一种基于自配置策略的新型 Peer to Peer 平台系统*

黄维雄[1], 黄铭钧[1], 陈建利[1], 王晓宇[2], 凌 波[2], 周傲英[2+]

[1](新加坡国立大学 计算机科学系,新加坡)

[2](复旦大学 计算机科学与工程系,上海 200433)

A Novel Peer-to-Peer System Based on Self-Configuration

Ng Wee-Siong[1], Ooi Beng-Chin[1], Tan Kian-Lee[1], WANG Xiao-Yu[2], LING Bo[2], ZHOU Ao-Ying[2+]

[1](Department of Computer Science, National University of Singapore, Singapore)

[2](Department of Computer Science and Engineering, Fudan University, Shanghai 200433, China)

+ Corresponding author: Phn: 86-21-65643024, Fax: 86-21-65643503, E-mail: ayzhou@fudan.edu.cn

http://www.fudan.edu.cn

Received 2002-05-09; Accepted 2002-08-14

Ng WS, Ooi BC, Tan KL, Wang XY, Ling B, Zhou AY. A novel peer-to-peer system based on self-configuration. *Journal of Software*, 2003,14(2):237~246.

Abstract: One of the most important features for a peer-to-peer (P2P) distributed system is to share information and services among nodes with equivalent capabilities and responsibilities by pooling their resources together. Nowadays, most of the existing P2P systems such as Napster and Gnutella only provide some kinds of coarse granularity information sharing without taking account of the content of the file. And, typically nodes (peers) are defined statically. In addition, there is no mechanism to support the join of nodes with temporary network addresses. In this paper, a prototyping P2P system, BestPeer is presented. The BestPeer is unique in several ways. Firstly, it combines the power of mobile agents into P2P systems to perform operations at peers' sites. Secondly, it is self-configurable. A node can dynamically select the set of peers with which it can communicate directly based on some optimization criterion. Thirdly, the BestPeer provides a location independent global named lookup server (LIGLO) to identify peers with dynamic (or unpredictable) IP addresses. The BestPeer is evaluated on a PC cluster consisting of 32 Pentium II running Java-based storage manager. The experimental results show that the BestPeer provides excellent performance compared with traditional non-configurable models. Further experimental study reveals its superiority over Gnutella's protocol.

Key words: Peer-to-Peer; mobile agent; dynamic reconfiguration

摘 要: 在分布式 Peer-to-Peer (P2P)系统中,每个节点都具有相同的能力,负有相同的责任.它们将自己的资源提供给系统,同时可以共享系统中的信息和服务.但是,大多数 P2P 系统都具有局限性,比如说 Napster,Gnutella,它们提供资源共享的粒度很粗糙,进一步地,它们忽略了文件在内容粒度上的共享.并且,大多数系统中节点的

* Supported by the Cross-Century Talent Raising Program of Ministry of Education of China under Grant No.2000-82 (国家教育部跨世纪优秀人才培养计划); the Fok Ying Tung Education Foundation under Grant No.81062 (霍英东教育基金青年教师基金)

第一作者简介: **黄维雄**(1973－),男,马来西亚人,博士生,主要研究领域为数据库技术,分布式计算.

Peer 是静态指定的,系统中不支持只具有临时 IP 的节点加入.提出了一个 BestPeer 的 P2P 系统,它具有以下几个特性:首先,它将移动 Agent 技术与 P2P 技术相结合,从而可以在数据提供者本地进行数据处理.其次,这个系统具有自配置的特性.一个节点可以动态地决定与哪个节点直接相连(决定哪个节点成为该节点的 Peer),从而使网络结构的配置达到最优.另外,BestPeer 系统提出了一个位置独立的全局名查找服务器(LIGLO).这个服务器可以惟一地识别具有动态 IP 地址的节点,从而使节点能够在它的 Peer 的 IP 地址发生改变后仍然能够找到它.BestPeer 系统建立后,提出了一套评估系统的方法.采用 32 台 Pentium II PC 机进行实验,每台机器上都运行基于 Java 的存储管理程序.实验结果表明,BestPeer 系统具有比传统的无自调整能力的 P2P 系统更好的性能.更进一步的实验还显示了 BestPeer 系统优于 Gnutella 协议下的系统.

关键词: Peer-to-Peer;移动 Agent;动态自配置

中图法分类号: TP311 **文献标识码**: A

Peer-to-Peer(P2P)技术也被称为 Peer 计算.由于这种技术可以重构分布式系统,因而被视为当前最有潜力的网络技术之一.P2P 是一种网络体系结构,在这种体系结构中的每台计算机都拥有同等的能力,负有相同的责任.P2P 技术去除了集中式服务器,使节点之间可以直接交换资源和服务.同时,在 P2P 系统中,任何一个节点可以自由地加入和离开该系统.这种设计的分布式特性为正在发展的一些重要应用提供了更好的环境.

然而,大多数 P2P 系统都存在如下几方面的局限性:(1) 它们只提供文件粒度的共享而缺少对基于不同内容粒度查询的支持.(2) 它们缺少可扩充性和灵活性.没有一种简单、快速的方法可以扩展系统的应用,适应用户新的需求.(3) 每个节点直接相连的 Peer 是静态定义或随机指定的.(4) 当前的 P2P 系统,或者是依靠 DNS 服务器解决域名问题,或者使用集中式服务器维持全局惟一的命名.对于前者来说,由于域名服务器中的条目通常指的是永久性 IP 地址,这使不定连接的节点和具有临时网络地址的节点无法加入系统.对于后者来说,用集中式服务器来维持全局命名方案(如 Napster),服务器可能成为整个系统的瓶颈.因而,集中式的方法是不可扩充的.

