非结构化离散型对等网络的枢纽副本复制机制

霍 林[1,2] 方 艺[2] 胡和平[1] 黄保华[2]
(华中科技大学计算机学院 武汉 430074)[1] (广西大学计算机与电子信息学院 南宁 530004)[2]

摘 要 P2P网络使得网络中的数据传输更加方便和高效。当前大多数P2P相关研究集中在路由算法及结构化网络拓扑方面,忽略了非结构化离散型副本复制的研究。提出了一种基于非结构化离散型对等网络的枢纽节点副本复制机制(JRM)。通过该机制,可以降低非结构化离散型对等网络中的数据流量并实现更好的负载平衡。给出了相关算法伪代码,并通过分析证明了该算法的优势。

关键词 副本复制,非结构化,超级节点,枢纽,负载均衡
中图法分类号 TP393.07 **文献标识码** A

Replication Mechanism Based on Unstructured Decentralized P2P Network

HUO Lin[1,2] FANG Yi[2] HU He-ping[1] HUANG Bao-hua[2]
(College of Computer Science & Technology, Huazhong University of Science & Technology, Wuhan 430074, China)[1]
(College of Computer, Electronics and Information, Guangxi University, Nanning 530004, China)[2]

Abstract P2P network makes computing and data transmission efficient and convenience. Nowadays, most of studies focus on routing algorithms and structured topology of the network. However they ignore the research in unstructured decentralized P2P network's replication mechanism. We focused on the area and proposed a new mechanism called Junction Replication Method (JRM) in unstructured decentralized P2P network. With the method, we can achieve better load-balancing by reducing information flow. At last, the paper gave the pseudo code of the algorithm and proven the advantage by deduction.

Keywords Replication, Unstructured, Supernode, Junction, Load-balancing

1 引言

1999年,随着Napster的问世和广泛传播,P2P对等网络迅速成为研究者们新的研究热点。数据[1]表明,P2P网络的流量已经占据60%Internet流量并且以每月3 exabytes的速度增加。相对于传统的C/S模式,P2P网络被认为是一种效率更高、负载更均衡和容错性更好的网络。通过P2P网络,大量的计算机可组织在一起形成一个高性能、高可靠性、高扩展性和低消耗的运算整体。

当前,P2P网络主要分为3类:集中型、离散型和混合型[3]。其中每一类又可进一步分为结构化、非结构化以及松散结构化3种。集中型对等网络,如Napster,通过使用稳定的中央目录服务器,将节点的查询请求快速定位,但容易造成单点失效,并且服务器性能往往成为制约整体性能的瓶颈。离散型对等网络中无中央服务器;非结构化离散型网络,如Gnutella[2],没有拓扑要求或可以控制文件排放的机制;结构化离散型网络,则有一定的控制约束,如利用分布式哈希表有效处理查询请求。混合型网络,如Kazaa,可以看作结合了集中型和离散型优点的网络,通过超级节点或放置目录服务器的方法达到提高效率的目的。松散结构化也是介于结构化和非结构化之间的折衷,利用文件存放沿途留下的线索指引发现文件,如Freenet。表1给出了不同类型对等网络的分类和典型应用。

表1 P2P存储系统分类

	结构化	非结构化	松散结构
集中型		Maze, Napster, itTorrent	
离散型	Chord, CAN, ceanStore, PAST	Gnutella	Freenet
混合型		Kazaa, Guntella	

由于P2P存储系统面临节点异构性、节点自私性、节点安全性以及节点失效性等诸多问题,在当今研究中,多数研究集中于搜索与定位机制,而忽略了网格中的基础要素:复制机制[4],特别是非结构化离散型对等网络中的副本复制机制。而在现实网络中,已经很难通过设置中央服务器来对节点进行结构化设置或添加约束条件,以满足不同网络需求。非结构化离散型P2P网络中的节点通过泛洪查询,保证查询的成功率,并且节点可以自由加入或退出,这些都代表了最真实的网络拓扑结构。基于上述原因,关于非结构化P2P网络的副

到稿日期:2009-07-02 返修日期:2009-10-28 本文受863项目(No. 2007AA01Z403)资助。
霍 林(1965-),女,博士生,教授,主要研究方向为网络与并行分布式计算、信息安全,E-mail:nnxhy@163.com;方 艺(1983-),男,硕士,主要研究方向为网络与并行分布式计算;胡和平(1953-),男,教授,博士生导师,主要研究方向为网络与并行分布式计算、信息安全;黄保华(1977-),男,博士后,主要研究方向为网络与并行分布式计算、信息安全。

