行分析，如页面的访问数、每个页面的平均停留时间等。

(5). 访问行为分析

访问行为分析主要是通过分析 web 日志文件来发现站点用户的行为模式，从而发现用户的访问模式、频繁路径等，以便能够更好的调整网站的布局 and 结构，提高客户的满意度。它包括进入网页分析、离开网页分析、访问次数分析、频繁访问路径分析等，然后在此基础之上，结合用户基本信息，通过聚类技术可以划分出具有不同行为特征的客户群，便于企业更好的进行个性化推荐服务。

(6). 流量分析

流量分析主要用于对网站在任意时段内的总的流量情况进行统计，包括流入和流出字节数，给出流量的最大时间段和最小时间段，便于企业合理安排服务器的工作，不造成资源浪费。

3.4 功能层设计

(张娥 等, 2003)认为用户访问日志研究的难点集中在三个方面:(1).如何综合利用客户端、代理服务器端和服务器的日志,对用户行为做适当的假设,从而确定用户个体;(2).如何对日志流进行预处理得到每个用户的访问内容;(3).如何

设计有效的算法处理这些日志文件，挖掘出用户访问规律。本文设计的 web 日志挖掘系统在功能层这一层解决了(2)、(3)两个问题。

功能层是 web 日志挖掘系统的“大脑”，它首先将原始 web 日志文件交由预处理管道进行数据清洗和融合工作；经过预处理的数据被存储在分布式文件系统中，然后通过使用数据挖掘等人工智能技术进行处理，将最符合用户需要的、最有价值的信息发送给用户。该层主要包括三个功能模块：预处理管道、分布式存储、分布式业务处理

3.4.1 预处理管道

在 Web 日志中，并不是所有的原始数据对分析都是必要的。例如我们在挖掘用户浏览偏爱路径时，就只关心访问页面以及参照页面，而代理等信息是我们所不关心的。因此需要对原始数据进行预处理，删除与会话无关的信息以及噪音数据。由中国互联网中心 1 月份发布的《第 27 次中国互联网发展统计报告》可知，截止 2010 年 12 月份，中国的网站数量达到 191 万个，网页数量达到 600 亿个，年增长 78.6%。当访问这些网站时，尤其是大型门户网站时，访问大量页面所产生的 Web 日志数据量也是很庞大的，因此针对海量日志文件的预处理其实也是应当重点研究的问题。本文采用了基于云计算的 MapReduce 机制完成对海量日志文件的并行预处理，达到了很好的效果。

例如，在挖掘用户浏览偏爱路径时，我们设计了如图 3.6 所示的数据预处理管道。从图 3.6 可以看出，该数据处理管道中分别在数据清洗和数据访问频度计算时使用了 MapReduce 编程模型。通过数据清洗，得到<Referer,URL>格式的干净日志文件。Referer 为引用页面，URL 为访问页面。访问频度计算步骤，得到<Referer,URL,m>格式的中间文件，m 为所有日志中，从 Referer 页面访问 URL 页面的次数，即该条路径的访问频度。

本文充分了利用了 MapReduce 模型的高度并行，在 Hadoop 集群框架下实现了如图 3.6 所示的数据处理管道。这个数据管道也可以做成中间件，供其他挖掘系统调用。经过预处理后的日志文件所包含的数据量是原始数据的 1/10-1/4(李军华, 2010)，这为后续的挖掘工作减少了不少工作量。

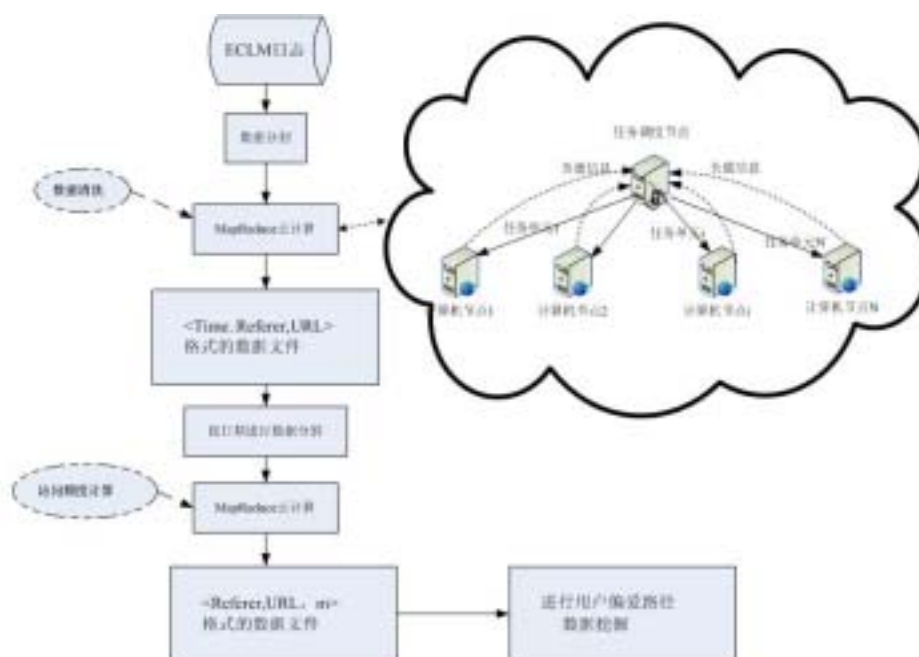


图 3.6 基于云计算的数据处理管道

3.4.2 分布式存储

借助 Hadoop HDFS 的设计思想，将海量数据分布存储在多台计算机集群上，实现高可靠的分布式数据文件存储功能。该层为业务处理层的分布式计算提供输入数据、中间结果以及输出数据，也可以把业务系统的数据直接存储到分布式文件系统进行管理和访问。该层应具备的如下功能：

- (1) 能够将 web 上收集到的日志文件经预处理管道处理后装入分布式存储系统中；
- (2) 能够自动复制日志文件，复制的日志文件被存储在另一个服务节点上，防止由于某个服务节点瘫痪而带来的数据丢失问题；
- (3) 由于在分布式计算环境中，网络带宽是相对稀缺的资源，因此直接让数据保存在构成集群机器的本地磁盘上。当执行计算任务时，大部分数据在本地机器读取，减少网络带宽的占用；
- (4) 提供大量分布式数据集的访问接口；
- (5) 分布式文件系统中新的服务节点加入或有旧的服务节点删除时，能够自动更新。

分布式存储系统负责 web 日志文件的存储和读取，它由一个主节点（NameNode）和多个子节点（DataNode）构成。在实际中，单个存储节点失效的情况是经常存在的，在系统设计时必须将不可信节点的失效屏蔽在系统之内，因此该存储系统使用副本复制存储策略来实现文件系统的高可靠性。分布

式文件系统结构如图 3.7 所示。

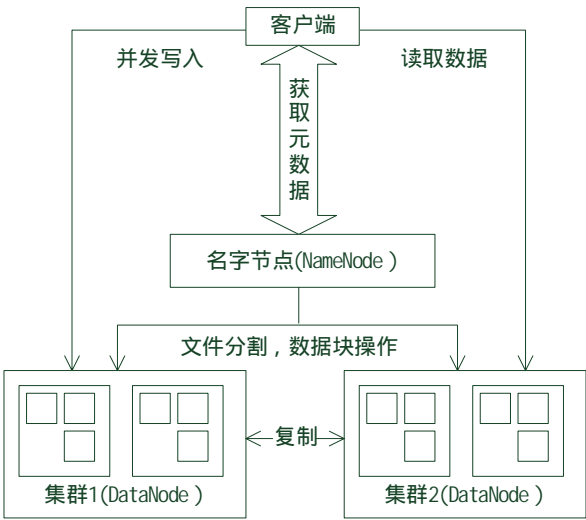


图 3.7 文件存储系统的结构

在这个分布式文件系统中包含一个名字节点(主服务器 NameNode)和多个块服务器(DataNode)。文件被分割成固定尺寸的块，保存在块服务器的本地硬盘上。为了保证可靠性，每个块都会复制到多个服务器上。NameNode 管理文件系统所有的元数据，包括文件信息、每一个文件对应的文件块的信息和每一个文件块在 DataNode 的信息。DataNode 是文件存储的基本单元，它将块存储在本地文件系统中，并保存了块的元数据，同时周期性地将所有存在的块信息发送给 NameNode。客户端跟 NameNode 交互进行元数据操作，但是所有的数据操作的通信都是直接和块服务器进行的。

NameNode 中有两张元数据表，存储着每一个日志文件的元数据以及每一个 DataNode 的元数据，具体设计如表 3.2 和表 3.3 所示。

表 3.2 日志文件的元数据

Id	文件编号
Size	文件大小
Num	分成几个子文件
Ip	存放子文件的 DataNode 的 ip 地址

表 3.3 DataNode 节点的元数据

Ip	DataNode 节点在网络中的 ip 地址
Status	节点当前所处状态（是否能正常工作）
MaxSize	最大存储空间
LeftSize	剩余存储空间
Cp_ip	副本存放位置

通过 NameNode 我们可以对存储在分布式文件系统中的日志文件进行访问和处理。NameNode 还负责管理文件的存储等服务，但实际的数据并不存放 NameNode 上。DataNode 用于实际数据的存放，对 DataNode 上数据的访问并不