本文提出了一种能够克服上述问题的新型的 P2P 平台系统(我们称其为 BestPeer 系统).在这个系统中,首先,我们把移动 Agent 技术与 P2P 技术有机地结合在一起.其次,我们根据某种原则采用动态的配置策略,从而保证对节点最有益的 Peer 会直接连接到该节点.最后,我们提出了位置独立的全局名查找服务器(LIGLO),用以识别在 BestPeer 网络中的所有节点.

1 BestPeer 网络概览

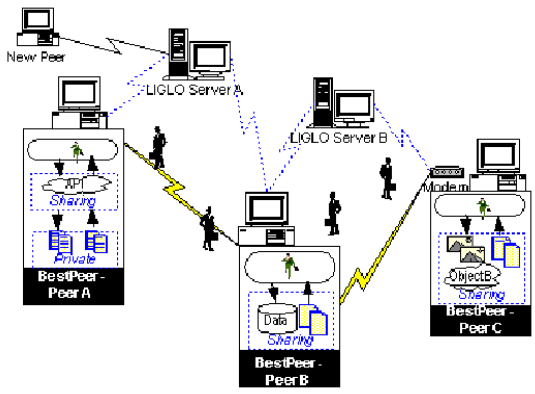


Fig.1 BestPeer network
图 1 BestPeer 网络

在进行大量研究之后,我们实现了 BestPeer.它是一个通用的 P2P 系统平台,在这个平台上可以简单、有效地开发各种 P2P 应用.同时,我们提出了一套评估 P2P 系统性能的系统化方法,这种评估方法既考虑了 P2P 系统的效率,又考虑了系统的有效性.根据这种方法,我们在 32 台 Pentium Ⅱ PC 上进行实验,结果表明,BestPeer 系统具有传统的无可配置性系统不具备的良好性能.同时,我们还在 Gnutella 协议下测试了 BestPeer 的性能,实验结果表明,BestPeer 的性能优于 Gnutella.

BestPeer 网络由两类实体组成(如图 1 所示),包括大量的节点计算机和少量的位置独立的全局名查找服务器(LIGLO 服务器).网络中的每个节点都运行 BestPeer(基于 Java)软件,可以与该网络中任意节点交互信息,共享网络资源.每个节点上有两种数据:私有数据和共享数据.任意节点只能访问其他节点的共享数据.

随着每个节点的不断加入、离开 BestPeer 系统,它的有效 IP 地址可能会发生变化.为了解决这个问题,我们提出了 LIGLO 服务器.它的功能是惟一地确定一个节点.任意节点可以通过 LIGLO 服务器确切地知道它的 Peer 是谁.这样,在一个 IP 地址改变的 Peer 再次登录 BestPeer 网络时,不会被认为是一个新加入的节点.当然,如果一个节点不关心它的 Peer 具体身份,那么就不需要使用 LIGLO 服务器.

在我们实现的 BestPeer 软件中,为每个节点提供了 Agent 可以驻留和执行操作的环境.这使 BestPeer 系统有更强大的处理能力和更高的可扩展性.

以下介绍一个节点是如何加入 BestPeer 系统的.

如果一个节点是首次登录 BestPeer 网络,将进行以下操作:

(1) 该节点在 LIGLO 服务器上登记.这类似于用户在 Internet 环境中在邮件服务器上登记.

(2) 节点登记后,LIGLO 服务器为节点分配全局惟一标识,我们把这个标识叫作 BPID(BestPeer ID).服务器并不考虑节点当前的 IP 地址,而是通过 BPID 确认节点.BPID 由一对标识组成(LIGLOID,NodeID),LIGLOID 是节点登记的 LIGLO 服务器的 IP 地址,NodeID 是 LIGLO 服务器为节点分配的 ID.

(3) 在节点登记的同时,LIGLO 服务器会把一个由(BPID,IP)对组成的表发给登记节点.表里保存的是可以与登记节点直接交换信息的节点信息,也可以称为登记节点的 Peer(将与之建立直接连接的节点).在这个表中,第 i 项的 BPID 值代表第 i 个 Peer 的标识,相应的 IP 值就是第 i 个 Peer 当前的 IP 地址.但是,在节点离开 BestPeer 网络时,它不必通知 LIGLO,那么在它离开后,其 IP 地址就成为无效值.因此,在 BestPeer 系统中,LIGLO 服务器要定期检查登记节点是否在线,确定节点的 IP 地址是否有效.

(4) 在进行上述步骤之后,登录节点成为 BestPeer 网络中的一员.它可以和任意一个 Peer 通信,并且不再通过 LIGLO.

另一方面,如果一个节点在它离开后想再次加入 BestPeer 网络中,则会进行如下操作:

(1) 该节点申请登录,然后将自己当前的 IP 地址发给 LIGLO 服务器.如果它的 IP 地址发生变化,LIGLO 服务器会自动进行记录更新.

(2) 登录节点将它的每一个 Peer 的 BPID 发给 Peer 登记的那个 LIGLO 服务器.比如说 Peer p,节点通过 p 的 BPID 可以检索到 p 登记的 LIGLO 服务器.

(3) 登录节点的 LIGLO 会向 Peer p 的 LIGLO 发消息确认 p 是否在线.如果 p 在网络中,p 登记的 LIGLO 服务器就会返回 p 的 IP 地址,否则就说明 p 不在线.这种操作对每一个刚刚登录的节点来说是很有必要的,因为如果 Peer 的 IP 地址发生改变,节点就无法确定 Peer 的位置,因而节点的需求可能不会被满足.但是,由于 Peer 离开网络的时候并不通知 LIGLO 服务器,因此,Peer 的在线信息可能会不准确.

