**调研报告**

1. **项目名称：FastDFS的去中心化和负载均衡优化**
2. **项目背景**

**随着互联网的高速发展，信息技术的日益普及，人们处理的数据量正在成千上万的增长，对数据的计算能力和存储能力的要求不断提高，海量数据的时代已经来临。而作为这些数据信息的载体，web的易用性催生出了电子商务、社交网络、在线旅游等垂直性网站。根据中国互联网络中心发布的《第41次中国互联网络发展状况统计报告》【2】显示，截止至2017年12月，我国网民数已达7.72亿，相较于2016年增幅为4074万。该报告显示，手机终端的网络音乐、网络视频、网络游戏等网民规模相较2016年分别增长了9.36%、9.73%、15.77%，保持了较高的增长率。另外，以微博为代表的社交网站用户数也得到了迅速增长，2017年12月底，新浪微博活跃用户数已达3.76亿，网民中微博使用率达到了40.9%。无论是电子商务网站图片的展示，或社交网站图片的分享，图片数据量都达到了指数级别的增长。**

**以国内外互联网巨头来说，截至2014年底，Facebook用户上传的图片就已经达到了2500亿张，平均每天上传3.5亿张，在高峰期Facebook每秒能够处理55张照片。国外最大的图片分享网站Flickr则存储了接近6亿张图片，而且绝大多数图片还是高清的，单张图片大小一般达到了4-5M，公共占据了大约2.5PB存储空间，每秒的处理请求次数为38000次，每天新增的图片超过50万张。淘宝作为我国最大的电子商务网站，在线商品数量达到了10亿级别，图片存储服务器存储数量更是达到了290亿张图片，总量在1PB以上，而且每天仍在数以千万计的增长。因为图片传达信息的能力远大于文字描述，所以电子商务网站一般都特别注重上传时间，图片的质量和访问速度等问题。而根据淘宝的流量分析，淘宝所有的通信量中传输量更是达到了90%。我国另一社交公司巨头腾讯公司用户上传的图片更是达到了600多亿张，每人每天上传的图片为10张，图片的峰值访问次数更是达到了每秒50万次。**

**当今互联网就是数据为王的时代，谁拥有海量数据，谁就能吸引到用户。因此，网站设计之初，就规划好了数据采集的来源，其中主要的数据贡献来自于互联网用户。绝大部分数据是图片，大约占总数据量的80%。由于图像的数据量非常大，大量的图片需要大量的存储空间，而且网站必须快速响应用户的访问请求，因此这对存储系统的容错能力、可扩展性和高并发访问都提出了很高的要求。**

**同时，互联网信息是由位置定位的：一个特定的内容包含一个URL——统一资源定位器（Universal Resource Locator），它包含到一个网络服务器的IP地址，该服务器在一个特定的位置上承载内容。例如前往一个URL，如https://facebook.com/mypicture.jpg，把域名(facebook.com)解析到一个IP地址(如31.13.70.36)并在那里查找照片。URL的指令是“转到31.13.70.36，请求路径/mypicture.jpg上的内容”。URL总是指向那个位置。即使该图片保存在另一个网站上，甚至是你自己的电脑上，你仍然需要连接到那个IP地址并下载它的内容。**

**在日常的人类语境中思考这个问题是很有用的。想象一下，我们只能通过一个副本的物理位置来引用书籍，而不是通过标题、作者或ISBN（国际标准书号）——仅仅是物体的位置。所以，如果有人推荐你一本书，他们会说，“嘿，你应该读这本伟大的书，它在纽约公共图书馆，第九区，第三个书柜，最顶层的那个书架，从左边数第一本。”现在你需要去那儿检查一下，然后才能得到一份拷贝。在你去到之前，你不知道它实际上是什么，你只知道它的位置。这显然是非常低效的。这也是一种不稳定的情况——如果有人移动了这本书怎么办？如果那天图书馆闭馆怎么办？或者图书馆彻底停业了怎么办？或者假使你到了第三个书柜，最顶层的那个书架，拿到了从左边数第一本书，而就在这时你意识到其实你的背包里就有一本同样的书，那又怎么办？**

**这不仅仅是一个理论上的问题。哈佛大学最近的一项研究发现，在美国最高法院意见书中，有49%引用的超链接不再有效。这些法院意见书指向的地址在过去的某段时间内都保有正确的内容，但在那个位置的内容现在已经不存在了。这是脆弱、低效、不必要的，这是一种很愚蠢的行为。**