通过 NameNode，而是与用户直接建立数据通信。DataNode 每隔一段时间向 NameNode 发送一个信号，以证明该 DataNode 工作正常，没有出现故障。如果 NameNode 没有收到该信号，则表示 DataNode 出现故障，NameNode 则将保存在其他节点上的副本复制到另一个 DataNode 上，始终保持系统中每个日志文件都有 2 个以上，从而保证了系统的高可靠性。

用户写入日志文件的操作过程如下：首先客户端向 NameNode 提交写入请求，NameNode 根据文件大小和文件块配置情况，返回给客户端它所管理部分 DataNode 的信息。客户端将文件划分为多个块，根据 DataNode 的地址信息，按顺序写入到每一个 DataNode 块中。

用户读取文件的操作过程如下：客户端向 NameNode 发起文件读取的请求。NameNode 返回文件存储的 DataNode 的信息。客户端直接从 DataNode 上读取文件信息。

文件块复制过程如下：NameNode 发现部分 DataNode 失效，通知 DataNode 相互复制块。DataNode 开始直接相互复制。

3.4.3 分布式业务处理

该层采用 MapReduce 作为分布式并行计算模型，将大型任务分成很多细粒度的子任务，这些子任务分布式且并行地在多个计算节点上进行调度和计算，从而在云平台上获得对海量数据的处理能力。

分布式业务处理层的核心是任务调度，所有挖掘器统一由 Master 负责调度，一个典型的业务处理执行流程如图 3.8 所示。

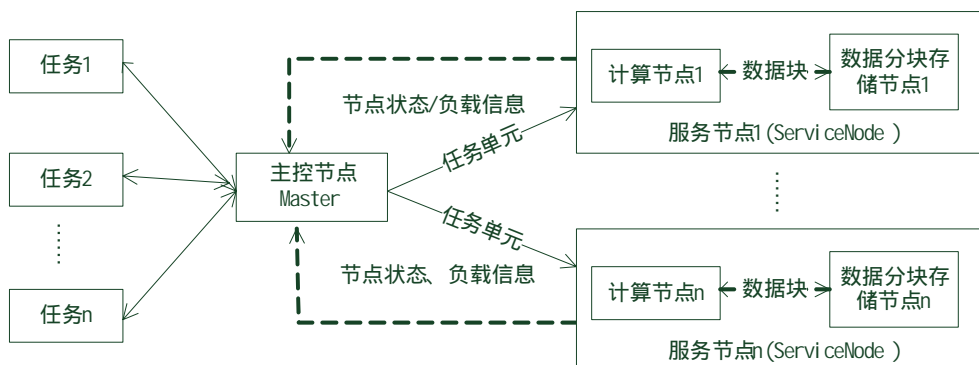


图 3.8 基于 MapReduce 的挖掘任务处理流程

主控节点接收用户的业务申请，向数据存储主节点（NameNode）获得各数据块的存储信息，然后直接通知这些数据块所在节点，挖掘任务立即在数据块存储服务器就地启动计算工作，完成后只向 Master 传送相关结果，并不向主控节点传送文件数据块，主控节点汇总后生成最终的结果返回给用户。这一过程

中没有了文件的传送和重组过程，计算和存储都在一个节点上面，节省了数据传输的时间。

第4章 基于云计算的 web 日志挖掘算法研究

对于一个大型的门户网站，每天访问量会达到几亿人次，产生的 web 日志文件会达到几十个 GB，甚至上百个 GB，因此，不管在数据处理方面还是算法效率方面，单机集中式 web 日志挖掘算法显然是达不到要求的，并行和分布式的算法以它自身所依赖的多处理机平台的优势，在海量数据的 web 日志挖掘中可以发挥重要的作用。依靠集群的处理优势，分布式并行算法将成为未来 web 日志挖掘发展的主流。

目前，可以对 web 日志文件进行挖掘的算法有很多，不同的挖掘目的采用不同的挖掘算法。对用户浏览偏爱路径进行挖掘，发现用户浏览网页的共同行为，从而提供个性化服务，同时对站点的结构调整也有重大意义。本文对 web 日志挖掘的目的就是要找到用户浏览偏爱路径。

目前基于 web 日志的用户浏览偏爱路径挖掘方法有很多。邢东山(2003 a)等从提出的支持-偏爱度的概念出发，给出了一种 web 站点访问的矩阵表示模型；邢东山(2004 b)等依据偏爱度的概念构造浏览偏爱树(PNT)，然后在 PNT 树上挖掘用户浏览偏爱路径；任永功(2008)(2009)等采用兴趣度=浏览次数×浏览时间/接收字节数的方法求解兴趣度，然后将这个兴趣度作为整个挖掘算法的基本分析元素，提出了存储矩阵、会话矩阵和路径矩阵的“三矩阵”模型。

这些算法都存在以下问题：单纯地考虑浏览频度，简单地认为用户的浏览频度就反应了用户的访问兴趣，这是非常不准确的。因为如果挖掘出来的一条用户偏爱路径恰好是网络拓扑图中的一条连续的路径(即页面间由链接连接起来)，那么这条访问路径的频度高显然是因为用户顺着网络链接点击下来的，并没有真实的反映出用户的真正兴趣所在，不是真正有用的知识。

针对以上算法的不足，本章集结合网络拓扑图，引入了有用偏爱度的概念，它削弱了具有直接链接关系的页面之间的影响。在此基础上，我们通过对文献[24]提出的用户浏览偏爱路径挖掘算法——NPPMA 进行改写，给出了一种用户浏览偏爱路径的挖掘算法(Preferred Browsing Paths Mining Algorithm based Useful-preference, PBPU)。最后针对 Hadoop 集群框架中 MapReduce 处理数据的特点，对 PBPU 算法进行了改进，使得改进后的算法能够在 MapReduce 分布式环境中执行，从而提高算法的执行效率。

4.1 基于有用偏爱度的用户浏览偏爱路径挖掘算法

4.1.1 有用偏爱度定义

4.1.1.1 网络拓扑图

一个网站是由大量网页组成的集合，网页是一种特殊的资源，由超链接联系在一起。李颖基(2003)等用有向图表示网络拓扑结构，它极大的反映了设计者的思路。图 4.1 是一个大型门户网站的三级网络拓扑结构。

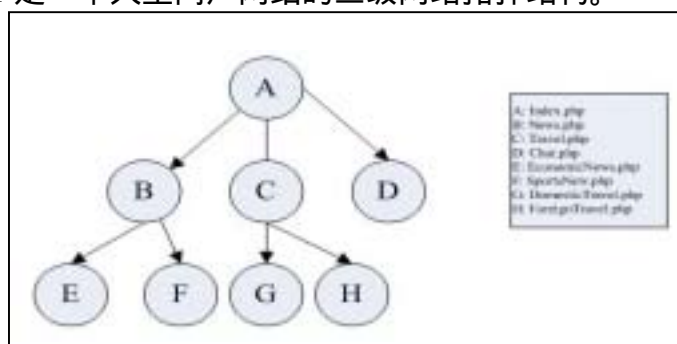


图 4.1 网络拓扑图

由图 4.1 可知，有向图由两种元素构成：结点和有向边。其中结点表示页面，有向边表示页面间的链接关系。用户由某一个页面进入（通常是主页），通过网页上的链接选择访问的下一页面。如果网络拓扑中的两个结点相距较远，表明按照网站设计者的意图这两个结点所代表的页面间的关联性是较低的，若从用户访问日志中发现了它们是一条可信度较高的频繁访问路径，显然用户的感兴趣度是非常高的。根据这个规律，网站设计者就可以在这两个页面之间增添链接或者进行网站拓扑重构，以减少用户的点击次数和等待时间，从而提高用户的满意程度。例如，E 和 G 是相距较远的网页，E 需要经过页面 B、A、C 才能到达 G。如果我们在 web 日志中发现 E→B→A→C→G 是一条用户浏览偏爱路径，则得到的这条频繁路径是有用的，可以帮助网站设计者在 E 页面中添加到 G 页面的链接，从而方便用户，增加了用户的满意度。

4.1.1.2 Web 站点访问矩阵

Web 日志文件是 web 使用挖掘中一个重要的数据来源，用户对网站的访问信息基本都记录在 web 日志文件中。

邢东山(2003)等将预处理完的日志表示为 $L=(Referer, URL)$ 的集合，其中，Referer 表示引用页，URL 表示访问的网页，每个 (Referer, URL) 看成是一个浏览子路径。本文采用访问矩阵+三元组的方式来存放预处理完的日志，其矩

阵如表 4.1 所示，三元组如表 4.2 所示。

表 4.1 网站用户访问矩阵

	NULL	A	B	C	D	E	F	G	H	总和
NULL	0	38	9	1	2	2	0	0	1	53
A	16	0	20	16	28	0	0	0	0	80
B	10	13	0	0	0	20	16	0	0	59
C	12	9	0	0	0	0	0	18	12	51
D	15	20	0	0	0	0	0	0	3	38
E	0	0	16	0	8	0	0	0	0	24
F	0	0	14	0	0	2	0	0	0	16
G	0	0	0	14	0	0	0	0	7	21
H	0	0	0	20	0	0	0	3	0	23
总和	53	80	59	51	38	24	16	21	23	365

表 4.2 网站用户访问矩阵的三元组表示 (部分)