本复制研究是十分有意义的。

本文基于非结构化离散型 P2P 系统，提出中心枢纽复制机制(Junction Replication Mechanism，JRM)。通过这个机制，P2P 网络中的分布式存储可以在保持或提高负载均衡的同时降低网络中多余的数据流量。同时，我们利用 JRM 分担超级节点压力，实现进一步的负载均衡。最后，通过推导证明本方法的正确性和优势。

2 非结构化网络中的副本复制

分散式非结构化对等网络中既没有集中的目录服务器，又对 P2P 网络拓扑和文件放置没有任何精确的控制。系统中每个节点只维护自己存储的文件及其索引。节点在加入网络时只需遵守一些松散的规则，形成随机的覆盖网络拓扑。查找文件的典型做法是以一定的半径在网络中泛洪查询消息。接收到消息的节点将其与所存储的文件进行比较，若存在正的匹配，则向提出查询的节点回答；否则，若其生命期(TTL)不为 0，则将查询消息向邻居节点泛洪，为 0 则将之丢弃。

非结构化设计的优点是能够适应动态系统，可以灵活地处理节点不断加入或者离开事件并支持多样化查询；缺点是当前的搜索算法对网络及参与者产生很大的负载，不利于扩展，并且由于使用限制范围的查找，即使目标文件存在于系统，仍可能找不到。这样，对稀少文件的定位非常不利。

在 P2P 网络中，节点的对等性决定了不能将数据单独存放在一个节点上。否则，既会造成数据难以查找，又会在大量访问时造成网络上的拥堵。P2P 副本复制机制通过追寻用户使用数据的习惯，改进 P2P 环境中的副本存储，以利于提高资源发现处理效率，达到更好地利用 P2P 网络。非结构化 P2P 网络的网络拓扑是动态和不可预测的，因此副本的地位更加重要。若不存在副本，节点将耗费远超正常的时间来获取数据并处理。通过复制机制，在网络中建立冗余，可以显著提高网络的效率和可用性。

在非结构化离散型 P2P 网络中，节点间没有约束机制保证紧密联系，数据或文件随意分布在网络中。节点将查询请求泛洪发出，以保障资源发现效率。数据持有节点接收到查询时，与查询请求节点建立连接。考虑到网络带宽和异构节点的不同性能，副本位置是制约 P2P 网络性能的重要因素。

当前在非结构化离散型 P2P 网络中有两种常用的典型副本选择机制[7,8]：

1)沿路缓存。节点需要文件时，首先查询自身节点，若存在则直接调用。如果目标数据文件不在查询源节点中，将查询消息泛洪发送至周围节点。若查询成功，并且查询次数超过一个阈值，则在查询源节点到目标数据文件节点的路径节点上缓存目标数据文件，通常是根据查询消息中的路由表向距离源节点更近的目标节点邻居中复制数据。

2)源节点缓存。同样，当节点需要文件时，先查询自身节点。不同在于，如果目标数据文件不在查询源节点中，并且查询次数达到一个阈值后，将目标数据文件沿请求中的路由返回，仅在查询源节点上缓存目标数据文件。

图 1、图 2 举例说明了两种机制的复制方法。假设 A1 和 A3 请求的数据唯一存在于 A6，当 A6 接收到查询请求后，建立通往 A1 和 A3 的路径以传输文件。沿路缓存策略中，当文件请求次数超过阈值，A6 将该副本向路径中离请求节点较近的邻居节点复制，图例中为 A5。源节点缓存策略中，则在超过阈值后，直接向查询源节点 A1 和 A3 复制该副本。