**相反，我们应该考虑一种不同的方式，考虑用“是什么”来定位信息，而不是“在哪里”。要做到这一点，我们需要使用一种不同的网络链接。我们不需要使用指向位置的链接，而是需要可以唯一描述内容本身的链接，就像指纹一样。这种内容寻址方法将“什么”和“哪里”分隔开，这样数据就能通过网络进行传输，因此就可以在任何地方为任何人所存储和服务。**

**近几年来，随着互联网技术越来越成熟、网络带宽不断提升、开源氛围越来越浓，人们对分布式文件系统的研究也越来越深入，目前为止已经出现了一批以Linux为基础的分布式文件系统，以支持高可靠、高性能、高并发的应用需求。在谷歌发布Google File System（GFS）的技术文档之后，相继出现了许多类GFS的分布式文件系统，如FastDFS、MogileFS、HDFS、TFS等。这些文件系统的目标是一致的，都是为了实现高可用性、可伸缩性和高容错性的分布式文件系统需求。然而在实现方式上有较明显的差异，因此形成了各自的技术特点。**

**在实际应用中，FastDFS以其独特的设计理念，获得了越来越多人的青睐。为了适应互联网应用的注重高可用、高性能、可拓展等指标，FastDFS充分考虑了冗余备份、负载均衡、线性扩展等机制，是为互联网应用量身定做的分布式文件系统。FastDFS轻量级的系统架构、特有的分组方式和对等结构，是它与现有的类GFS的重大区别。正是因为这些特点，FastDFS特别适用于互联网中以文件为载体的在线服务，如相册网站、视频网站、网盘等互联网应用**。

**分布式系统的发展：**

**第一代分布式文件系统（1980年代）**

**早期的分布式文档系统一般以提供标准接口的远程文档访问为目的，更多地关注访问的性能和数据的可靠性，以NFS和AFS(Andrew File System)最具代表性，他们对以后的文档系统设计也具备十分重要的影响。**

**NFS从1985年出现至今，已经历了四个版本的更新，被移植到了几乎任何主流的操作系统中，成为分布式文档系统事实上的标准。NFS利用Unix系统中的虚拟文档系统（Virtual File System，VFS）机制，将客户机对文档系统的请求，通过规范的文档访问协议和远程过程调用，转发到服务器端进行处理；服务器端在VFS之上，通过本地文档系统完成文档的处理，实现了全局的分布式文档系统。Sun公司公开了NFS的实施规范，互连网工程任务组（The Internet Engineering Task Force，IETF）将其列为征求意见稿（RFC-Request for Comments），这很大程度上促使NFS的很多设计实现方法成为标准，也促进了NFS的流行。NFS不断发展，在第四版中提供了基于租赁（Lease）的同步锁和基于会话（Session）语义的一致性等。**

**Carnegie Mellon大学在1983年设计研发的AFS将分布式文档系统的可扩展性放在了设计和实现的首要位置，并且着重考虑了在不安全的网络中实现安全访问的需求。因此，他在位置透明、用户迁移、和已有系统的兼容性等方面进行了特别设计。AFS具备很好的扩展性，能够很容易地支持数百个节点，甚至数千个节点的分布式环境。同时，在大规模的分布式文档系统中，AFS利用本地存储作为分布式文档的缓存，在远程文档无法访问时，依然能够部分工作，提高了系统可用性。后来的Coda File System、Inter-mezzo File System都受到AFS的影响，更加注重文档系统的高可用性（High Availability）和安全性，特别是Coda，在支持移动计算方面做了很多的研究工作。**

**早期的分布式文档系统一般以提供标准接口的远程文档访问为目的，在受网络环境、本地磁盘、处理器速度等方面限制的情况下，更多地关注访问的性能和数据的可靠性。AFS在系统结构方面进行了有意义的探索。他们所采用的协议和相关技术，为后来的分布式文档系统设计提供了很多借鉴。**

**第二代分布式文档系统（1990～1995）**

**20世纪90年代初，面对广域网和大容量存储应用的需求，借鉴当时先进的高性能对称多处理器的设计思想，加利福尼亚大学设计研发的xFS，克服了以前的分布式文档系统一般都运行在局域网（LAN）上的弱点，很好地解决了在广域网上进行缓存，以减少网络流量的难题。他所采用的多层次结构很好地利用了文档系统的局部访问的特性，无效写回（Invalidation-based Write Back）缓存一致性协议，减少了网络负载。对本地主机和本地存储空间的有效利用，使他具备较好的性能。**