I(请求页面序号)	1(NULL)	1	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4
J(访问页面序号)	2(A)	3	4	5	6	9	1	3	4	5	1	2	6	7	1	2	8	9
S_{ij}	38	9	1	2	2	1	16	20	16	28	10	13	20	16	12	9	18	12

4.1.1.3 有用偏爱度的定义

邢东山(2003)针对用户浏览偏爱路径挖掘提出了支持一偏爱度的概念，其定义分为三步。

采用公式 (4.1) 计算每个访问页面的平均支持度。

$$\overline{S_i} = (\sum_{j=1}^n S_{ij}) / n \quad (4.1)$$

其中， S_j 表示第 j 种选择的支持度，即用户通过第 j 种选择进入下一个页面的频度。

其中第 k ($k=1, 2, \dots, n$) 种选择的选择偏爱度 (preference) 可定义为:

$$P_{ij} = S_{ij} / \overline{S_i} \quad (4.2)$$

支持一偏爱度定义为:

$$\hat{P}_{ij} = S_{ij} \times P_{ij} \quad (4.3)$$

其他的很多论文中也大体采用类似的衡量标准。通过在海量的数据集上进行实验，发现这种针对用户访问兴趣度的衡量标准，在针对大型门户网站的数据挖掘时，会由于没有考虑网站的拓扑结构，而出现极大的误差。

假设图 4.1 为一个门户网站的部分主干三级拓扑结构，其他的略去。A 为网站的首页，B 为其中的新闻频道，C 为旅游频道，D 为交流频道。可以从表 4.1 的访问矩阵看出，在主干中的页面的进出访问率都是远高于其他页面的。比如

(NULL,A)=38,这是因为 A 为首页,大部分会话开始的用户都会首先访问 A 页面; (A,B)=31,因为 B 是主干,访问频率当然高。这样的由于网站拓扑造成的高频数据,虽然也反映了用户的偏爱路径,但是,也造成了奇异数据。这些数据势必会造成按照公式(4.1)-(4.3)计算出来的平均支持度、选择偏爱度、支持一偏爱度的严重偏移,给后面的挖掘过程造成误差。

因此,本文提出有用偏爱度的概念。其计算过程分成以下几步。

(1) 权值网络拓扑图。在传统的网站网络拓扑图的边上引入权值 W_{ij} ,表征该条边的访问路径的有用度。本文认为,网络拓扑的主干路径上的边有用度低,而远离主干路径上的边有用度高。这是因为网络拓扑是一个由链接连接起来的资源集,在网络拓扑中直接或间接相连的资源集(Web 拓扑概率较高)在用户访问时出现重复访问路径的可能性较高,显然,这些特别高频路径对于网络拓扑设计者以及商家是不大感兴趣的。而在拓扑中不相连或相距较远的资源集(Web 拓扑概率较低)在用户访问时重复访问路径的可能性较低,它们的适当高频的路径恰是网络拓扑设计者所期望得到的用于改善用户服务质量和提高网络性能的有关知识,也是商家投放信息的指导。改进后的网络拓扑图如图 4.2 所示。

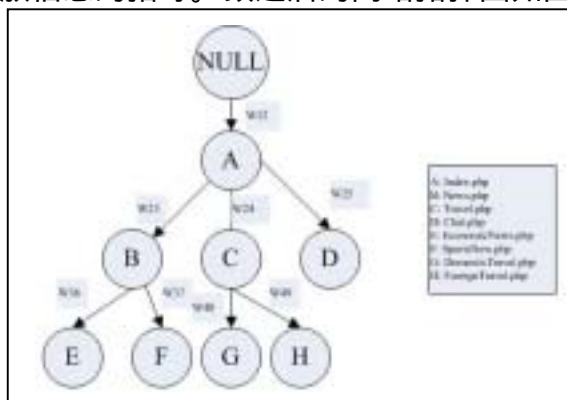


图 4.2 带权值的网络拓扑图

图中的权值可以作为一个参数,通过海量数据集训练得到。本文在这里不做详解,可以根据经验设定。总体来说,越是主干的地方,权值越低,表示该路径的有用度越低。

(1) 有用偏爱度。

为了避免在挖掘用户浏览偏爱路径时,产生主干路径上的高频路径这类不太有趣的知识,我们给网络拓扑中的每条边都引入有用权值,从而平衡主干路径的奇异数据。

本文将平均有用支持度定义为:

$$\overline{S_i} = (\sum_{j=1}^n W_{ij} S_{ij}) / n \quad (4.4)$$

本文将有用偏爱度定义为：

$$P_{ij} = (W_{ij} S_{ij})^2 / \overline{S_i} \quad (4.5)$$

4.1.2 基于有用偏爱度的用户浏览偏爱路径挖掘算法

本文采用的基于有用偏爱度的浏览偏爱路径发现算法如表 4.3 所示。

表 4.3 有用偏爱路径挖掘算法伪代码

算法 1 有用偏爱路径挖掘算法	
1:	输入：Web 站点访问三元组文件 f 有用支持度阈值 P_{lim}
2:	输出：有用偏爱路径集合 Paths 遍历 f 文件，得到三元组集合 F; $U=F$
3:	FOR EACH atom1 IN F //atom1 为一个三元组
4:	$U=F-atom1$
5:	$i=atom1.Referer$ // atom1 的/Referer 的序号,即访问矩阵的行号
6:	$J=atom1.URL$ // atom1 的/URL 的序号，即访问矩阵的列号
7:	$S_i.n=0$ // n 代表以 atom1 中的 Referer 页面出发的边的条数，即访问矩阵第 i 行不为 0 的列数
8:	$S_i.sum=0$ // sum 代表以 atom1 中的 Referer 页面出发的边的总的有用支持度，即访问矩阵第 i 行有用支持度的总和
9:	$S_{ij}=m$ //atom1 的访问频度，即访问矩阵的第 i 行，第 j 列的元素
10:	FOR EACH atom2 IN U
11:	$k=atom2.Referer$ //atom2 的行号
12:	$l=atom2.URL$ //atom2 的列号
13:	IF $k=i$ THEN //同一行的元素
14:	$S_i.n=S_i.n+1$ //该行的非 0 元素个数增 1
15:	$S_i.sum= S_i.sum + W_{kl} * atom2.m$ //改进的有用支持度之和增加
16:	END IF
17:	END FOR
18:	END FOR
19:	FOR EACH S_{ij} IN 访问矩阵
20:	IF $((S_{ij}/(S_i.sum/S_i.n))*S_{ij} \geq P_{lim})$
21:	将路径<i , j>写入偏爱子路径的 2 项集 itemset2 ;
22:	END IF
23:	END FOR
24:	参照邢东山(2003)的方法合并有用偏爱子路径
25:	输出有用偏爱路径集合 Paths

PBPU 算法发现偏爱子路径的时间复杂度为 $O(2(n+1)^2)$ ，合并偏爱子路径的时间复杂度为 $O(2(n+1)^2 \times (n-2))$ 。总的时间复杂度是 $O((n+1)^3)$ ， n 是网站中的 URL 数目。

4.2 基于 MapReduce 的用户偏爱路径挖掘算法

当互联网规模不断扩大，web 网页数量成为海量时，PBPU 算法对 web 日志数据的处理在串行情况下，实现起来需要耗费大量的时间，如果将其改成并行，利用多个处理机并行处理，时间效率将会有很大的提高。因此，本文基于开源云计算基础架构 Hadoop 将 PBPU 算法与 MapReduce 结合，把在集中式环境下独立执行的大型任务分割成若干个可以并行处理的子任务，并交由集群中不同的处理机执行，从而平摊计算与存储消耗。

我们把基于 MapReduce 的 PBPU 算法的计算过程分为两步：第一步是利用 MapReduce 编程架构对 web 日志文件进行预处理。第二步是利用 MapReduce 编程架构实现 PBPU 算法的并行执行。每一步由一个 MapReduce 过程完成，每个 MapReduce 任务被分割成多个子任务，然后分派给集群中各个服务节点并行执行。在并行 PBPU 算法的具体实现过程中，首先将存储在分布式文件系统上的 web 日志文件进行预处理，删除与挖掘无关的字段，利用 MapReduce 编程架构对一条访问路径(Referer, URL)进行频度统计。然后再利用 MapReduce 编程架构修正访问频度，最后将修正后满足用户定义的有用支持度的偏爱子路径合并后输出。

4.2.1 基于 MapReduce 的数据预处理

常见的 web 日志一般有两种格式：CLF 和 ECLM，实际中一般采用 ECLM 日志模式，它的结构如表 2.1 所示(高勋, 2010)。Time/Data 表示 Web 服务器接收到用户请求的时间；URL 表示用户请求访问页面的 URL 地址；Referer 是指引用页 (指向被请求文件的页面) 的 URL，如果用户直接输入 URL 进行访问或利用书签进行访问，则该栏为空；Agent 是指用户浏览所用的操作系统和浏览。

在 Web 日志中，并不是所有的原始数据对分析都是必要的，因此，对 Web 日志在提交给挖掘算法前的数据预处理应该包括数据清洗、频度扫描两步。我们在挖掘用户浏览偏爱路径时，只关心访问页面和参照页面，因此需要对原始数据进行预处理，删除与会话无关的信息以及噪音数据。对于一个大型的门户网站，每天访问量会达到几亿人次，产生的 web 日志文件会达到几十个 GB，甚至上百个 GB，因此针对海量日志文件的预处理其实也是应当重点研究的问题。