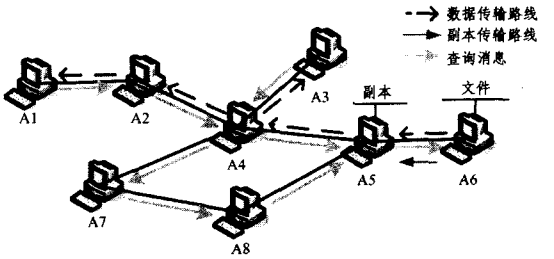

图 1 沿路缓存副本复制方法

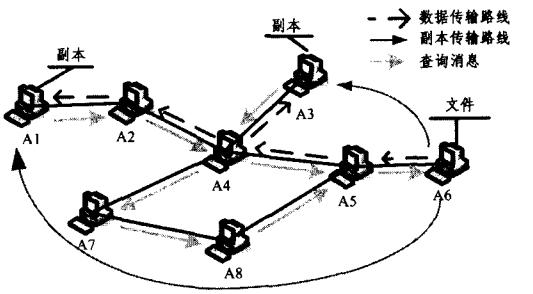

图 2 源节点缓存副本复制方法

沿路缓存可看作一种阶段性提高负载均衡的方法，但该方法不能迅速解决网络负载均衡问题。而源节点缓存作为一种高效的解决负载均衡的方法，易造成资源被大量复制，由于节点的存储空间都是有限的，无法同时保存大量副本，进而增大节点存储的负载压力。如何在保证高处理效率的同时降低网络流量和节点负载压力，是值得研究的重点问题。在下一节中，介绍基于枢纽节点的副本复制机制。

3 基于枢纽节点的副本复制方法

本节介绍基于枢纽节点的副本复制机制(JRM)。此算法的灵感来自于马路上的交叉路口。试想 A，B 两条路在交叉路口处汇集成一条路 C，从 A，B 来的司机都想到达 C 处，从而获取那儿的资源。假设 A，B，C 道路一样宽，而 A，B 原有的道路负载已经达到 80%，则在交汇后 C 路的负载在不考虑原有负载的情况下达到 160%，远远超过其运输能力。而负载失衡的源头就起源于 A，B 和 C 的交叉路口。

传统的方法可以理解为将资源从 C 的终点逐渐拉近至 A 和 B，这种方法虽然最终能解决负载失衡的问题，但耗费时间过长。C 的路径越长，沿路缓存的效果越不明显，源节点缓存则是将资源直接发送到 A，B 端请求资源的位置，这种方法效果明显，但当用户请求多个资源时，会增加自身的存储压力。负载失衡的起点是其枢纽节点，即 A，B 的交叉点，因此最好的解决方法是在枢纽节点建立副本。通过枢纽节点的副本，来自不同方向的查找源节点可以在快速得到资源的同时不提高网络占用率。同时，枢纽节点相比沿路缓存中目标节点的邻节点可以影响更大的范围。因此理论上在非结构化对等网络的枢纽节点建立副本是有益的。图 3 介绍了这种机制。

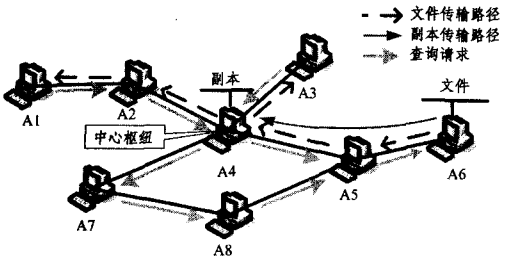

图 3　基于枢纽节点的副本复制机制

在 JRM 机制中，它可以在数据存储节点记录所有来访查询的路由并提取其中共有路线。同图 1 实例，A1 和 A3 发出的查询在 A4 处交汇并传递到 A6，说明 A4 是通过 JRM 提取出的枢纽节点，而 A4 到 A6 是查询的共有路线。设置触发机制，当对 A6 中的某资源请求次数超出阈值后，JRM 即启动副本复制机制，直接将数据副本复制到 A4。该方法直接降低 A4 到 A6 段负载，同时不增加查询源节点到交汇前的负载。接下来，将详细介绍基于枢纽节点的副本复制机制。

3.1　JRM 框架

在 JRM 框架中，需要维护一个数据库并设置合理的算法来触发该机制。换言之，有 4 个表和 1 个算法需要维护。本节首先介绍 JRM 框架中的数据库表设置。

首先我们关注的是路由表(Routing Table，RT)。表 2 给出路由表格式。在非结构化离散型 P2P 网络中，一个节点不知道网络中所有的节点，只能获取周围邻居节点的信息。这种特性造成了其发出的请求信息中，要保存所有经过路由的信息，这样使得节点发出的查询请求通过包含路径信息的方法使得请求数据能沿原路返回。因此，保存数据的目标节点在接收到查询请求时就能收集到返回源节点的完整路由信息。将该路由信息提取出来，并建立专门的路由表储存这些信息。在这里，将路由信息以序列的方式记录。以上文出现的例子来说，A6 节点接收到两个查询请求并转换成序列并记录为 Sequence1{A6，A5，A4，A2，A1}和 Sequence2{A6，A5，A4，A3}。

