基于网络拓扑和节点异构的Chord系统

郭松梅 王新生 龚 华 李春风
(燕山大学信息科学与工程学院 秦皇岛 066004)

摘 要 结构化P2P系统在建立逻辑覆盖图时并没有考虑实际的物理拓扑结构,导致覆盖网络与底层物理网络的严重不匹配。另外,结构化P2P系统也没有考虑节点的性能差异,这都影响了系统的路由效率。在结构化对等网络Chord基础上,提出了一种改进的路由算法THChord(Topology and Heterogeneity-based Chord),把物理拓扑相近的节点聚类,并引入超级节点对查询过的信息和热点信息进行缓存。仿真实验表明,THChord的路由性能与Chord相比有了明显的提高。

关键词 结构化对等网络,拓扑匹配,界标簇,聚类,超级节点
中图法分类号 TP393 **文献标识码** A

Chord System Based on Network Topology and Heterogeneity of Nodes

GUO Song-mei WANG Xin-sheng GONG Hua LI Chun-feng
(Department of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China)

Abstract Structured P2P system didn't consider the actual physical topology when building the logical overlay, which would couse a serious mismatching between logical overlay network and physical network. In addition, Structured P2P system didn't attention the heterogeneity of nodes. All of these could affect the routing performance of the system. An advanced algorithm--THChord(Topology and Heterogeneity-based Chord) based on Chord was proposed. It clustered the close physical node, and used super node to cache searched information and hot information. Simulation experiments show that THChord could improve system's routing performance obviously.

Keywords Structured P2P, Topology matching, Landmark binning, Clustering, Supernode

1 引言

近年来,对等计算(P2P)网络得到了业界的普遍关注,已经被越来越广泛地应用于文件共享、对等计算、协同工作、实时通信、广域网络存储等多个领域。P2P资源定位技术是P2P技术领域的基础性关键技术。

P2P网络根据节点信息存储和资源查找的方式主要可以分为3种类型:(1)集中式P2P网络,利用中心索引服务器存储数据的元数据信息,例如Napster[1]。(2)非结构化P2P网络,借助泛洪广播查询请求,例如Gnutella[2]。这两种方案由于设计之初并没有考虑运行在大规模网络环境下,因此扩展能力有限。(3)结构化P2P网络,利用分布式哈希表(distributed hash table,DHT)技术来进行资源定位,例如Chord[3],CAN[4],Pastry[5]和Tapestry[6]。结构化P2P网络因其高效快速的资源定位方式成为P2P网络发展的方向。

结构化P2P系统在形成逻辑覆盖网络时并没有考虑物理网络拓扑结构,所以二者不可避免地会出现不匹配的情况:在逻辑网络中相邻的两个节点在物理网络上可能相距很远,这必然会影响查询响应时间,增大网络的通信代价。另外,通过对现有P2P网络的测量,已发现网络上节点的处理能力和行为方式差别很大[7,8]。这种节点的异构性(heterogeneity)表明,我们应将它们赋予不同的角色,让能力强的节点担当更多的任务。本文在Chord基础上提出一种改进的路由算法THChord,首先利用界标簇算法[9](landmark binning)和RTT(round trip time)探测技术将物理上临近的节点划分为一组,从组中选出一个超级节点来管理组内节点信息,并缓存查询过的信息及热点信息,节点在资源定位过程中首先选择近距离的节点进行路由,从而有效地提高了路由效率。

2 Chord简介

Chord是由MIT提出的一种结构化P2P网络。在Chord系统中节点通过对自身IP地址做哈希运算得到一个m位的标识号。这些节点(最多2^m个)按照标识号从小到大的顺序组成一个逻辑环,环上的每个节点有唯一的前趋节点(predecessor)和后继节点(successor)。节点的资源也通过哈希运算得到一个m位的key,这个key和节点的标识号映射在相同的地址空间内。每个环节点负责那些key小于自身节点但是大于该节点前趋节点的资源。每个节点都要维护一张含m个表项的finger表,节点N的finger表中的第i项是圆环上距离N至少2^{i-1}的第一个节点S,例如S=successor