**Tiger Shark并行文档系统是针对大规模实时多媒体应用设计的。他采用了多种技术策略确保多媒体传输的实时性和稳定性：采用资源预留和优化的调度手段，确保数据实时访问性能；通过加大文档系统数据块的大小，最大限度地发挥磁盘的传输效率；通过将大文档分片存储在多个存储设备中，取得尽量大的并行吞吐率；通过复制文档系统元数据和文档数据，克服单点故障，提高系统可用性。**

**基于虚拟共享磁盘Petal的Frangipani分布式文档系统，采用了一种新颖的系统结构—分层次的存储系统。Petal提供一个能够全局统一访问的磁盘空间。Frangipani基于Petal的特性提供文档系统的服务。这种分层结构使两者的设计实现都得到了简化。在Frangipani中，每个客户端也是文档系统服务器，参和文档系统的管理，能够平等地访问Petal提供的虚拟磁盘系统，并通过分布式锁实现同步访问控制。分层结构使系统具备很好的扩展性，能够在线动态地添加存储设备，增加新用户、备份等，同时系统具备很好的机制来处理节点失效、网络失效等故障，提高了系统的可用性。**

**Slice File System（SFS）考虑标准的NFS在容量、性能方面存在的限制，采用在客户机和服务器之间架设一个μproxy中间转发器，以提高性能和可扩展性。他将客户端的访问分为小文档、元数据服务、大文档数据三类请求。通过μproxy将前两种请求转发到不同的文档服务器上，将后者直接发送到存储服务器上。这样SFS系统就能够支持多个存储服务器，提高整个系统的容量和性能。μproxy根据请求内容的转发是静态的，对于整个系统中负载的变化难以做出及时反应。**

**第三代分布式文档系统（1995～2000）**

**网络技术的发展和普及应用极大地推动了网络存储技术的发展，基于光纤通道的SAN、NAS得到了广泛应用。这也推动了分布式文档系统的研究。 在这个阶段，电脑技术和网络技术有了突飞猛进的发展，单位存储的成本大幅降低。而数据总线带宽、磁盘速度的增长无法满足应用对数据带宽的需求，存储子系统成为电脑系统发展的瓶颈。这个阶段，出现了多种体系结构，充分利用了网络技术。**

**出现了多种分布式文档系统体系结构，如Global File System（GFS）、General Parallel File System （GPFS）、惠普的DiFFS、SGI公司的CXFS、EMC的HighRoad、Sun的qFS、XNFS等。**

**数据容量、性能和共享的需求使得这一时期的分布式文档系统管理的系统规模更大、系统更复杂，对物理设备的直接访问、磁盘布局和检索效率的优化、元数据的集中管理等都反映了对性能和容量的追求。规模的扩展使得系统的动态性，如在线增减设备、缓存的一致性、系统可靠性的需求逐渐增强，更多的先进技术应用到系统实现中，如分布式锁、缓存管理技术、SoftUpdates技术、文档级的负载平衡等。**

**第四代分布式文档系统（2000年以后）**

**随着SAN和NAS两种结构逐渐成熟，研究人员开始考虑如何将两种结构结合起来。网格的研究成果等也推动了分布式文档系统体系结构的发展。**

**随着SAN和NAS两种体系结构逐渐成熟，研究人员开始考虑如何将两种体系结构结合起来，以充分利用两者的优势。另一方面，基于多种分布式文档系统的研究成果，人们对体系结构的认识不断深入，网格的研究成果等也推动了分布式文档系统体系结构的发展。这一时期，IBM的StorageTank、Cluster的Lustre、Panasas的PanFS、蓝鲸文档系统（BWFS）等是这种体系结构的代表。各种应用对存储系统提出了更多的需求：**