表 2　路由表格式

	Routing Sequence
Sequence 1	A6，A5，A4，A2，A1
Sequence 2	A6，A5，A4，A3
……	……
Sequence N	A6，A5，A8，A7

其次，在一个节点中，根据本节点存在的不同数据建立与之对应的独立的中枢请求表，以记录针对这些数据的请求次数。这里的请求次数是枢纽节点的请求次数。利用算法调用路由表数据，通过提取公共路径可以找出枢纽节点。若发现多个查询请求经过这个枢纽节点转至目标节点，则目标节点在所请求数据对应条目针对这个枢纽节点的次数加 1。请求次数在 JRM 中至关重要，决定着何时触发副本复制机制。该表的结构可以表示为(File ID，Node ID，request number，Node ID，request number …)。同时，JRM 数据库中设置一个采用同样格式的节点请求表，记录查询源节点信息，Node ID 为查询源节点，并记录源节点对数据的查询次数。通过记录查询源节点对特定数据的查询次数，使其与中枢请求表呈互补的作用。

最后，为了方便节点删除以及进一步平衡普通节点与超级节点的地位，在这里需要建立一个联系表(Contact Table，CT)，如表 3 所列。该表储存着节点中所有储存资源的原始节点信息及其对应的超级节点位置。下一节将利用这个表。

表 3　联系表格式

File ID	原始节点	超级节点
File 1	A5	A4
File 2	A3	A4
……	……	……
File N	A8	A4

3.2　JRM 算法

JRM 算法用于从路由表中提取出枢纽节点。因为已将查询请求中的路由信息转换成序列形式存储在路由表中，提取过程可以简单理解成通过序列比较挖掘共同项的过程。

整个过程可以表述如下：当保存数据的节点接收到新的查询请求时，该节点中的 JRM 机制将请求中的路由信息保存在路由表中，通过比较来发现枢纽节点。在中枢请求表中，JRM 统计每个枢纽节点的请求次数并设置阈值进行约束。同时，JRM 机制记录查询源节点的请求次数，因为若某一源节点对某特定数据请求次数达到一定阈值时，根据局部性原理，这些数据在不久的将来仍可能会被使用，即数据仍会被请求，此时枢纽副本复制机制就会因每次数据请求都有传输副本的消耗而相对缺乏效率，所以，有必要将源节点缓存设为枢纽副本复制的补充，即将被请求数据直接在源节点上进行复制。因此，需要设置另一个阈值来与查询源节点访问次数相关联，并触发源节点缓存机制。综上所述，通过源节点缓存对 JRM 机制进行补充，可以在降低网络中节点存储压力的同时降低网络负载。JRM 的算法伪代码如下：

(1) 发现并存储枢纽节点

```
Begin
//Distill and calculate junction node and requester
    If i>1 & Sequencei's status is unread
    For (j=1;j=i-1;j++)
    {a=null;
//Compare Sequencej with Sequencei
    For(x=1;x=Minlength (Sequencej,Sequencei);x++)
    {If Sequencej(x) =Sequencei(x),a= Sequencej(x) =Sequencei
    (x);
    Else break;}
//Save the algorithm result in Request Table
    If a≠null and in Request Table,Numbera++;
    Else if a≠null add a as a new element and set Numbera=1;}
    Change Sequencei's status to read;
    Save Sequencei's last member as requester's information to ta-
    ble;
End
```

(2) 触发不同副本复制

```
Begin
//Spring different mechanisms
  If request number in contact table over limit α
      Spring JRM and clear the number to zero;
  If request number in requester table over limit β
      Spring Replication to requester and clear the number to zero;
End
```

在设计算法过程中，将算法分为两个部分。第一部分是

发现并存储枢纽节点，第二部分是通过设置不同阈值触发不同副本复制机制。在这里，设置了两种不同的阈值，阈值 limit α，limit β 分别用于触发枢纽副本复制机制和源节点缓存机制。由于源节点缓存只是作为枢纽副本复制的补充，通常情况下，设置 limit α<limit β 以保证 JRM 机制的优先权。作为完整过程的一部分，JRM 机制利用最近最久未访问算法(LRU)来删除节点中的副本，从而降低节点的存储压力。利用 LRU 算法时，为了保证副本不是唯一存在于当前网络中，首先检查关于该副本的原始节点和超级节点中的存储信息。若存在相同的副本，则可删除，以便为新副本留下空间。