本文充分利用 MapReduce 模型的高度并行，建立了如图 3.6 所示的数据处理管道，完成对海量日志文件的并行预处理，达到了很好的效果。

在 mapreduce 框架中，map 方法负责提取 web 日志中每条访问记录中的 Referer 和 URL，并将其作为输出被发送。

表 4.4 Map 阶段的输入与输出

Map 阶段的 Input	Map 阶段的 Output
(记录在文件中的偏移量,记录 1)	(Referer ₁ , URL ₁)
(记录在文件中的偏移量,记录 2)	(Referer ₂ , URL ₂)
.....	
(记录在文件中的偏移量,记录 n)	(Referer _n , URL _n)

Map 函数的输出先由 MapReduce 框架处理，然后再被发送到 reduce 函数。这一过程根据键来对 key/value 进行排序和分组，即把具有相同(Referer, URL)的输出进行合并。其中 m 表示不同(Referer , URL)的个数，n₁、n₂.....n_m 分别代表其对应 (Referer , URL) 的访问频度。表 4.4 给出了 Map 阶段的输入与输出。

表 4.5 Reduce 阶段的输入与输出

Reduce 阶段的 Input	Reduce 阶段的 Output
(Referer ₁ , URL ₁)	(Referer ₁ , (URL ₁ , n ₁))
(Referer ₂ , URL ₂)	(Referer ₂ , (URL ₂ , n ₂))
.....	
(Referer _m , URL _m)	(Referer _m , (URL _m , n _m))

Reduce 将每个 key 的 values 合并后，以 (Referer, (URL, n)) 的形式输出到输出文件，最后将输出文件存入 HDFS 中。表 4.5 给出了 Reduce 阶段的输入与输出。

web 日志文件在经过数据预处理后，只提取了 Referer 和 URL 两个有用字段，而剔除了其余的与挖掘无关的信息，并且对相同记录进行合并，大大压缩了日志文件的数据量。表 4.6 给出了基于 MapReduce 的数据预处理阶段的伪代码。

表 4.6 基于 MapReduce 的数据预处理阶段伪代码

算法 2 基于 MapReduce 的数据预处理阶段伪代码

- 1: 输入：web 日志文件 f
- 2: 输出：(Referer, list(URL , n)) //n 为路径 Referer URL 的访问频度
遍历 f 文件中的每一条记录
//定义 Map 函数
//input key:该条记录在整个文件中的偏移量(以字节为单位) value:每一条记录的实际内容
//Output key: Referer value: list(URL)
- 3: FOR EACH Record IN f
//Record 为 f 中的一条访问记录
- 4: Key=Referer

表 4.6 基于 MapReduce 的数据预处理阶段伪代码(续)

```

5: Value=URL
6: Emit(Key , Value)
7: END FOR
   //定义 Reduce 函数
   //input key:Referer value:list(URL)
   //Output key:Referer values:List(URL, n)
8: List[i]={1} // 数组 List[] 记录每条路径<Referer, URL>的访问频度
9: FOR EACH atom IN list(URL) //atom 为一条 <Refereri, URLi>记录
10: IF URLi 与 URLk 相同(i≠k)
11: List[i]++
12: key=Refereri
13: values=(URLi, list[i])
14: Emit(key, values )
15: END IF
16: END FOR

```

4.2.2 基于 MapReduce 的 PBPU 算法

根据前述基于单机集中式环境的 PBPU 算法的执行流程，MapReduce 并行计算的过程是集群中的主控节点把经过预处理管道处理后存储在 HDFS 中的记录页面间链接关系的文件分成多个文件块(inputSplit[]), 然后分派给集群中的运行 Map 任务的节点。该节点拿到存储链接关系文件的一个文件分块后，将块中的每一条记录转化成一定格式的<key, value>对输出至主控节点。运行 Reduce 任务的结点接着读取 Map 任务输出的中间结果，根据输入的 key 值对中间结果进行汇总，最终以一定格式的<key, value>对保存到 HDFS 中。

从 MapReduce 并行计算任务执行的特点，它的每一次计算任务的输入输出都要访问 HDFS，因此可通过减少 MapReduce 计算过程的个数，从而避免多次从 HDFS 中读取数据，减少网络传输量，占用少量网络带宽，也可以降低算法执行时间。

基于 MapReduce 的 PBPU 算法只执行一次 MapReduce 操作，用来计算经过有用偏爱度修正的访问频度值 n' 。在该 MapReduce 阶段，Map 方法从 HDFS 中读取 4.2.1 节中设计的数据预处理阶段 Reduce 的输出，作为这一阶段的输入。表 4.7 给出了 PBPU 算法在 Map 阶段的输入与输出。

表 4.7 PBPU 算法在 map 阶段的输入与输出

Input	key: Referer _i value: list[(URL _i , n _i)]
Output	key: list[(Referer _i , URL _i)] value: list[(n _i , W _{r ui})]

Reduce 方法读入 Map 的输出列表，根据 key 值进行分组，不同的组分到不同的 Reduce 上进行合并操作。表 4.8 给出了 PBPU 算法在 Reduce 阶段的输入与

输出。

表 4.8 PBPU 算法在 reduce 阶段的输入与输出

Input	key: list[(Referer _i , URL _i)]	value: list[(n _i , W _{r_ui})]
Output	key: list[(Referer _i , URL _i)]	value: list[p*n _i ² /n _i *W _{r_ui}]

表 4.9 给出了基于 MapReduce 的 PBPU 算法在求解偏爱子路径时的伪代码。

表 4.9 基于 MapReduce 的 PBPU 算法伪代码

算法 3 基于 MapReduce 的 PBPU 算法生成偏爱子路径阶段的伪代码

```

1: 输入：从 HDFS 中读取数据预处理阶段 Reduce 的输出文件 f
    有用支持度阈值 Plim
2: 输出：修正后的有用支持度 n' 满足有用支持度阈值 Plim 的 list(<Referer, URL>, n') // n' 为路径 Referer URL 基于有用偏爱度修正的访问频度
    遍历 f 文件中的每一条记录
    //定义 Map 函数
    //input key: Refereri value: list([URLi, ni])
    //Output key: list[(Refereri, URLi)] value: list[(ni, Wr_ui)]
3: FOR EACH Record IN f
    //Record 为 f 中的一条访问记录
4: Key= (Refereri, URLi)
5: Value= list[(ni, Wr_ui)]
6: Emit(Key, Value)
7: END FOR
    //定义 Reduce 函数
    //input key: Refereri value: list([URLi, ni])
    //Output key: list[(Refereri, URLi)] values: List[p*ni2/∑ni*wr_ui]
8: Total=0// 数组 List[] 记录每条路径<Referer, URL>的访问频度
9: m=0//统计具有相同 Refereri 的(Refereri, URLi)个数
10: FOR EACH URLi IN list(<Referer, URL>)
11:     m=m+1
12: total=total+ni*wr_ui // wr_ui 是页面 Refereri 和 URLi 之间的权值
13: END FOR
14: avg=total/m
15: FOR EACH URLi IN list[(Referer, URL)]
16: n'=n2/avg
17: END FOR
18: key=(Referer, URL)
19: Value=n'
20: Emit(key, value)

```

第 5 章 实验分析

本文将采用 Hadoop 平台作为第 4 章提出的 PBPU 算法的开发和调试环境。本章主要介绍了基于 Eclipse 的开发环境和实验环境的搭建，并对基于 MapReduce 的 PBPU 算法与基于单机运行的 PBPU 算法的实验结果进行了对比分析，同时还将 PBPU 算法与邢东山(2003)等提出的 NPPMA 算法进行了对比分析。

5.1 系统开发环境

实验运行在由 5 台电脑组成的集群上。机器名分别为 master、slave1、slave2、slave3、slave4。Master 是主控节点，作为 NameNode 和 JobTracker。其余四台即作为计算节点 TaskNode 又作为数据存储节点 DataNode。5 台电脑均安装 Ubuntu9.04 操作系统、hadoop-0.20.0、sun-java6-jdk 以及 ssh。5 台电脑的 IP 地址分别为：192.168.10.1、192.168.10.2、192.168.10.3、192.168.10.3、192.168.10.4、192.168.10.5。本平台是在 eclipse 集成开发环境中进行程序开发的。

5.1.1 系统开发平台

Hadoop 安装包的 contrib/eclipse-plugin 目录下有一个插件 hadoop-0.20.0-eclipse-plugin.jar。使用这个插件，我们可以配置基于 eclipse 的 hadoop 应用开发环境。基于这个插件，可以在 eclipse 中创建 MapReduce 应用程序，并且提供基于 MapReduce 框架类开发的向导，打包成 JAR 文件包，部署一个 MapReduce 应用程序到一个 hadoop 服务器，大大简化了 hadoop 并行编程的过程。安装 hadoop 的 eclipse plugin 过程很简单，就只需要将 Hadoop 安装包的 contrib/eclipse-plugin 目录下的 eclipse 插件复制到 eclipse 的 plugins 目录下即可，命令如下。

```
$ cd /opt/hadoop-0.20.0
$ sudo cp
/opt/hadoop-0.20.0/contrib/eclipse-plugin/hadoop-0.20.0-eclipse-plugin.jar/opt/eclipse/plugins
```