(4) 登录节点加入 BestPeer 网络,它可以和 Peer 交换信息,也可以进行其他操作.

BestPeer 网络中的每个节点都会提供一部分资源允许其他节点共享,同时它也可以共享其他节点的资源.当一个节点希望得到某种资源的时候,它会向与它直接相连的所有 Peer 发出请求.这些 Peer 会将请求发给它们的 Peer.如此下去,一个请求会在网络中按照这种直接相连的原则有限制地广播,直至得到好的结果或者达到广播的极限(我们会在后面介绍广播的原则及极限的规定).如果一个节点可以响应请求,它会直接向请求者发出消息,而不是按照请求发送的路径返回消息.

2 BestPeer 的特性

2.1 移动Agent与P2P技术的结合

据我们所知,BestPeer 网络是第 1 个将移动 Agent 技术与 P2P 技术相结合的 P2P 系统.P2P 技术本身就是为了节点之间能够更好地共享资源,而移动 Agent 技术进一步提高了共享能力.因为 Agent 可以携带代码和数据,所以它可以有效地执行任何指定的功能.有了移动 Agent,BestPeer 不但可以提供文件和原数据的共享,而且可以在数据提供者本地处理数据,然后只返回处理结果.在 BestPeer 中,节点(起始节点)可以将 Agent 发送到它的 Peer 上,Agent 可以携带文件名并处理文件的代码.到达目标节点后,Agent 根据文件名处理文件中的内容,将结

果返回给起始节点.

在 BestPeer 系统中,并没有采用现有其他系统的 Agent 模式,而是采用我们自己的基于 Java 的 Agent.这种 Agent 技术和其他系统的 Agent 一样,要求 Agent 和它要执行代码的那个类(class)要同时存在于目标机上.如果目标机上没有类,单纯地发送 Agent 是无法执行任务的,因此在发送 Agent 的时候,要同时将类传送到目标机上.BestPeer 系统采用的也是这种纯粹的代码发送策略.使用这种 Agent 方法,可以使各个 Peer 并行地对数据进行操作,从而避免了节点成为瓶颈的可能.

Agent 的另一个重要功能是使 BestPeer 的节点在整个 BestPeer 网络中收集信息,然后进行处理,并且这种处理可以离线进行.Agent 在可以提供数据的目标机一端执行操作,同时可以将目标机的信息返回给起始节点.起始节点根据返回的信息判断哪个节点能为它提供更好的服务,可以代替现有的 Peer,从而使节点间的连接得到更好的配置.

2.2 资源共享

共享是 Internet 迅猛发展的一个主动力.绝大多数应用 P2P 技术的系统都会支持静态数据的共享.BestPeer 系统同样支持静态数据共享,同时也支持活动对象的共享,这使得数据共享可以在更好的粒度上实现;更为重要的是,BestPeer 系统支持计算能力的共享,使 BestPeer 网络的功能更加强大.在系统中,为了保持数据的一致性,所有对共享资源发出的请求都通过 Agent 来处理.

在 BestPeer 系统中,任何格式的数据文件都可以被完全交换,这些文件包括文本文件、word 文档、图像、音乐文件、电影文件、可执行代码等等.在大多数应用中,用户对文件的访问会有级别限制,不同的用户拥有不同的权限.一个用户可能被允许访问整个文件,而另一个可能只可以访问文件的部分内容.为了支持更好的共享粒度,在 BestPeer 系统中增加了活动对象的概念.活动对象由两类元素组成:数据元素和活动元素.数据元素用来描述对象的内容;活动元素就是一个活动节点,这个活动节点就是可以在上面进行各种计算和操作的节点.活动对象就是一个“黑盒子”,它可以接受、发送消息,通过接口与外部对象相互作用,根据请求者的权限返回相应粒度的文件内容.

作为一种 P2P 网络系统,BestPeer 的一个特性就是实现了数据计算能力的共享.共享计算能力具有的优点是:(1) 可以执行数据提供者端没有的操作;(2) 允许请求者根据它的要求过滤文件内容;(3) 具有灵活的可扩展性——加入新的算法或程序不会对系统的其他部分造成影响;(4) 可以很容易地扩展已经实现的系统;(5) 可以提高网络带宽的利用率.

2.3 重新配置BestPeer网络

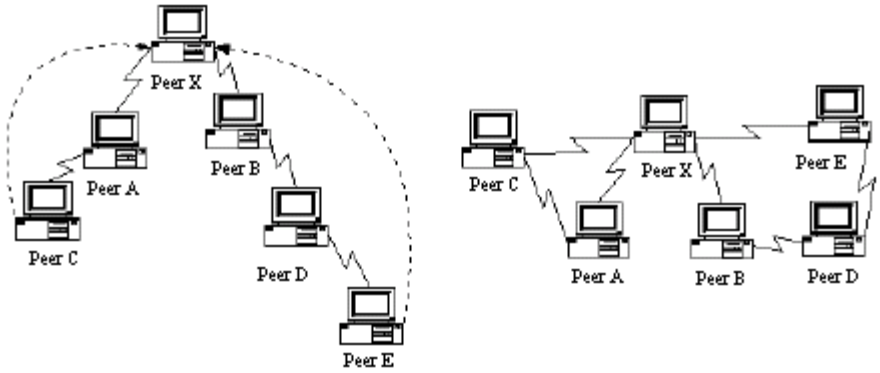
现有的 P2P 系统在 Peer 网络中,或者采用人工静态指定 Peer,或者随机决定 Peer.我们在 BestPeer 系统中采用完全不同的方法,使 BestPeer 网络中的节点可以动态地重新配置.这种重新配置是根据一个简单的假设实现的:曾经对一个节点有益的节点,在以后的查询中可能仍然是有益的.因此,BestPeer 系统中的节点会把曾经对它有益的节点作为直接的邻居(Peer),当然,每一个节点的 Peer 数量是有限制的.图 2 示例了一个节点的重新配置.在图 2(a)中,Peer X 是发出请求的节点.Peer X 的请求被直接发送给 Peer B 和 Peer A.但是,只有在 Peer C 和 Peer E 中才有相应的 Peer X 当前请求的对象,则 Peer X 直接从 Peer C 和 Peer E 获得结果.同时,Peer X 会发现尽管 Peer C 和 Peer E 对它有益,但是它们并不是直接邻居,因此,Peer X 重新配置自己直接相连的 Peer,将这两个节点加入其中.新的网络图结果如图 2(b)所示.