到稿日期:2008-05-30
郭松梅(1983—),女,硕士研究生,主要研究方向为对等网络;王新生(1949—),男,教授,主要研究方向为计算机网络、信息安全;龚 华(1982—),女,硕士研究生,主要研究方向为对等网络;李春风(1984—),男,硕士研究生,主要研究方向为对等网络。

$(N+2^{i-1})$,$1 \leqslant i \leqslant m$。对任意一次的资源查找请求,Chord 利用 finger 表最多查找 m 次可以定位环上的资源。

图 1 是一个 $m=6$ 且只有 10 个节点的查找示意图,其中节点标识前加上 N 而关键字标识前加上 K 加以区别,图中给出了节点 $N8$,$N42$,$N51$ 的 finger 表。节点 $N8$ 发起的查询 $K54$ 的分组转发路线为:$N8$ 首先找到其 finger 表中节点标识位于 $K54$ 之前且距离 $K54$ 最近的节点 $N42$,然后把查询分组转发给 $N42$,而 $N42$ 按同样的规则找到其 finger 表中的节点 $N51$,并将查询分组转发给 $N51$,节点 $N51$ 发现 $K54$ 落在它的后继节点 $N56$ 之前,即 $N56$ 就是要找的节点,把 $N56$ 返回给节点 $N8$。此后 $N8$ 直接和 $N56$ 建立连接以获取 $K54$ 相应的文件信息。

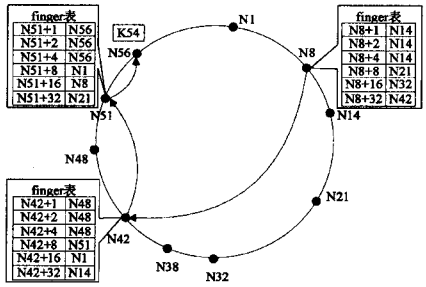

图 1　根据路由表查找 $K54$ 的过程

3　THChord 模型

THChord 的基本思想是:当节点加入系统时,利用界标簇算法来收集系统全局信息,再用 RTT 探测技术收集局部信息,将物理位置相近的节点聚类,从而将网络划分为多个组,每个组中根据节点性能选出一个超级节点(组长节点)及两个超级节点的备份,超级节点除具有普通节点的功能外,还保存组内节点的信息,并对查询过的信息及热点资源信息进行缓存,当进行资源定位时,节点首先在本组内进行查询。

3.1　数据结构

系统中所有节点分为两种:普通节点和超级节点。每个节点保存原 Chord 系统中的 Finger 表外,普通节点维护的表有:(1) Supernode 表,保存所在组的超级节点及其副本节点信息;(2) Bucket 表,缓存一定数目的经过该节点查询的源节点地址信息,表的结构为〈NodeID, IP〉。超级节点除了维护普通节点所有的表,还需要维护的表有:(1) Local 表,保存所有组内节点的信息;(2)Cache 表,缓存一定数目的组内节点查询成功的结果信息,表的结构为〈key, NodeID, IP〉;(3) HotResource 表,缓存一定数目的系统内热点资源信息,表的结构为〈key, NodeID, IP〉。

3.2　组的划分

我们利用界标簇算法来收集系统全局信息,在界标簇算法中,首先在全局中固定几个节点作为界标节点(landmark nodes),每个节点与几个界标节点进行延时测量,然后根据延时按照某种顺序(升序或降序)对界标节点进行排序,所得的序列即为节点的界标序列,具有相同界标序列的节点被认为在物理上也是相近的。但是界标是粗粒度的,对落入同一个区间的节点不能进一步地区分,在最坏情况下,系统中所有的节点可能落入同一个区间中,所以我们再用 RTT 探测技术来划分位于同一界标区域中的节点。