然后重新启动 eclipse，这样就可以在 eclipse 环境中建立一个 Hadoop 应用程

序了。工作界面如图 5.1 所示。

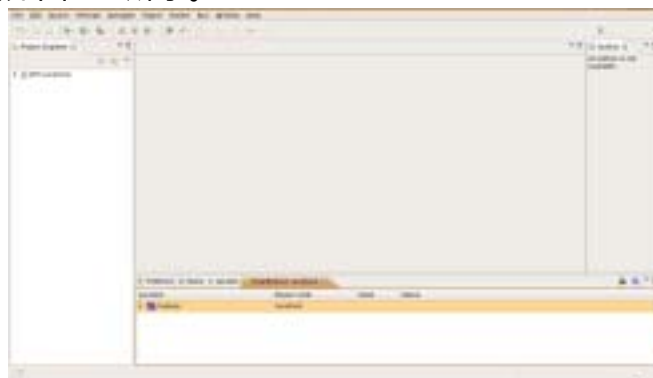


图 5.1 基于 eclipse 的 Hadoop 开发环境

5.1.2 Hadoop 环境的搭建

Hadoop 集群支持三种运行模式：单机模式、伪分布式模式和完全分布式模式。单机模式的 Hadoop 被配置成一个以非分布式模式运行的独立 Java 进程，它没有使用分布式文件系统，因此适合对程序的功能进行测试。本论文搭建的 Hadoop 环境采用完全分布式运行模式。

5.1.2.1 单节点 Hadoop 环境的搭建

伪分布式运行方式也是在一台机器上运行，但是启用不同的 Java 进程模拟分布式运行中各类节点。

(1).Hadoop 配置

修改 conf 目录下的配置文件 `hadoop - env.sh`.修改如下:

```
export JAVA_HOME=/usr/lib/jvm/java-6-sun
```

```
export HADOOP_CLASSPATH=/opt/hadoop
```

```
export PATH=$PATH:/opt/hadoop/bin
```

修改 conf 目录下的配置文件 `core-site.xml`，修改如下:

```
<property>
```

```
  <name>fs.default.name</name>
```

```
  <value>hdfs://localhost:9000</value>
```

```
</property>
```

```
<property>
```

```
  <name>mapred.job.tracker</name>
```

```
  <value>hdfs://localhost:9001</value>
```

```
</property>
```

修改 conf 目录下的 `hdfs-site.xml` 文件，修改如下:

```
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
```

修改 conf 目录下的 mapred-site.xml 文件，修改如下：

```
<property>
  <name>mapred.job.tracker</name>
  <value>localhost:9001</value>
</property>
```

(2).免密码 SSH 设置

首先安装 SSH，然后生成密钥对，并将生成的密钥保存在/home/ms/.ssh/id_rsa 文件中，最后设置密码为空。命令如下所示。

```
$ sudo apt-get install ssh
$ cp id_rsa.pub authorized_keys
$ ssh-keygen-t rsa
```

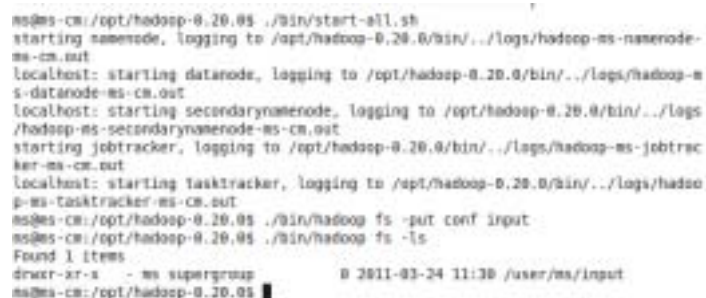
(3).HDFS 文件系统格式化

```
$ ./bin/hadoop namenode -format
```

(4).开始和结束守护进程

```
$ ./bin/start-all.sh
$ ./bin/hadoop fs -put conf input
$ ./bin/hadoop fs -ls
```

结果如图 5.2，则表示操作无误。



```
ms@ms-cm:/opt/hadoop-0.20.0$ ./bin/start-all.sh
starting namenode, logging to /opt/hadoop-0.20.0/bin/../logs/hadoop-ms-namenode-
ms-cm.out
localhost: starting datanode, logging to /opt/hadoop-0.20.0/bin/../logs/hadoop-
ms-datanode-ms-cm.out
localhost: starting secondarynamenode, logging to /opt/hadoop-0.20.0/bin/../logs
/hadoop-ms-secondarynamenode-ms-cm.out
starting jobtracker, logging to /opt/hadoop-0.20.0/bin/../logs/hadoop-ms-jobtrac
ker-ms-cm.out
localhost: starting tasktracker, logging to /opt/hadoop-0.20.0/bin/../logs/hadoo
p-ms-tasktracker-ms-cm.out
ms@ms-cm:/opt/hadoop-0.20.0$ ./bin/hadoop fs -put conf input
ms@ms-cm:/opt/hadoop-0.20.0$ ./bin/hadoop fs -ls
Found 1 items
drwxr-xr-x - ms supergroup 0 2011-03-24 11:30 /user/ms/input
ms@ms-cm:/opt/hadoop-0.20.0$
```

图 5.2 SSH 配置最终结果图

成功执行以上步骤后，将会在机器上启动 NameNode、DataNode、JobTracker、TaskTracker 和 Secondary NameNode 五个新的 Java 进程。访问 <http://localhost:50070> 可以查看 NameNode 以及整个分布式文件系统的状态，浏览分布式文件系统中的文件以及日志等；访问 <http://localhost:50030> 可以查看 JobTracker 的运行状态；访问 <http://localhost:50060> 可以查看 TaskTracker 的运行状态。



图 5.3 NameNode 节点状态

按照上述步骤在五台电脑分别做一次，则集群中每台机器都实现了单机 Hadoop 伪分布式计算平台。

5.1.2.2 Hadoop 集群环境的搭建

该集群环境由五台机器组成。它们的 ip 地址分别为：192.168.10.1、192.168.10.2、192.168.10.3、192.168.10.4、192.168.10.5。使用 192.168.10.1 这台机器作为 NameNode 和 JobTracker 节点，其余四台机器作为 DataNode 和 TaskTracker 节点。首先在所有的机器上建立目录/home/cloud 作为 hadoop 的安装路径。安装好 hadoop 后，需做如下配置：

(1). 配置 NameNode 和 DataNode

配置成功的关键在于确保各机器的主机名和 ip 地址之间能正确解析。因此修改 5 台机器./etc 目录下的 hosts 文件。在作为 NameNode 机器上的 hosts 文件中加入如下代码：

```
192.168.10.1 master
192.168.10.2 slave1
192.168.10.3 slave2
192.168.10.4 slave3
192.168.10.5 slave4
```

其余作为 DataNode 机器上的 hosts 文件中就只需添加自己本机的 ip 地址和 NameNode 的 ip 地址。

(2).配置 SSH

这一步主要是为了实现机器之间执行指令时不需要输入密码。首先在 5 台机器上都建立.ssh 目录(\$ mkdir .ssh)；接着在 master 上生成密钥对(\$ ssh -keygen -t rsa)；master 生成密钥后，接着执行如下命令：

```
$ cd ~/.ssh
$ cp id_rsa.pub authorized_keys
```

```
$ scp authorized_keys slave1:/home/cloud/.ssh
```

```
$ scp authorized_keys slave2:/home/ cloud /.ssh
```

```
$ scp authorized_keys slave3:/home/ cloud /.ssh
```

```
$ scp authorized_keys slave4:/home/ cloud /.ssh
```

最后进入所有机器的.ssh 目录，改变 authorized_keys 文件的许可权限：

```
$ chmod 644 authorized_keys
```

这时从 master 向其他机器发起 SSH 连接，只有在第一次登录时需要输入密码，以后则不需要。

(3). 在所有机器上配置 Hadoop

首先配置 master。修改 conf 目录下的 core-site.xml 文件

```
<property>
```

```
<name>fs.default.name</name>      //NameNode 节点的主机 IP 地址和端口
```

```
<value>hdfs://192.168.10.1:9000</value>
```

```
</property>
```

```
<property>
```

```
<name>hadoop.tmp.dir</name>      //Hadoop 默认的临时路径
```

```
<value>/home/cloud/hadoop-0.20.0/tmp</value>
```

```
</property>
```

```
<property>
```

```
<name>mapred.job.tracker</name> //JobTracker 节点的主机 IP 地址和端口
```

```
<value>hdfs://192.168.10.1</value>
```

```
</property>
```

```
<property>
```

```
<name>dfs.replication</name>     //文件备份数
```

```
<value>1</value>
```

```
</property>
```

修改 conf 目录下的 master 文件，输入主机名 master。修改 conf 目录下的 slave 文件，输入 slaves 的主机名，即 slaves1、slave2、slave3、slave4。

把 Hadoop 安装文件复制到其他机器上。编辑所有机器 conf 文件夹下的 hadoop-env.sh 文件，将 JAVA_HOME 变量设置为各自 JAVA 安装的根目录。

(4).运行 Hadoop

在 master 节点上格式化分布式文件系统：`$ bin/hadoop/ NameNode -format`

在 master 上启动 NameNode、JobTracker 和 Secondary NameNode，在 slave1、slave2、slave3、slave4 上启动 DataNode 和 TaskTracker：`$ bin/start-all.sh`