这种重新配置方法的实质就是将最有益的节点保持在最近邻位置上.每个节点都可以自动地重新配置,并且节点可以重新定义它的直接近邻数量,以便为自己提供更好的服务.

在 BestPeer 中,有两个预先给定的重新配置的策略.

第 1 个策略是 MaxCount.通过 Peer 返回给节点的结果数量决定怎样重新配置.它的处理过程是这样的:

(1) 节点提出请求后得到返回结果.节点根据每个 Peer 返回结果的数量将 Peer 排序.这是根据返回结果多的 Peer 会更好地满足将来的查询这一假设.



(a) Before reconfiguration (b) After reconfiguration

(a) 配置前 (b) 配置后

Fig.2 An example of self-configuration of BestPeer

图 2 BestPeer 自配置特征的一个例子

(2) 节点找出 k 个返回结果数量最高的 Peer(如果有的话),与它们建立直接连接(k 是系统参数,由节点来设置),重新配置网络.

通过这种策略可以方便地得到节点的 k 个 Peer,但是它不能把与 k 个 Peer 相连的 Peer 的信息包括进去.

第 2 个策略是 MinHops.如果可以从不远处的节点那里得到结果,那么就没有必要把它作为直接 Peer,因为不必经过很长路径就可以找到它.而找到更远处的有利节点可能会很耗时,所以可以将远处的节点作为直接邻居,这样可以在总共步数最小的情况下获得响应结果.根据这样的假设,节点根据查询步数排序,然后建立与那些步数多的节点的直接连接.

2.4 位置独立的全局名查找服务器(LIGLO)

由于 P2P 系统中的节点可能随时加入或者离开网络,所以节点的 IP 地址可能会经常发生改变.对于域名服务器来说,节点的 IP 地址是确定节点的标志.一个节点的 IP 地址发生变化后会被认为是一个新加入的节点.有一些应用要求能够惟一识别网络中的节点,在使用域名服务器的网络中无法满足这一要求.而在 BestPeer 系统中,我们使用了 LIGLO 服务器.通过 LIGLO 服务器可以惟一识别节点.在节点首次登录的时候,LIGLO 服务器会为它指定惟一的 BPID,以后节点登录只使用这个 BPID,而不必考虑它的 IP 地址是什么.

LIGLO 服务器是一个有固定 IP 的节点,在这个节点上运行 LIGLO 软件.LIGLO 服务器主要有两个功能:一是为每一个节点产生一个 BestPeer 全局标识(BPID),另一个是记录 Peer 的信息,比如说 IP 地址、Peer 是否在线.

因为集中式服务器要处理大量的请求,所以系统对这些服务器的要求很高,集中式服务器还需要维持整个网络中全局名的惟一性.而在 BestPeer 系统中,LIGLO 服务器的数量不受限制,每一个 LIGLO 服务器只需要保存它的成员名字,并只需维持成员内名字的惟一性.LIGLO 服务器可以根据自身的能力指定成员的数量,因此,BestPeer 系统中不会对 LIGLO 服务器有很高的要求.当 LIGLO 服务器中的成员数达到极限的时候,LIGLO 服务器会拒绝分配新的 BPID,以保证自身的效率.当然,被拒绝的 Peer 可以去另外的 LIGLO 上申请.

3 实验研究

我们对实现了上述特性的 BestPeer 软件进行了性能测试.实验中,网络中的每个节点都安装了 BestPeer 软件,并设置了若干个 LIGLO 服务器,所有节点都在 LIGLO 服务器上登记.我们在不同的网络拓扑结构下进行实验,以比较 BestPeer 系统和单线程 Client/Server 体系结构(SCS)、多线程 Client/Server 体系结构(MSC)的性能.

比较中,我们设计了两个版本的 BestPeer 系统——不可以重新配置的静态 BestPeer(static BestPeer),称为 BPS;具有重新配置特性的动态 BestPeer(reconfigurable BestPeer),称为 BPR.这可以使我们观测到动态配置策略的优越性.实验表明,具有重新配置特性的动态 BestPeer 系统具有更好的性能.在讨论实验结果以前,我们将先提

出一种评估 P2P 系统的方案.

3.1 评估方法

任何系统都必须考虑两方面的因素:有效性(effectiveness)和效率(efficiency).前者考虑的是系统返回结果的质量;而后者则是考虑性能方面的表现.不像现有的分布式系统,目前还没有一个清晰的评估 P2P 系统的标准.有些像 Internet 环境的搜索,P2P 系统的查询结果依赖于所搜索的 Peer,这些 Peer 不可能包括网络中的所有 Peer.而且,同样的查询可能会涉及不同的 Peer,因为网络中的 Peer 总是不断地在改变(加入网络或退出网络).