4 利用 JRM 分担超级节点任务

利用超级节点(Supernode，SN)改进非结构化离散型 P2P 网络的缺陷已经得到认可。超级节点是由用户自行选出的高性能节点，可以弥补非结构化离散型 P2P 网络的先天不足，如无法预知的网络拓扑、没有中央服务器[5]。传统的选取方式是在新节点进入系统后，主动广播自身能力信息，包括计算能力、存储能力等，通过其它在线用户投票选出。如今也有研究提出了基于区域划分的超级节点选取机制，通过 K-平均算法对系统中的节点进行区域划分，使物理位置相近的节点处于同一个域。域中默认超级节点，并可根据超级节点能力参数衡量选取新的超级节点。SN 的功能就如 C/S 模式下的中央服务器，但是更灵活。SN 可以让离散型网络自组织成域进而提高性能。但如果过分依赖超级节点，将会在 P2P 网络的效率和负载均衡上产生问题。

将超级节点引入 JRM 机制的非结构化离散型 P2P 网络后，经研究发现，超级节点地位特殊：一方面，JRM 提取出来的枢纽节点的性能未知，若出现低带宽、低存储容量等问题，利用超级节点可以起到很好的辅助作用；另一方面，可以通过 JRM 机制，降低超级节点的压力，避免对其过分依赖，进而防止单点失效并且更好地平衡网络中的负载。

综上所述，超级节点和 JRM 机制呈互补关系，我们提出一种新的机制来平衡 JRM 和 SN。在这里，假设网络中所有节点均已知道自己所在域对应的超级节点。

首先，设立一种机制去平衡普通节点和超级节点的地位。超级节点同其它节点一样，运行 JRM。若普通节点收到副本，说明它们是源节点或者是枢纽节点，它们将该副本复制至所在域的超级节点，并记录信息于联系表中。同时，针对普通节点设置阈值 limit γ，代表其最大连接数。limit γ 是动态可变的，其计算方法为

$$\text{Limit } \gamma = \text{Min}\left(\frac{CA}{CAeachcost}, \frac{SA}{SAeachcost}\right) \tag{1}$$

CA 和 SA 分别表示节点的计算和带宽能力，CAeachcost 和 SAeachcost 分别表示每次数据传输平均消耗的计算能力和占用带宽。通过该方法，可以发现制约节点性能的瓶颈并设置合理的阈值，预防过载发生。当最大连接数超过 limit γ，节点将自身状态从数据保存者转换成一个普通的转发节点，对于新到的数据请求不予回应，而是将请求转发至其对应的同样保存该副本的超级节点。通过请求中的路由信息，超级节点会自动选择最短路径，将查询源节点请求数据返回。根据此方法，超级节点只有在当枢纽节点满负荷的情况下才会调用自身能力提供副本，可以做到对周围节点的最好利用。同样，由于超级节点中也存在 JRM 机制，若超级节点过载，也可触发副本复制机制，将热门数据发送到新的枢纽节点中，从而提高整个网络中的容错性，实现负载均衡和性能提高。

5 JRM 分析

在这里将所有从查询源节点到枢纽节点的路径称为支流(Limb)，把从枢纽节点到数据保存节点的路径称为主干(Trunk)，如图 4 所示。

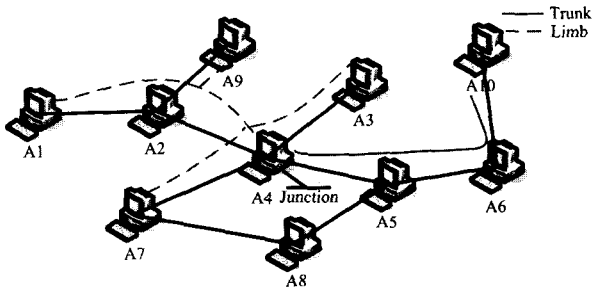

图 4 实验拓扑图

S_T 表示主干中的平均路径段数；S_L 代表支流中的平均路径段数；C 表示每段路径传输副本的消耗，假设每段路径消耗相等；A 和 B 为触发不同复制机制的阈值；R 表示查询的总数量，设 $R>\text{MAX}(A,B)$。