此时，访问 <http://master:50070> 可以查看 NameNode 以及整个分布式文件系统的状态，浏览分布式文件系统中的文件以及日志等

5.2 基于 MapReduce 的 PBPU 算法在 Hadoop 上的实现

在 Hadoop 集群架构下开发基于 MapReduce 的 PBPU 算法的步骤如下：

(1) 启动 eclipse，打开 Window->Open Perspective->other..->map/reduce，打开一个 Map/Reduce 视野。

(2) 打开 window->show view->other.., 选则 Map/Reduce Tools, 点击 Map/Reduce Locations，会打开一个 view, 添加 Hadoop location，其中 Host 和 Port 的内容根据 conf/core-site 的配置填写。如图所示 5.4。



图 5.4 eclipse 环境配置-1

(3) 建立一个 MapReduce 专案

file->new->project->Map/Reduce->Map/Reduce Project->next

在 new MapReduce Project Wizard 向导框的 Project name 中输入 web - log - mining(随意)，在 use default hadoop 中输入 /opt/hadoop-0.20.0，点击 finish。这时在 eclipse 窗口的左侧就出现了专案 web - log - mining。右击 web - log - mining，进行属性详细配置。

(4) 专案属性详细配置。

在 properties for web - log - mining 窗口，配置 Java Build Path->Libraries，添加 JAR：hadoop-0.20.0-ant.jar、hadoop-0.20.0-core.jar、hadoop-0.20.0-tools.jar。

以 hadoop-0.20.0-ant.jar 为例：在 Source attachment 处输入 /opt/hadoop-0.20.0/src，在 Javadoc location 处输入：file:/opt/hadoop-0.20.0/docs/api/。配置完成后如图 5.5 所示。

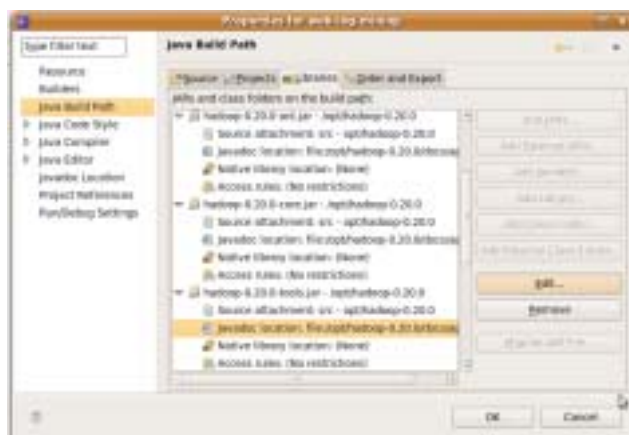


图 5.5 eclipse 环境配置-2

(5) 连接 Hadoop server

进入 New Hadoop location 向导，进行 eclipse 与 hadoop 间的配置。

Location Name -> 输入：hadoop (随意)

Map/Reduce Master -> Host-> 輸入 : localhost

Map/Reduce Master -> Port-> 输入：9001

DFS Master -> Host-> 输入：9000

Finish

(6) 编写程序

首先创建 mapper.java

file->new->mapper,输入 Name : mapper

在 mapper.java.输入如下代码.

```
package Sample;

import java.io.IOException;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.OutputCollector;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapred.Reporter;

public class mapper extends Mapper<LongWritable, Text, Text, Text> {

    private Text word=new Text();

    private Text referer = new Text();

    public void map(LongWritable key, Text value, OutputCollector<Text,
```

```

Text> output, Reporter reporter)

    throws IOException, InterruptedException {
    String[] url=new String[2];
    String line=value.toString();
    int i=0;
    Pattern p=Pattern.compile("https?:\\/[*\\s]+");
    Matcher m=p.matcher(line);
    while (m.find()) {
        url[i]=m.group();
        i++;
    }
    word.set(url[0]);
    referer.set(url[1]);
    output.collect(referer, word);
}
}

```

接着创建 reduce.java

file->new->reducer, 输入 name:reducer

在 reducer.java.编写如下代码:

```

package Sample;

import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapred.Reporter;

public class reducer extends Reducer {

    public void reduce(Iterator<Text[]> key, IntWritable values,
        OutputCollector<Text[], IntWritable> output, Reporter reporter)
    throws IOException {
        IntWritable count=new IntWritable(0);
        Text[] word=new Text[2];
    }
}

```

```

    int i=0;
    word=key.next();
    while (key.hasNext()) {
        i=i+1;
        count.set(i);
    }
    output.collect(word, count);
}

```

最后创建主函数(main function)urlsplit.java,它用来驱动 mapper 和 reducer。

File->MapReducer Driver , 在弹出的 New MapReduce Driver 向导中输入 name : urlsplit

在 urlsplit.java 中输入如下代码 :

```

package Sample;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

public class urlsplit {

    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        String[] otherArgs = new GenericOptionsParser(conf, args)
            .getRemainingArgs();

        if (otherArgs.length != 2) {
            System.err.println("Usage: urlsplit <in> <out>");
            System.exit(2);
        }

        Job job = new Job(conf, "url split");
        job.setJarByClass(urlsplit.class);
        job.setMapperClass(mapper.class);
        job.setCombinerClass(reducer.class);
    }
}

```

```

    job.setReducerClass(reducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

(7) 程序测试

首先产生 Makefile 文件：gedit Makefile

在该文件中输入如下内容：

```
JarFile="sample-0.1.jar"
```

```
MainFunc="Sample.WordCount"
```

```
LocalOutDir="/tmp/output"
```

```
all:help
```

```
jar:
```

```
jar -cvf ${JarFile} -C bin/ .
```

```
run:
```

```
hadoop jar ${JarFile} ${MainFunc} input output
```

```
clean:
```

```
hadoop fs -rmr output
```

```
output:
```

```
rm -rf ${LocalOutDir}
```

```
hadoop fs -get output ${LocalOutDir}
```

```
gedit ${LocalOutDir}/part-r-00000 &
```

```
help:
```

```
@echo "Usage:"
```

```
@echo " make jar      - Build Jar File."
```

```
@echo " make clean    - Clean up Output directory on HDFS."
```

```
@echo " make run       - Run your MapReduce code on Hadoop."
```

```
@echo " make output    - Download and show output file"
```

```
@echo " make help      - Show Makefile options."
@echo " "
@echo "Example:"
@echo " make jar; make run; make output; make clean"
```

(8) 结果展示

我们选取一婚纱摄影公司网站的某一天日志文件作为样本数据,该日志文件大小为 11370KB,包含 34140 条访问记录。将该日志文件作为上述 MapReduce 过程的输入,执行 Map 和 Reduce 操作后,得到输出文件 part-r-00000,部分结果如图 5.6 所示。