我们提出了 3 种评估的方案:首先,基于固定数目的节点来评估不同的方案.这对于研究利用 P2P 技术实现一组节点间的协作是有用的.

其次,在 P2P 环境中,结果返回的速率是非常重要的一个评估指标.这是因为用户对于那些 Peer 将返回查询结果以及有多少 Peer 将会被搜索是一无所知的.所以过长时间的等待将不会被用户所接受.

第三,结果返回的质量和数量也是重要的评估指标.一个节点可能会很快地返回查询的结果,但是查询结果可能很少或是跟查询不是太相关.尽管质量是基于查询语意的,但是查询结果的数量更容易获得并作为评估的标准.

3.2 实验环境

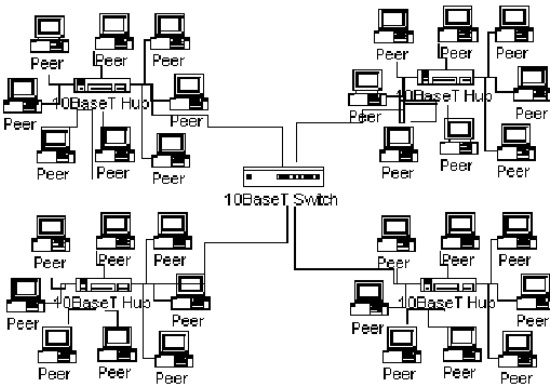


Fig.3 The environment of the experiments

图 3 实验环境

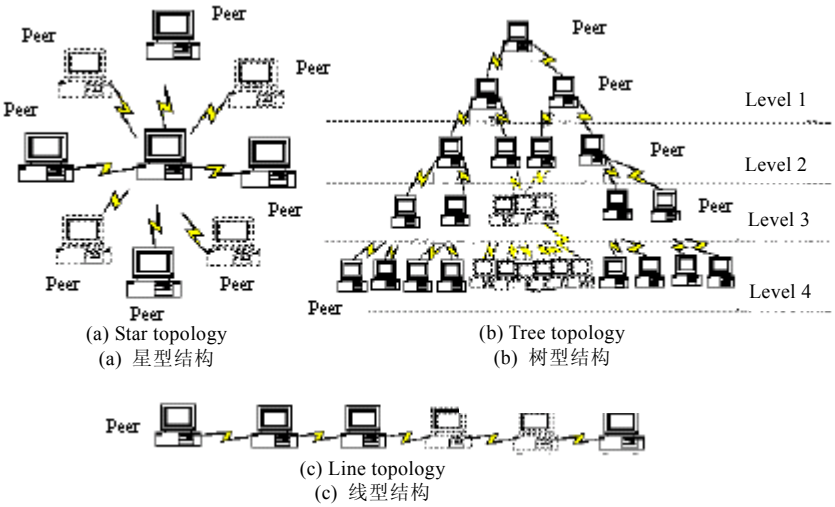
实验是在 32 台装有 Pentium200MHz 处理器,64M 内存的 PC 机上进行的.机器的系统配置如下:9 台机器安装 WinNT4.0,22 台机器上安装 Windows98,1 台机器上安装 Windows Millennium.网络的物理结构如图 3 所示.实验展现的结果对应于至少 3 次操作的平均.

实验中我们为每一台机器安装了一个叫做 StorM 的完全基于 Java 的永久存储管理程序[2,6].节点中可以共享的数据存储在 StorM 中.在这个实验中,每个节点的 StorM 中存储了 1 000 个共享对象,Agent 可以通过 StorM 的接口访问这些对象.我们实现了与 StorM 对应的 StorM Agent.可以使用 StorM Agent 输入用户请求,搜索整个 BestPeer 网络.这样做的目的是用 StorM 中的对象完成用户查询的匹配.发送 StorM Agent.提出请求的节点将 Agent 发送给与自己直接相连的 Peer.执行 StorM Agent.接收到 Agent 的节点产生一个线程执行 Agent.与 StorM 对象交互.Agent 将请求与共享 StorM 数据库中的每一个对象相比较,所有匹配的结果被保存在一个临时数组里.

3.3 不同的拓扑结构

我们首先在不同的网络拓扑结构下评估 BestPeer 系统,即在星型、树型、线型网络结构下进行实验(如图 4 所示).在星型结构下(图 4(a))中心节点是查询的发起节点,其他节点和中心节点直接相连.在树型结构(图 4(b))中,根节点是查询的发起节点,除叶节点外的所有节点均有 k 个直接相连的节点.在线型结构(图 4(c))中,除了两端的节点,每个节点均有两个直接相连的节点.这里我们用最左边的节点发起查询.

实验中我们从 1~32 变换节点的数目,每种策略运行多次,使用完成时间(completion time)作为性能标准.完成时间是指网络所有节点的查询结果都被收到的时间.图 5 显示了不同拓扑结构下的实验结果.