**大容量：现在的数据量比以前任何时期更多，生成的速度更快；**

**高性能：数据访问需要更高的带宽；**

**高可用性：不但要确保数据的高可用性，还要确保服务的高可用性；**

**可扩展性：应用在不断变化，系统规模也在不断变化，这就需要系统提供很好的扩展性，并在容量、性能、管理等方面都能适应应用的变化；**

**可管理性：随着数据量的飞速增长，存储的规模越来越庞大，存储系统本身也越来越复杂，这给系统的管理、运行带来了很高的维护成本；**

**按需服务：能够按照应用需求的不同提供不同的服务，如不同的应用、不同的客户端环境、不同的性能等。**

**处于这个阶段的系统都在研究中，但从中也能够看出一些发展趋势：体系结构的研究逐渐成熟，表现在不同文档系统的体系结构趋于一致；系统设计的策略基本一致，如采用专用服务器方式等；每个系统在设计的细节上各自采用了很多特有的先进技术，也都取得了很好的性能和扩展性。另外，在协议方面的探索也是研究的热点之一，如Direct Access File System利用了远程内存直接访问的特性，借鉴了NFS第四版本和Common Internet File System等协议，设计了一套新的网络文档访问协议。**

1. **立项依据**

**一、负载均衡算法对分布式文件系统的高伸缩性和高容错性起到至关重要的作用，在日益增加的数据量面前，设计一个适用的负载均衡算法显得尤为重要。负载均衡调度的主要依据是各个服务器自身的实时负载和当前任务所带来的负载量，同时还要考虑策略本身对服务器造成的额外开销。目前使用较多的经典负载均衡算法根据调度策略的不同可以分成静态和动态两种。静态负载均衡算法实现简单、使用方便、占用较少的系统资源，但是没有能够实时的反映出节点服务器的真实负载情况。动态负载均衡算法虽然能够有效的反映出节点服务器的真实负载状况，但其实现较复杂、占用较大的系统资源、节点间通信开销较大。用最小的开销换取节点服务器的最真实负载状态，是负载均衡算法的理想化状态，采用何种方式让服务器用较小的代价换来节点服务器较真实的负载，作为节点服务器负载分配的依据，让服务器能够长时间运行且不出现负载倾斜的现象，正是负载均衡技术研究的方向所在。**

**然而，随着互联网的蓬勃发展，需要的存储空间越来越多，并发量越来越大，FastDFS原有的静态负载均衡策略难以适应高并发下的线性扩容。目前的FastDFS发布的各个版本的负载均衡策略，均采用的是静态的负载均衡策略，在扩容时会因为没有考虑节点的实时负载，使存储服务器在一段时间内出现负载倾斜的情况，为了适应数据量急剧增加的互联网应用，寻找更适合的负载均衡策略迫在眉睫。**

**二、此处应介绍中心化系统的缺点和去中心化系统的独特优势。**

1. **前瞻性/重要性分析**

**在长远的发展来看，文件系统使用的集群会变得非常庞大，这个集群它不再是某个大型数据中心，也可能会是很多个数据中心的集合体，而这些数据中心也会分布在不同的区域范围内。未来集群不能再是外部数据中心那么简单唯一，甚至最终有可能的是通过互联网来建立“唯一”的一个集群。而当下使用一台服务器是不行的，会阻碍发展的需求。如果集群的范围变小，结果将会是集群特别不稳定，影响用户对数据的需求，遏制了集群的快速发展。所以分布式文件系统的发展前景很广阔。在大规模使用分布式文件系统后，它的安全性、性能等因素将会是研究的重点。故我们对以FastDFS为代表的分布式文件系统的负载均衡和去中心化研究很有意义。**

1. **相关工作（科研及工业界）**

**一、负载均衡**

**数十年以来，国内外学者在分布式系统的负载均衡方面做了很多研究并且取得了一些理论成果，研究人员在负载均衡算法方面也提出了一些可行的负载均衡算法，大多数的算法主要针对负载均衡的三个方面：信息的表示与收集、任务迁移和定位查询。**

**文献【3】描述的是一种加权轮询的负载均衡机制，主要应用于大型网络分布式中，它考虑的是节点的综合负载能力，所以能更好的评价节点的负载。但缺点是由于用队列来存储节点信息，不能保证过载服务器找到的空闲服务器是最佳的，也就是说它不能保证任务迁移能够立即执行。另一个缺点是它考虑的负载指标较少，反映不出最新的负载情况。**