```
- index.html 6859
index.html hunsha.html 554
index.html xiezhen.html 356
index.html kepian.html 678
index.html aboutus.html 113
index.html ~/bbs/index.html 4346
- ~/bbs/index.html 3242
~/bbs/index.html ~/bbs/misc.php?action=nav 4544
~/bbs/index.html ~/bbs/forumdisplay.php?fid=15 6229
~/bbs/index.html ~/bbs/forumdisplay.php?fid=4 2348
~/bbs/index.html ~/bbs/forumdisplay.php?fid=6 871
~/bbs/index.html ~/bbs/forumdisplay.php?fid=8 214
~/bbs/index.html ~/bbs/forumdisplay.php?fid=10 112
~/bbs/index.html ~/bbs/forumdisplay.php?fid=13 95
~/bbs/misc.php?action=nav ~/bbs/forumdisplay.php?fid=15 1450
~/bbs/misc.php?action=nav ~/bbs/forumdisplay.php?fid=4 1021
~/bbs/misc.php?action=nav ~/bbs/forumdisplay.php?fid=6 897
~/bbs/misc.php?action=nav ~/bbs/forumdisplay.php?fid=8 147
~/bbs/misc.php?action=nav ~/bbs/forumdisplay.php?fid=10 46
~/bbs/misc.php?action=nav ~/bbs/forumdisplay.php?fid=13 17
```

图 5.6 基于 MapReduce 的数据预处理结果展示(部分)

5.3 实验结果分析

5.3.1 PBPU 算法准确性分析

为了验证本文提出的 PBPU 算法在挖掘用户浏览偏爱路径时的准确性,我们设计了两组实验将本文改进的算法与邢东山(2003)提出的 NPPMA 算法进行了准确性比较分析。比较方法是:在第一个实验中,我们让两算法在一定的阈值下挖掘相同数量的用户浏览偏爱路径,与已知的网站经验兴趣路径比较,得到准确性度量。结果如图 5.7 所示。从图中可以看出,本文提出的 PBPU 算法的准确率总体上好于 NPPMA 算法。且随着挖掘出的路径数增多,NPPMA 算法的准确性急剧降低,而本文提出的 PBPU 算法趋于稳定。这是由于要挖掘出的路径

个数越多，相关的兴趣度度量阈值越低，挖掘出的兴趣路径的可信程度降低，但本文提出的算法在一开始就考虑到了兴趣路径的可信程度，因此准确率不会发生明显改变。

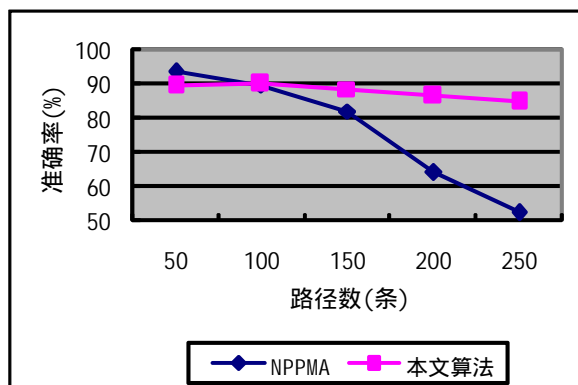


图 5.7 算法准确性比较-1

在第二个实验中，我们让两算法在一定阈值下分别对有 10 个网页、50 个网页和 100 个网页的 3 个网站进行用户浏览偏爱路径挖掘，与已知的网站经验路径比较，得到准确性度量。结果如图 5.8。从图中可以看出，网站中的网页数量较少时，PBPU 算法的准确率不如 NPPMA 算法，其原因可能是由于网页数量较少，网站结构对网页间的影响较小，凭经验进行网页间的权值设定反而不如实际来得真实；但是随着网站中网页数目的增多，网站结构变得复杂，它对偏爱路径挖掘的影响就变大，因此 PBPU 算法在网页数目较多时，能更好地找出真实的用户浏览偏爱路径。

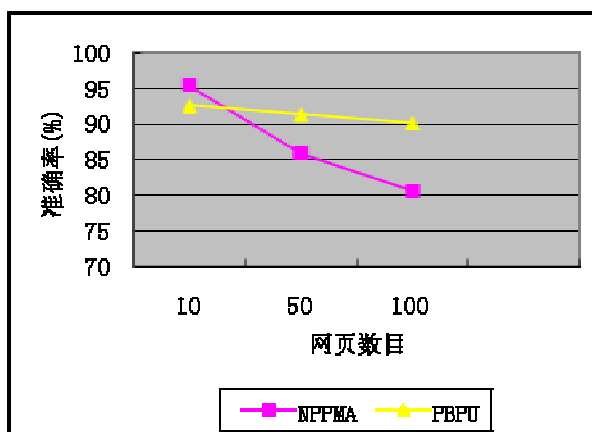


图 5.8 算法准确性比较-2

由此可见，我们提出的基于云计算的用户浏览偏爱路径算法 PBPU 优于邢东山(2003)等提出的 NPPMA 算法，尤其使用于大型门户网站用户浏览偏爱路径的挖掘。

5.3.2 基于 MapReduce 的 PBPU 算法执行效率分析

为了分析基于 MapReduce 的 PBPU 算法和传统集中式环境的 PBPU 算法的性能，本文分别设计了两个实验环境：一个是基于单机的集中式环境，一个是基于集群的分布式环境。一个用于测试单机环境的性能，另一组用于测试云计算平台的性能。

数据来源于某公司门户网站服务器上下载了五个大小不同的日志文件，分别在单机环境、有3个节点的分布式集群环境和有5个节点的分布式集群环境中执行 PBPU 算法，其数据处理时间如图 5.9 所示。从测试结果可以看出，当数据量较小时，在单机环境中执行 PBPU 算法优于并行环境，随着日志文件的增大，并行环境的性能将明显优于单机环境，且日志文件越大，两者的处理时间差距也越来越大。

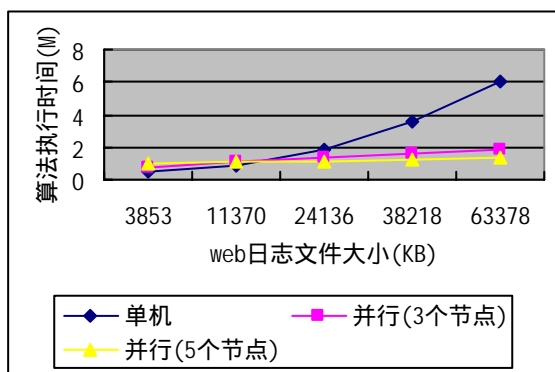


图 5.9 单机环境和云计算并行环境中数据处理时间比较

分析基于 MapReduce 的 PBPU 算法的整个执行过程，可以知道基于 MapReduce 的 PBPU 算法的时间消耗主要在 Hadoop 开启阶段、Hadoop 运行阶段以及主/从节点之间通信。在日志文件较小时，并行环境中虽然将日志文件分割成更小的子文件供不同节点处理，使得子文件的处理时间小于整个文件的处理时间，但减少的这点时间消耗并不能弥补在并行环境中开启、运行 Hadoop 以及节点之间通信所带来的时间消耗，导致并行处理总时间反而大于单机执行的时间。随着日志文件的增大，虽然在数据传输及开启、运行 Hadoop 时仍然会花费额外的时间，但这点增加的时间远远小于由于将文件并行处理所节省的时间，使得并行处理总时间小于单机处理的时间，且随着文件的增大，并行处理在时间节省上的优势越明显。

为了更好的衡量基于 MapReduce 的 PBPU 算法并行处理的性能和效果，我们引入评价指标——加速比。式 5.1 给出了加速比的计算公式。

$$S_p = T_1 / S_p \quad (5.1)$$

由公式(5.1), 我们分别计算了 3 个节点和 5 个节点并行处理的加速比, 结果如图所示。从图可以知道在日志文件较小时, 3 个节点并行环境的处理时间小于 5 个节点的并行环境, 这是因为节点数目增加, 必然将导致节点之间的通信开销增大, 导致加速比较低。随着日志文件的增大, 节点间的通信开销远远小于对大文件处理的时间开销, 导致加速比明显提高。

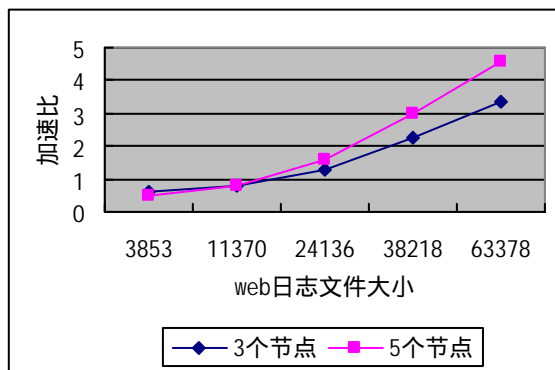


图 5.10 3 个节点和 5 个节点并行处理加速比比较

综合图 5.9 和图 5.10 可以知道随着参与计算的节点数目的增加, 计算时间也会下降, 它与集群中的节点数量成反比。

为了分析基于 MapReduce 的数据预处理和普通的并行机制的性能, 我们又设计了一组集群用于测试基于 MPI 的并行的性能。MPI 的并行实验同样运行在由上述 5 台 PC 机组成的集群上。每台 PC 机还需安装 mpich 1.2.4。

然后, 我们在两个集群中分别对上述 5 个大小不同的 web 日志数据进行预处理, 其数据处理时间如图 5.11 所示。从图中可以看出, 两种集群环境对数据的处理时间相差不大, 在数据量较小时, MPI 集群略优于云计算集群环境, 但当数据量增大时, 由于 MPI 会开启大量进程, 占用大量内存, 导致对数据的处理时间高于云计算。

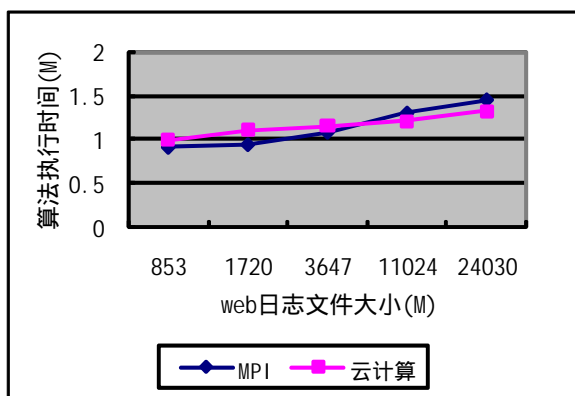


图 5.11 两种集群环境中数据处理时间比较

通过实验可以知道, PBP 算法在并行执行时的时间消耗主要是: 各节点建立连接、计算任务分配、HDFS 中文件的读取与写入、map 以及 reduce 执行时

间、中间结果以及最终结果在各节点之间的传输时间。同时,影响并行处理系统性能的主要因素有集群中节点的数目以及 web 日志文件的大小。如果节点数目太多或日志文件太小,这会增加各个节点之间协作的代价,增加通信开销,降低系统的性能;如果节点数目太少或日志文件太大,这又会导致每个文件分块过大,不能够最大化的进行并行处理。因此该系统中的节点数量应根据不同的 web 日志大小而有所不同。

第6章 总结与展望

6.1 本文工作总结

本文基于 Hadoop 集群框架下的 MapReduce 编程模型，提出了基于云计算的 web 日志挖掘平台的设计方案。主要工作如下：

(1) 对云计算相关技术和 web 日志挖掘做了一个简单的概述，分析了现有的 web 日志挖掘系统的缺点，利用云计算技术的优点，提出了基于云计算的 web 日志挖掘系统方案。

(2) 对基于云计算的 web 日志挖掘系统的总体架构进行设计，将该系统分为三个层次：可视化交互层、功能层以及硬件资源层，并详细地阐述了可视化交互层和功能层的结构及功能。

(3) 针对目前基于 web 日志的用户浏览偏爱路径挖掘算法注重访问频度而忽略用户真实兴趣度的问题，结合网站拓扑结构修正基于频度的用户浏览偏爱路径的衡量标准，提出了有用偏爱度的概念，从而剔除了页面放置和链接等因素对挖掘的影响，并给出了基于有用偏爱度的用户浏览偏爱路径算法。

(4) 搭建了 Hadoop 分布式云计算平台，并利用该平台对本文提出的用以挖掘用户浏览偏爱路径的改进算法的准确性与任务执行高效性进行了实验的对比分析。结果表明，改进的算法在进行大型网站的偏爱路径挖掘时，准确率高且趋于稳定，不会随着偏爱路径的增多以及网页数目的增多而导致准确率降低。同时，在云计算环境中进行 web 日志挖掘，通过“云”中多个资源完成原先由一个节点承担的挖掘工作，使资源得到了充分的利用，提高了数据处理的效率。

6.2 展望