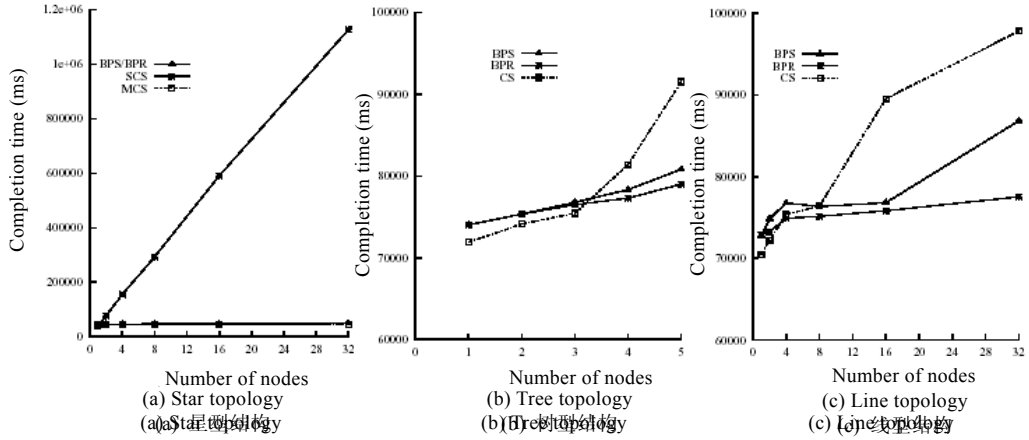
(a) Star topology
(a) 星型结构

(b) Tree topology
(b) 树型结构

(c) Line topology
(c) 线型结构

Fig.4 Different topologies in the experiments

图 4 实验中使用的不同拓扑结构



(a) Star topology
(a) 星型结构

(b) Tree topology
(b) 树型结构

(c) Line topology
(c) 线型结构

Fig.5 Experimental results on different topologies

图 5 不同拓扑结构下的实验结果

3.3.1 星型结构

图 5(a)是星型结构的实验结果.首先,我们注意到静态 BestPeer(BPS)和自配置 BestPeer(BPR)具有相似的表现.这是因为在星型结构中,这两种方案没有差别.正如结果显示的,随着我们增加网络的规模,单线程 C/S(SCS)比其他模型表现要差,这是因为 SCS 在任何时候仅处理一个连接,它必须在处理完对一个节点的操作之后才能转向另一个节点的操作.我们还注意到,MCS 和基于 BestPeer 方案的性能均显著好于 SCS.这是因为这两个方案都试图并行处理多个连接,并将查询同时传输给所有 Peer.我们还观察到,MCS 要略优于 BPS/BPR.我们会在讨论树型和线型结构时对此做进一步的解释.由于 SCS 的性能很差,我们将不再对其做进一步讨论.在下面的实验里,我们仅考虑 MCS.为简便起见,我们把 MCS 用 CS 表示.

3.3.2 树型结构

图 5(b)是树型结构的实验结果.在该实验中,我们在第 5 级用 48 个节点而不是 63 个节点.我们得到了一些有趣的结果.首先,我们注意到 CS 比 BPS/BPR 性能要优越.这是可以预料的,因为 BPS/BPR 本质上是代码传输策略:不仅要传输 agent 到 Peer,同时还有额外的在 Peer 端 agent 的构造开销.而在 CS 模型下,只需要简单地将查询提交到 Server,然后由 Server 端的算法处理查询.因此,当树型结构仅有一级时,CS 的性能要优于 BPS/BPR(即

星型结构),但是当树型结构的级数增加时,CS 的性能开始下降.这是因为 CS 需要数据按照查询的原路径返回.而 BPS/BPR,查询结果直接被返回到查询节点.

比较 BPR 和 BPS,显然 BPR 要优于 BPS.这是由于 BPR 能够通过自配制策略使网络结构变得更优.BPS 却不得不总是传递查询给同样的节点,而无论它们是否提供了查询的结果.

3.3.3 线型结构

线型结构的实验结果(图 5(c))显示了和星型结构类似的行为.本质上,不同的方案在线型结构上表现出和树型结构同样的相对性能.例如,在大多数情况下,BPR 要优于 BPS,而 BPS 要优于 CS(除了在网络节点数量很少的情况下).

3.4 请求的响应时间

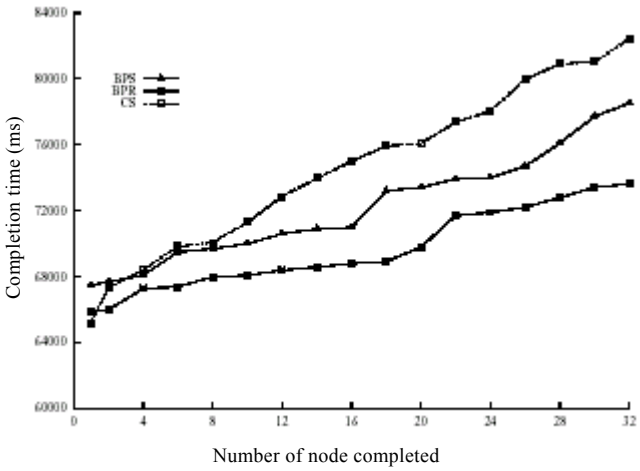


Fig.6 Return rate of query

图 6 查询结果返回率

我们分别考察了静态 BestPeer、动态 BestPeer、CS 模型上查询结果的返回时间.实验中,节点的数量是 32,我们使用树型结构.一个查询被发出 4 次,实验结果取 4 次的平均值.实验结果如图 6 所示,在图中,点(K,T)显示在 T 时间后得到 K 个节点的响应.我们注意到,有可能在不同的模型下,不同的节点会以不同的速度返回不同的查询结果.关于这一点的讨论,我们将留到下一个实验中进行.

整个实验中,动态 BestPeer(BPR)的性能一直是比较好的,在进行若干次查询以后,动态 BestPeer 会进行重新配置,所以在以后的查询中节点可以直接访问有益的 Peer.实验中我们注意到,CS 结构返回结果比 BestPeer 要慢(除了在最开始的几个节点的时候).这是由于查询结果在 CS 中要按原查询路径返回.