**分布式存储系统分为文件级存储系统和块级存储系统。Li Zhou【4】等人基于块级存储系统提出了一种基于负载和性能等因子的策略方法，该方法能够有效的解决负载均衡问题。Lalitha【5】等人认为一般的计算机基础设施很难实现负载均衡和资源管理的功能。Olga【6】认为在大规模并行计算中，负载均衡对整个系统的性能有着至关重要的影响，他们曾经提出过一种负载均衡模型，即用精确可用的负载度量方法，一定程度地抽象出应用程序信息，并且具体提出了成本模型来调整负载不均衡的状态。Li Yang【7】等提出的另一种负载均衡策略被广泛应用于数字气田的分布式系统中。在这种负载均衡算法中，他们没有采用静态策略，而是采用动态分布的策略，那么类似的应用程序也可以运行在多服务器上，这样一来服务器的性能将会有很大提升。**

**在大规模系统中经常出现调度瓶颈的问题，Hendrickson和Devine【8】总结出一些适合集中式调度的应用。这些应用中包括了并行蒙特卡洛计算（Parallel Monte Carlo Calculations）、光线追踪【9】和可视化。集中式调度的最大弊端是：较重的负载会使得主处理器出现调度瓶颈，因此不适合较大的应用系统。而分布式调度策略则允许处理器可以自己执行负载均衡。Chengzhong和Lau【10】描述了一种最近邻负载均衡方法，即处理器的操作仅仅与它直接邻居的工作负载情况相关，这就减轻了调度器的压力。**

**而在分布式存储系统的应用方面，国内外比较著名的在线图片分享网站Flickr以及社交网站Facebook、Twitter等每天都会有大量的图片要处理，而且他们的用户数以亿计，因此更需要整体架构的健壮、底层存储系统的可扩展和高性能。**

**FastDFS是作者余庆根据MogileFS的设计思想改进而来的类GFS“键值对”的轻量级分布式文件系统，通过专有API来提供文件的存储和访问。FastDFS提供的API可以实现基本的文件存储和文件管理功能，是分布式存储的一种廉价的解决方案。2008年4月开始研发FastDFS，到2008年7月份发布V1.0版，到现在的最新版V5.04，已经相继发布更新了70多个版本。目前，已经有多家公司使用FastDFS来搭建存储平台及应用，如UC、支付宝、飞信、赶集网、迅雷等。**

**张慧芳【11】研究并提出了一种加权最小连接数算法，主要应用于服务器集群的负载调度，它考虑的是把服务器当前的连接数和该服务器的负载权值两者糅合起来，作为负载的衡量标准，所以能很好的评价节点的负载。**

**童瑞霞【12】研究的基于动态反馈的负载均衡算法，主要是将服务器节点的CPU和内存利用率作为衡量节点负载的标准，所以该算法能够更加精确的评价节点的负载。**

**Bin Dong【13】等人提出一种动态自适应的负载均衡算法，该算法完全基于分布式架构，自适应的采集服务器节点的负载来减少信息交换带来的负载消耗，根据采集到的负载信息，对节点进行负载的分配，有效的克服了动态负载均衡代价大的缺点。**

**魏钦磊【14】提出一种基于负载权值的动态反馈负载均衡算法，对服务器节点的性能和负载进行定量的衡量，通过这些值了来计算负载权值和分配概率，以此作为负载分配的依据。**

**韩增曦【15】通过将FastDFS负载均衡的考虑因子由总存储空间变为存储空间效率、**

**连接数和性能三个因子，对负载均衡算法进行改进。**

**二、去中心化**

**1、IPFS**

**星际文件系统IPFS是一种协议，它允许我们使用那些内容寻址链接来交换数据。在IPFS中，文件或数据的哈希指纹就是它的地址。我们使用这些指纹来识别内容，而不是使用服务器的物理位置。这样，当你试图加载一个文件时，你可以从任何位置检索它。如果文件已经在你自己的计算机上，你可以从那里检索它。如果你在网络中的直接邻居有这个文件，你可以从他们那里检索到。你可能从原始服务器，从网络中的其他人，或者任何人那里获取它。IPFS可以使用高效的路由选择算法来搜索网络，并且它可以为隐私进行调整：例如只向你信任的节点请求数据。这类似于其他P2P系统所做的事情，但却被极大地扩展了。你可以使用它来与任意数量的节点交换任何类型的文件或数据，并直接构建到网络中。**

**IPFS通过基于“是什么”而不是”在哪里“来寻址信息，使得网络去中心化。这种去中心化的模式允许在本地网络中工作的网络应用程序脱离原始来源，它可以是一个失去上行链路的办公室的聊天室，一本在各种图书馆都保存的科学期刊，一个网络较差的偏远村庄的维基百科，或是一场危机中的家庭对话短信。它加强了我们的数字信息，使数据能够适应底层互联网的故障，并以密码学来保护它，让它永久存在。你，或是那些分享你的数据的人可以保存信息的副本，并且数年都在相同的链接上。**