由于论文受到时间和个人能力的限制，只能将基于云计算的 web 日志挖掘系统的各个模块进行设计，而功能却没有实现，同时没有开发出更多的能够用于分布式并行环境挖掘的算法。对本文提出的用以挖掘用户浏览偏爱路径的改进算法也没有给出一个很好的权值设定方法，使得改进的算法在实际应用时更加准确。

因此，接下来还有以下工作可以继续：

(1) 在系统设计的基础上，实现系统各个功能模块的建设。

(2) 对影响网页间权值的因素进行研究，给出一个能准确衡量网页间权值的方法。

(3) 对一些经典的数据挖掘算法进行并行改进，使得改进的算法能够在分布式环境中并行处理。

参考文献

- Bing Liu 著, 愈勇, 薛贵荣, 韩定一译. web 数据挖掘[M]. 北京-清华大学出版社.
2009:322-343
- 高勋. 基于云计算的 web 结构挖掘算法研究[D]. 北京交通大学, 2010
- 郭本俊, 王鹏, 陈高云, 黄健. 基于 MPI 的云计算模型[J]. 计算机工程. 2009, 24(30):84-86
- 何清. 第二届中国云计算大会[EB/OL].(2010-05-21)[2010-6-30]
- 纪俊. 一种基于云计算的数据挖掘平台架构涉及与实现[D]. 青岛大学, 2009
- 李军华. 云计算及若干数据挖掘算法的 MapReduce 研究[D]. 电子科技大学, 2010
- 李雪锋. 基于云计算环境的 web 数据挖掘算法研究[D]. 北京交通大学, 2010
- 李颖基, 彭宏, 郑启伦, 曾炜. Web 日志中有趣关联规则的发现[J]. 计算机研究与发展. 2003,
40(3): 435-439
- 刘华元, 袁琴琴, 王保保. 并行数据挖掘算法综述[J]. 电子科技. 2006, 1:65-68
- 刘鹏. 云计算[M]. 北京-电子工业出版社. 2010:186-215
- 刘业政, 李亚飞, 杨善林. 电子商务环境下基于移动 Agent 的 web 数据挖掘.计算机工程[J].
2004, 10(30):107-108
- 齐玉成, 郑丽英, 高三营. 基于网格的数据挖掘算法[J].电脑知识与技术.2010,2(06):871-874
- 任晓霞. 一种 web 日志数据挖掘系统的设计与实现[D]. 北京邮电大学,2008
- 任永功, 付玉, 张亮, 吕君义. 一种新的基于 Web 日志的挖掘用户浏览偏爱路径的方法[J].
计算机科学. 2008, 35(10):192-196
- 任永功, 付玉, 张亮. 一种改进的用户浏览偏爱路径挖掘方法[J]. 计算机工程. 2009,
35(8):47-49
- 唐卫宁, 耿国华. 电子商务中基于 CORBA 的 Web 数据挖掘研究[J].计算机应用研
究,2002,(7):121-123
- 万至臻. 基于 MapReduce 模型的并行计算平台的设计与实现[D]. 浙江大学, 2008
- 王鄂, 李铭. 云计算下的海量数据挖掘研究[J]. 现代计算机. 2009.11(39): 22-25
- 王庆波, 金津, 何乐等著. 虚拟化与云计算[M]. 北京-电子工业出版社. 2010:166-180
- 邢东山, 沈钧毅, 宋擒豹. 从 web 日志中挖掘用户浏览偏爱路径[J]. 计算机学报. 2003,
11(26):1518-1523
- 邢东山, 沈钧毅. 一个可以准确反映 web 浏览兴趣的度量值--偏爱度[J]. 控制与决策. 2004,
19(3):307-310
- 徐超. 云计算技术在中国农村信息化建设中的应用[D]. 山东大学, 2010

- 姚洪波, 杨炳儒. Web 日志挖掘数据预处理过程技术研究 [J]. 微计算机信息(管控一体化), 2006, 22(6-3): 234-236
- 张娥, 冯耕中, 郑斐峰. Web 用户访问日志数据挖掘研究[J]. 情报杂志. 2003, (09): 48-50
- 郑晶. 基于网格的并行数据挖掘算法的实现[J]. 福建工程学院学报. 2010.2(08):83-86
- 周斌, 吴泉源, 高洪奎. 用户访问模式数据挖掘的模型与算法研究. 计算机研究与发展[J]. 1999, 36(7):870-875
- B.Sarwar, G. Karypis, J. Konstan and J. Riedl. Application of Dimensionality Reduction in Recommender Systems-A Case Study. In Proc. of the KDD Workshop on WebKDD'2000, 2000
- Cappos,J; Beschatnikh,I; Krishnamurthy,A; Anderson,T. Seattle: a platform for educational cloud computing[J]. SIGCSE Bulletin 41 1 111-15 March 2009
- K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, M. Tsugawa. Science Clouds: Early Experiences in Cloud Computing for Scientific Applications[C]. Cloud Computing and Its Applications 2008 (CCA-08), Chicago, IL. October 2008
- M Cannataro, D Talia, P Trunfio. KNOWLEDGE GRID: High Performance Knowledge Discovery on the Grid. Lecture Notes In Computer Science, Vol. 2242, Proceedings of the Second International Workshop on Grid Computing, 2001, pp. 38-50
- Massimo Gaggero, Simone Leo, Simone Manca. Parallelizing bioinformatics applications with MapReduce.CCA-08: Cloud Computing and its Applications, Chicago, Illinois.2008
- Michael C. Schatz. Cloudburst: highly sensitive read mapping with MapReduce[J].BIOINFORMATICS 25 11 1363-1369 JUN 1 2009
- Q.C.Chen, L.Q.Wang. MRGIS: A MapReduce-Enabled High Performance Workflow System for GIS[OL]. www.chinacloud.cn/show.aspx?id=1949&cid=28
- Robert Grossman, Yunhong Gu. Data Mining Using High Performance Data Clouds: Experimental Studies Using Sector and Sphere[C]. Proceeding KDD '08 proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. New York, 2008
- Yunhong Gu, Robert L. Grossman. Sector and Sphere: the design and implementation of a high-performance data cloud[C]. UK e-Science All Hands Meeting, 2008

致谢

三年的研究生生活转眼即逝，在这期间，我学到了很多知识，结识了很多对我重要的老师、同学。首先，我要感谢我的导师——陈华平教授。在研究生三年的学习中，陈老师渊博的学识和严谨的科研态度深深的感染了我，让我受益匪浅。同时，在学习和研究过程中，陈老师给了我精心的指导，帮助我克服了学习上的许多困难。本论文也是在陈老师悉心的指导下完成的，从选题到完成，每一步都倾注了导师大量的心血。在此，谨向导师表示崇高的敬意和衷心的感谢！

感谢实验室的各位兄弟姐妹们，通过与他们的讨论与交流，很好的解决了学习过程中遇到的各种问题。正是由于他们的帮助和支持，我才能克服一个又一个困难和疑惑，直至本论文的顺利完成。

感谢三年来，伴随我成长的室友、同学，与他们相处的三年充满了快乐的回忆，让我度过了人生中最宝贵的阶段。

同时，我要感谢我的父母，是他们多年来的关心、支持和鼓励，才使得我在困境面前能够继续前进，不断取得新成绩。

最后，谨向百忙之中抽出宝贵时间评审本论文的各位专家致以最诚挚的谢意！

2011 年 5 月

在读期间发表的学术论文与取得的其他研究成果

已发表论文：

- [1] 程苗. 基于云计算的经济预测系统研究. 激光杂志[J],2011(2)：

待发表论文：

- [1] 程苗, 陈华平. 基于 Hadoop 的 web 日志挖掘. 计算机工程,2011(11)
[2] 程苗, 陈华平. 基于云计算的用户浏览偏爱路径挖掘算法. 计算机工程与应用,2011.
[3] 程苗, 陈华平. 基于云计算的 web 数据挖掘. 计算机科学,2011.

云计算技术在web日志挖掘中的应用研究

作者：[程苗](#)

学位授予单位：[中国科学技术大学](#)

本文读者也读过(1条)

1. [林振立](#) [云计算环境下的内存资源共享技术研究](#)[学位论文]2010

本文链接：http://d.g.wanfangdata.com.cn/Thesis_D141819.aspx