3.5 结果的质量

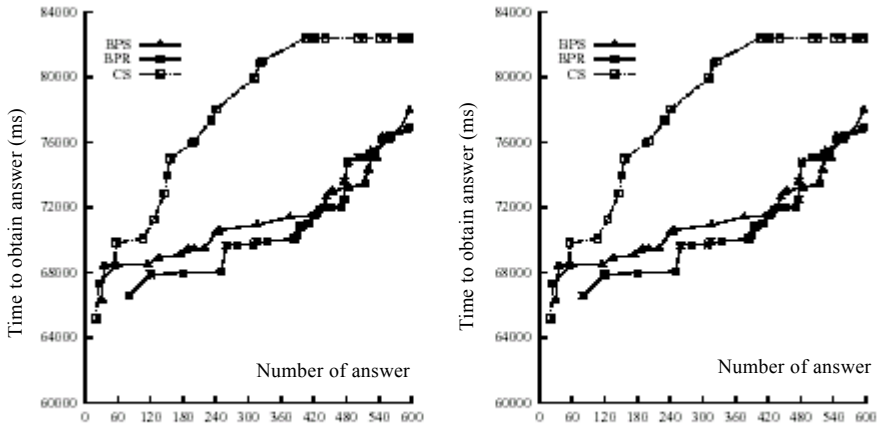
只有高的响应速度是不够的,我们还要对返回结果的质量进行比较.因为有可能那些响应迅速的节点仅提供了为数很少的查询结果.这里,我们所考虑的质量就是返回结果的数量.在早先的实验中,我们记录了每个节点返回查询结果的数量.图 7 显示了该实验的结果.显然,CS 最先返回的结果要比 BPS/BPR 都快.这是由于在 CS 下,最先直接相连的收到查询后立刻将结果返回.而对于 BPS/BPR,代码传输的开销使得最初的响应会有所滞后.但是当更多的结果返回时,BPS/BPR 要优于 CS 模型.这说明 P2P 模型要优于传统的 CS 模型.我们还注意到,BPR 的性能总体上要优于 BPS.

3.6 BestPeer与Gnutella的比较

FURI[1]是一个基于 Gnutella 协议的 Java 程序,通过该程序可以加入到 Gnutella 网络.它是一个具有用户界面的完整程序,提供了绝大多数 Gnutella 系统的功能.在该实验中,我们将比较 Gnutella 和 BPR(下面我们用 BP 来表示 BPR).实际上,Gnutella 系统与静态 BestPeer 类似(例如一个节点有固定数目的 Peer,没有动态的配置策略),所以我们只对 Gnutella 和动态 BestPeer 进行比较.该实验中,每个节点有 1 000 个可共享的文本文件(由于我们从文献[1]获得的代码仅能评估在文本文件中的关键词检索).我们还预先设定查询结果仅从固定的若干个节点中返回.这样,完成时间就由所有查询结果返回的时间来决定.我们重复单一的查询 4 次.通过若干次实验获得平均的结果.实验结果如图 8 所示.在图 8(a)中显示了每次查询的结果,每个节点有 8 个直接相连的 Peer.我们观察到,Gnutella 对查询重复的次数并不敏感.这是因为它每次都是重复同样的搜索路径.相反地,我们发现对于

BP,第 1 次查询的完成时间要远大于后几次的查询时间.这是因为对于第 1 次查询,BP 同样也要通过路由所有的中间 Peer 到达有查询结果的节点.但是,对于后续的查询,BP 的自配制策略使得这些有查询结果的节点被直接连接到了查询发起节点.这样,后续的查询完成时间明显地减少了.我们还观察到,BP 的性能在每一次查询中均优于 Gnutella.这是因为在实验中,BP 直接将查询结果返回给查询节点.而在 Gnutella 系统中,查询结果是按照查询路由原路返回给查询节点的.

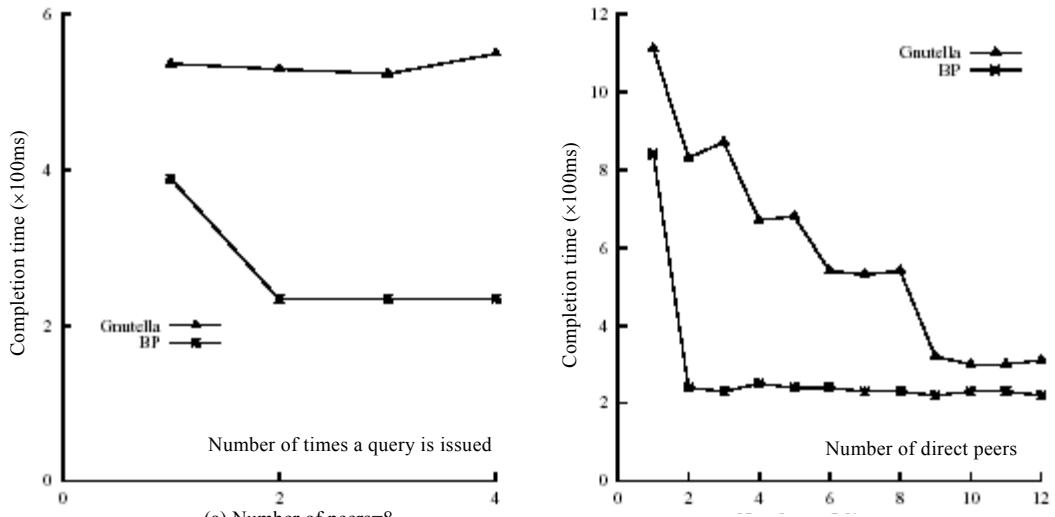
从图 8(b)中我们可以看到,直接相连 Peer 的数目对查询完成时间产生一定的影响.当直接相连的 Peer 数目增加时,BP 的性能仍然优于 Gnutella.尽管 Gnutella 的性能随着直接相连 Peer 的增加而有所提高,但是每次查询遍历同样的路径以及查询结果原路的返回仍然使 Gnutella 具有较差的性能.