**2、FastDHT**

**FastDHT 是一个高性能的分布式哈希系统 (DHT) ,使用 Berkeley DB 做数据存储,使用libevent做网络IO处理,提供 Java 版的客户端接口包｡适合用来存储用户在线､会话等小数据量信息｡**

**FastDHT存储Key Value Pair支持两种存储方式:缓存方式的MPOOL和持久存储方式的BDB｡Key包括三部分:Namespace,ObjectID和Key｡ Key可设置过期时间,自动清除过期数据。Server端划分group,同group数据互相备份,并且可自动压缩binlog.服务端可使用单线程,多线程模式｡**

**FastDHT一些特性:**

**虚拟farm,便于扩容;**

**分布式算法client端实现,不需要中心服务器;**

**二进制通信协议,支持Proxy;**

**使用libevent,异步IO方式,支持大并发;**

**自动failover;**

**支持长连接｡**

**FastDHT集群由一个或多个组(group)组成,每个组由一台或多台服务器组成,同组服务器上存储的数据是相同的,数据同步只在同组的服务器之间进行｡组内各个服务器是对等的,对数据进行存取时,可以根据key的hash值来决定使用哪台服务器｡数据同步采用推(Push)的方式,由源服务器主动将数据同步到本组的其他服务器｡FastDHT集群由一个或多个组(group)组成,每个组由一台或多台服务器组成,同组服务器上存储的数据是相同的,数据同步只在同组的服务器之间进行｡组内各个服务器是对等的,对数据进行存取时,可以根据key的hash值来决定使用哪台服务器｡数据同步采用推(Push)的方式,由源服务器主动将数据同步到本组的其他服务器。**

1. **参考文献**
2. **FastDFS分布式文件系统负载均衡算法的改进研究\_熊建波**
3. **第41次中国互联网络发展状况统计报告**

**3、王友良，叶柏龙 分布式系统中动态负载均衡的研究【J】 科学技术与工程 2005(9)**

**4、Zhou L,Wang Y C,Zhang J L.et al.Optimize block-level cloud storage system with load-balance strategy【J】.Parallel and Distributed Processing Symposium Workshops & phD forum,2012:2162-2167**

**5、BINDU P L H,VENKATESAN R,RAMALAKSHMI K. Perspective study on resource level load balancing in grid computing environments【C】. The 3rd International Conference on Electronics Computer Technology. IEEE,2011:321-325.**

**6、PEARCE O,GAMBIN T,SUPINSKI B R.et al.Quatifying the effectiveness of load balance algorithms【C】. Proceedings of the 26th ACM International Conference on Supercomputing,2012:185-194.**

**7、YANG L,ZHANG J,GEN X,et al.Research on dynamic load balance model of distributed storage system in digital gas fields【C】. The 3rd International Conference on Computer Research and Development.IEEE,2011:264-267.**

**8、HENDRICKSON B,DECINE K. Dynamic load balancing in computational mechanics【J】. Methods Appl,2000,184(2-4):485-500.**

**9、GUSTAFSON J,BENNER R,SEARS M.etc.A radar simulation program for a 1024-processor hypercube【C】. In the Proceedings of the Supercomputing’ 98,ACM/IEEEs,Reno,12-17 November,1989:96-105.**

**10、XU C,LAU F.Load Balancing in Parallel Computers: Thoery and Practice【M】.the Netherlands:Kluwer Academic publishers,1996.**

**11、张慧芳 基于动态反馈的加权最小连接数服务器负载均衡算法研究【D】 华东理工大学 2013**

**12、童瑞霞 基于动态反馈机制的集群负载均衡算法研究【D】 武汉理工大学 2011**

**13、Bin Dong,Xiuqiao Li,Qimeng Wu,Limin Xiao,Li Ruan. A dynamic and adaptive load balancing strategy for parallel file system with large-scale I/O servers【J】 Journal of Parallel and Distributed Computing,2012,7210.**

**14、魏钦磊 基于集群的动态反馈负载均衡算法的研究【D】 重庆大学，2013**

**15、韩増曦 分布式文件系统FastDFS的研究和应用【D】 大连理工大学，2014**