为计算方便，假设系统只有一个副本供使用，所有查询请求者以相同频率发送对该副本的请求，并且在每段路径中传输副本的负载消耗是相同的，对本机中副本处理的消耗可忽略不计。阈值 A 用来触发源节点缓存机制，阈值 B 用来触发沿路缓存和本文提出的 JRM 方法，则每个节点对同样规模的数据副本发出相同次数的请求，在不同副本复制机制下的消耗表达式分别如下。

源节点缓存消耗为

$$S_T \cdot C \cdot A + S_L \cdot C \cdot A + S_T \cdot C + S_L \cdot C \tag{2}$$

沿路缓存消耗为

$$S_T \cdot C \cdot B + S_L \cdot C \cdot B + (S_T - 1) \cdot C \cdot (R-B) + S_L \cdot C \cdot (R-B) + C \tag{3}$$

中心枢纽缓存消耗为

$$S_T \cdot C \cdot B + S_L \cdot C \cdot B + S_L \cdot C \cdot (R-B) + S_T \cdot C \tag{4}$$

上述 3 式中，前两项是请求次数超出阈值前的消耗，其余项是不同的副本复制机制启动后的消耗，在这部分要考虑到复制过程的消耗和分支路径的消耗等因素。

由式(3)－式(4)，得

$$(S_T-1) \cdot C \cdot (R-B) + C - S_T \cdot C = (S_T-1) \cdot C \cdot [(R-B)-1] \tag{5}$$

由式(2)－式(4)，得

$$S_T \cdot C \cdot A + S_L \cdot C \cdot A + S_T \cdot C + S_L \cdot C - S_T \cdot C \cdot B - S_L \cdot C \cdot B - S_L \cdot C \cdot (R-B) - S_T \cdot C = S_T \cdot C \cdot (A-B) + S_L \cdot C \cdot (A-B) + S_L \cdot C \cdot [1-(R-B)] = S_T \cdot C \cdot (A-B) + S_L \cdot C \cdot (A+1-R) \tag{6}$$

从式(5)可知：当主干中的路径段数 $S_T \geqslant 1$，且查询总数量 $R \geqslant B+1$，即当需要启动副本复制机制时，枢纽节点缓存的消耗比沿路缓存的消耗要少，并随着 S_T 或 R 值的增大，枢

(下转第 85 页)

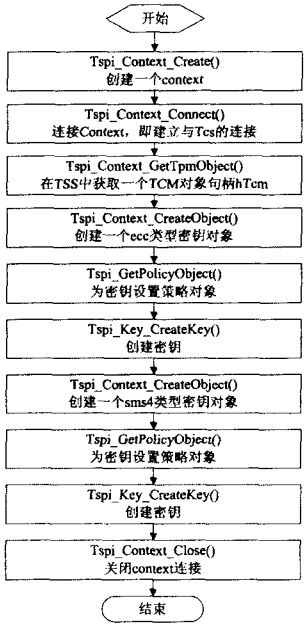

图 4　应用层密钥生成服务接口调用

结束语　本文对 TPM，TCM 的密钥机制差异以及对两种芯片的密钥管理方式进行了对比分析，对可信软件栈的体系结构与密钥管理功能进行分析，对 TCG 体系中的密钥创建流程进行描述，得出了通过重构可信软件栈对上层实现密钥服务应用兼容的思想。在此基础上，提出了对可信软件栈 TSP 层和 TCS 层的具体改进方案，即提供支持 TCM 的密钥机制，重新实现密钥生成流程中的部分环节，以解决基于不同密码算法开发的可信密码模块 TCM 所面临的密钥服务应用兼容问题。

参 考 文 献

[1] Trusted Computing Group. TPM specification version 1. 2. Part 1 Design Principles Revision 103[EB/OL]. 2007:19-21. http://www. trustedcomputinggroup. org/resources/tpm_specification_version_12_revision_103_part_1__3

[2] Strasser M. Software-based TPM Emulator for Linux[D]. 2004: 29-35

[3] 郭菲菲. 可信密码模块的密码配用研究[D]. 北京：北京交通大学，2008：20-30

[4] Zhang Xing, Zhou Ming, Zhuang Jun-xi. Implementation of Ecc-based Trusted Plaform Module[C]// Proceedings of the Sixth International Conference on Machine Learning and Cybernetics. 2007:2168-2173

[5] Trusted Computing Group. TCG Software Stack Specification Version 1. 2 Level1 ErrataA[EB/OL]. https://www. trustedcomputinggroup. org/specs/TSS/TSS_1_2_Errata_A-final. pdf, USA, 2007:516-543