3.3　节点的加入

假设系统中有 m 个已知界标节点,那么就有 $m!$ (m 个延迟作全排列)个界标区域。给每个界标区域分配一个唯一标识 ID,设区域 b_X(b_X 表示节点 X 所在的界标区域) 的 ID 为 id_{bX},使用哈希函数对该 ID 进行哈希运算,得到一个值 $p=hash(id_{bX})$,假设关键字标识 p 存储在 Chord 环上的节点 O 上,则把落入 b_X 区域的所有超级节点的信息存储在节点 O 上,记 O 为 $Node(b_X)$。节点加入或离开时,需要移交该信息给相应的节点。节点加入算法如下:

(1)节点 A 加入系统时,首先按原 Chord 系统加入算法加入到 Chord 环中,并初始化 Finger 表,然后节点加入到一个组中。

(2)节点 A 与 m 个界标节点进行 RTT 探测,A 将所得探测结果按顺序(升序或降序)进行排序,从而确定节点 A 所属的界标区域,记为 b_A。

(3)节点 A 从节点 $Node(b_A)$ 获取落入 b_A(b_A 为节点 A 所属的界标区域)的超级节点信息,假设获取的节点信息组成集合 S_0。

(4)节点 A 对 S_0 中的每个节点进行 RTT 探测,将 RTT 最小的节点记作 B,如果 A 到 B 的探测距离小于某一阈值,则节点 A 加入到超级节点 B 所在的组。

(5)如果探测距离大于某一阈值,则节点 A 自己创建一个新的组,节点 A 作为超级节点,把自己的信息保存到节点 $Node(b_A)$。

当超级节点 A 变化时通知节点 $Node(b_A)$,把新的超级节点信息存入,删除原超级节点 A 的信息。

3.4　节点的退出

节点离开系统时,先按照原 Chord 系统中节点退出算法离开 Chord 环,然后离开其所在的组。对于普通节点来说,节点离开时只需通知超级节点及其副本删除 Local 表中相应信息即可。当超级节点 A 离开时,它会通知组内所有备份节点,由第一个响应的备份节点 B 充当超级节点,并通知节点 $Node(b_A)$保存新的超级节点信息。从组中选择一个普通节点补充作为超级节点备份,通知组内所有节点更新 Supernode 表。

3.5　路由算法

一个节点 N 要查询关键字 key 的路由过程为:

(1)首先判断关键字 key 是否落在它和后继节点(successor(N))之间,如果是就返回后继节点,查询结束。否则转向(2)。

(2)通过 supernode 表将查询发送到超级节点。超级节点查询其 Local 表,Cache 表和 HotResource 表。如果找到就返回节点地址,查找结束。否则返回 key 之前且距离 key 最近的节点,并转向(3)。

(3)节点查询 Finger 表和 Bucket 表,分别找出一个在 key 之前且距离 key 最近的节点。

(4)将(2)和(3)中返回的节点进行比较,选取在 key 之前且距离 key 最近的节点,将查询发送给该节点。

(5)当一个节点收到查询时,它将在 Bucket 表中缓存源节点地址信息。当一个查询完成时,节点通知超级节点及其副本缓存查询成功的节点信息。表满时利用 LRU(least recently used)替换策略管理表中的信息。

3.6 热点资源的管理

在实际的查询过程中，有些节点保存的信息被频繁地查询，形成热点资源。在 THChord 中，给网络中每个节点都设一个计数器来记录该节点保存的信息被访问的次数，当访问次数超过某一个阈值时，我们认为它是热点资源，我们把热点资源的信息发送到各个组，缓存在超级节点及其副本的 HotResource 表里。为了避免与 Cache 表信息冗余，热点资源的信息存入 HotResource 表之前首先检索 Cache 表是否有相同信息，没有则存入。若 HotResource 表已满，则替换旧的热点资源信息。