(a) First search query
(a) 第 1 个搜索查询

(b) Last search query
(b) 最后一个搜索查询

Fig.7 Number of query results returned
图 7 查询返回结果数



(a) Number of peers=8
(a) Peer 数目=8

(b) Effect of number peers
(b) Peer 数目的影响

Fig.8 BestPeer vs. Gnutella
图 8 BestPeer 对比 Gnutella

4 相关工作与总结

由 Mistubishi Electric 开发的 Concordia 平台[3,12]提供了基于 Java 的移动 Agent.Agent 的移动性是通过 Java 顺序化和类的装载机制实现的.每一个 Agent 对象与一个单独的 Itinerary 对象相联系,它指定了 Agent 的移动路径.这种方法可以在任一台主机上应用.在文献[9]中,Aglets 环境允许创造可以相互合作的、用来完成复杂任务的 Agent 组.在文献[10]中也实现了一个基于 Java 的系统——Ajanta,它是利用 Java 的可串行性(serialization)来支持 Agent 的移动性.

基于 Agent 的系统充分利用了 Agent 的合作性和交互性,但是这些无法用来支持 P2P 技术.在这些平台上开发 P2P 应用需要付出更多的努力、更大的代价.如前所述,已开发成功不少 P2P 系统[4,5,8,11~13].但是,在用户需求发生改变时,多数 P2P 系统无法很容易地扩展它的应用.已有研究者开始将 P2P 技术引入数据库领域[7,14].文献[7]提出了解决数据布局(data placement)问题的方法.在文献[14]中,Garcia-Molina 等人研究了一些杂合型的 P2P 系统,但这些系统仍然以集中式为主.具体来说,他们提出了一种描述系统性能的分析模型,通过对不同结构的系统(例如链型结构、完全复制结构、HASH 结构非链型结构等)的比较,证实了分析模型的有效性.

本文介绍了一个具有良好性能的 P2P 平台系统——BestPeer,可以用这个平台来支持许多应用.BestPeer 具有若干很好的特性:(1) 因为结合了 Agent 技术和 P2P 技术,所以它可以很容易地被扩展;(2) 它基于某种优化原则,具备重新配置网络结构的机制;(3) 它使用分布式的 LIGLO 保存网络中节点的一些重要信息.性能实验表明,BestPeer 是一个很有前景的分布式处理系统.我们将在以下几个方面进一步完善该平台,并在该平台上进一步开发针对特定环境的应用:(1) 在当前的系统实现中,节点总是将它的 Agent 转发给目的节点,并在目的节点处处理数据.我们计划尽可能地使每个节点具有更多的智能,使它在运行期间可以自己决定发送的策略是代码传输还是数据传输,从而决定是在目的节点处处理数据还是在本地处理数据.(2) 进一步研究怎样在 BestPeer 系统中进行数据布局和数据复制策略,以便提高整个平台系统的性能.(3) 在该平台的基础上开发一个校园网环境下的基于 P2P 技术的缓存共享系统,有关该系统的具体构想参见文献[15].

References:

[1] FURL. http://www.jps.net/williamw/furi.

[2] Bressan S, Goh CL, Ooi BC, Tan KL. Supporting extensible buffer replacement strategies in database system. In: Delis A, *et al*., eds. Proceedings of the ACM SIGMOD International Conference on Management of Data. Philadelphia: ACM Press, 1999. 52~64.

[3] Wong D, Paciorek N, Walsh T, DiCelie J, Young M, Peet B. Concordia: an infrastructure for collaborating mobile agents. In: Rothermel K, *et al*., eds. Mobile Agents, First International Workshop. Berlin: Springer-Verlag, 1998. 86~97.

[4] Freenet Home Page. http://freenet.sourceforge.com.

[5] Gnutella Development Home Page. http://gnutella.wego.com.

[6] Goh CL, Bressan S, Ooi BC, Anirban M. Storm: a 100% java peersistent storage manager. In: Chaudhri AB, ed. Proceedings of the OOPSLA Worshop on Java and Object. Denver: Hermes Penton Science, 1999. 12~13.

[7] Gribble S, Halevy A, Ives Z, Rodrig M, Suciu D. What can database do for peer-to-peer. In: Mecca G, *et al*., eds. Proceedings of the 4th International Workshop on the Web and Databases. Santa Barbara, 2001. 31~36.

[8] ICQ Home Page. http://www.icq.com.

[9] Karjoth G, Lange DB, Oshima M. A security model for aglets. Lecture Notes in Computer Science, 1997,1419:188~205.

[10] Karnik NM, Tripathi AR. Security in the Ajanta mobile agent system. Software-Practice and Experience, 2001,31(4):301~329.

[11] LOCKSS Home Page. http://lockss.stanford.edu.

[12] Napster Home Page. http://www.napster.com.

[13] SETI@home Home Page. http://setiathome.ssl.berkely.edu/.

[14] Yang B, Garcia-Molina H. Comparing hybrid peer-to-peer systems. In: Apers P, *et al*., eds. Proceedings of the Very Large Data Base. Roma: Morgan Kaufmann, 2001. 561~570.

[15] Wang X, Ng W, Tan K, Ooi B, Zhou A. BuddyWeb: A P2P-based collaborative web caching system. In: Cugola G, *et al*., eds. Proceedings of the International Workshop on Peer-to-Peer Computing. Pisa: Springer Press, 2002. 72~77.