[6] Trusted Computing Group. TCG Specification Architecture Overview Specification Revision1. 4 [EB/OL]. https://www. trustedcomputinggroup. org/groups/TCG_1_4_Architecture_Overview. pdf, USA, 2007:5-41

[7] 董玉娟. 支持异构可信平台的可信软件栈研究[D]. 北京：北京工业大学，2008：29-46

[8] 刘毅. 一种可信软件栈的兼容性改进方案[J]. 武汉大学学报：理学版，2009，55(1)：57-61

[9] Yoder K. Linux TCG Software Stack Low Level Design Version 0. 8r2[EB/OL]. http://trousers. sourceforge. net, USA, 2007: 20-30

[10] IBM. TrouSerS 0. 2. 9[EB/OL]. http://trousers. sourceforge. net, USA, 2007

(上接第 81 页)

纽节点缓存副本复制机制的优势愈发明显。

在式(6)中，设 $A \geqslant B$，因而可知，当 R 的值小于 $A+1$ 时，中心枢纽缓存副本复制机制具有优势。但当 $S_T \cdot C \cdot (A-B) + S_L \cdot C \cdot (A+1-R) \leqslant 0$，即 $R \geqslant [S_T \cdot (A-B) + S_L \cdot (A+1)] / S_L$ 时，源节点缓存机制则开始显示其优势。当查询请求数量 R 达到一定数值并还在不断增大时，源节点缓存有直接明显的优势，因为更多请求可以在本节点内处理。但这种方法将会带来副本安置问题。考虑到节点的存储能力有限，使用源节点缓存在一定时间内将造成网络中副本过多，因此本研究只将源节点缓存作为 JRM 机制的补充，即设置阈值 $A>B$，以保证优先执行 JRM 机制。

综上所述，枢纽节点缓存与源节点缓存相结合对于非结构化离散型对等网络有较好的副本复制性能。同时，我们需要将查询路由信息保存一定的时限，在降低数据库容量的同时为更新枢纽节点提供数据。下一步工作中，需要找出平衡数据库容量和准确提取枢纽节点的方法。在理论上，枢纽节点副本复制机制可以随着非结构化离散型对等网络的扩展而扩展。

结束语　本文提出基于枢纽节点的副本复制机制为对等网络的研究起到了补充作用，综合了传统副本复制机制的优点，并通过降低超级节点的运算量进一步平衡了网络中的负载，降低了网络流量并使得节点工作效率更高。

为了使得系统更加完善，仍然需要进行一些后续工作。首先，需要考虑 JRM 的安全性问题。在当前的 P2P 网络中，节点位置容易暴露的问题尚未得到很好的解决，而且匿名访问也对网络安全构成极大挑战。其次是副本一致性的维护。本文只利用超级节点和源节点保证了副本不会唯一存在，但副本更新时的一致性维护也是非结构化网络所必须面对的问题。最后，随着计算机的发展，超级节点的性能与普通节点的性能之间的差距越来越小，因此需要设计出更多的方法来进一步平衡网络中的负载，以达到提高网络效率的最终目标。

参 考 文 献

[1] The Exabyte Era White Paper[R]. Cisco Systems, January 2008

[2] Ripeanu M. Peer-to-peer architecture case study: Gnutella network[C]// Proceedings of P2P'01. 2001

[3] Tewari S, Kleinrock L. Search and Replication in Unstructured Peer-to-Peer Networks [R]. UCLA-CSD-TR050006. UCLA Computer Science Dept, March 2005

[4] Tewari S , Kleinrock L. Analysis of Search and Replication in Unstructured Peer-to-Peer Networks[C]// SIGMETRICS'05. June 2005

[5] Lo V, Zhou D. Scalable supernode selection in peer-to-peer overlay networks[C]// Second International Workshop. July 2005

[6] Mohammad, Salimullah, Raunak. A Survey of Cooperative Caching, [EB/OL]. http://lass. umass. edu/-raunak/survey. ps

[7] Clarke I, Sberg O, Wiley B. Freenet: A distributed anonymous information storage and retrieval system[C]// Workshop on Design Issues in Anonymity and Unobservability. 2000:311-320

[8] Kubiatowicz J, Bindel D, Chen Y. Oceanstore : Architecture for global-scale persistent storage[C]// Proceedings of th ASPLOS' 2000. Cambridge, MA, 2000:190-201