4 仿真实验和性能分析

4.1 实验设计

仿真实验采用美国乔治大学的 GT-ITM[10] 拓扑生成器生成 Transit-Stub 模式的底层物理网络拓扑。该模型存在多个不同网络距离(时延)级别的域，能够较好地模拟 Internet 网络拓扑结构。在 Internet 网络中，域内网络延迟远远小于域间网络延迟，因此在本实验中，我们设定 transit 域间的网络延迟为 100ms，stub 与 transit 域之间的网络延迟为 20ms，stub 域内的网络延迟为 5ms。实验采用 4 个界标节点，规定每组内有一个超级节点和两个副本节点，Cache 表为 5000 项，Bucket 表为 100 项，HotResource 表为 5000 项。仿真实验主要验证改进后的算法相对于 Chord 在路由跳数及路由延时上的性能改进。实验针对分别由 1000 到 8000 个节点组成的网络进行大量仿真，比较改进前后的性能。

4.2 性能分析

从图 2 中可以看出，改进算法的平均逻辑路由跳数与 Chord 算法相比平均小于 1，这是由于 THChord 中超级节点中包含所有组内节点信息，使得对组内信息的查询可在一跳内完成，并且超级节点对查询过的信息及热点信息的缓存使得多数重复查询和热点查询都能在较少的逻辑跳数内完成。

图 3 给出了在不同节点规模的环境下平均查询延时的分布情况，从图中可以看出，改进算法的平均查询延时与 Chord 相比明显减少，这是由于 THChord 结合了物理拓扑信息来路由，节点进行资源定位时首先在组内进行查找，即优先选择近距离的节点进行路由，而且如果目的节点在超级节点的缓存中被发现，将直接发送消息给目的节点，超级节点和缓存机制共同作用的结果是大大减少了路由开销。

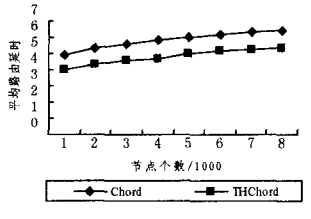

图 2 不同节点规模下的路由跳数比较

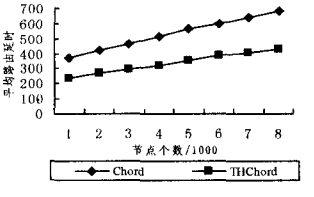

图 3 不同节点规模下的查询延时比较

图 4 是节点数为 5000 时，随着查询次数的增加节点的平均时延的变化情况图。从图中可以看出，随着查询次数的增加，缓存的作用越来越大，平均查询延时越来越小，当查询中重复查询的比例增大时，路由效率的提高会更明显。

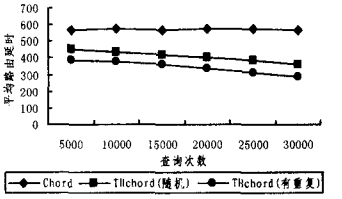

图 4 查询次数增加时节点的平均时延的变化情况

结束语 本文在 Chord 的基础上提出了一种路由优化算法 THChord，该算法考虑到了物理网络拓扑，在资源定位时尽量选择物理上邻近的节点来路由，从另一个方面解决了逻辑网络与物理网络拓扑不匹配的问题，超级节点及缓存策略的引入进一步提高了系统的路由性能。仿真实验表明该算法可以有效地降低路由跳数和网络延时，查询次数越多，重复查询越多，效果越明显。在下一步的工作中将对节点频繁加入退出的系统震荡问题作进一步的研究。

参 考 文 献

[1] Napster - file sharing system [EB/OL]. http://www.napster.com, 2002-11

[2] Gnutella Website [EB/OL]. http://gnutella.wego.com, 2004-09

[3] Stoica I, Morris R, Karger D, et al. Chord: a scalable peer-to-peer lookup service for internet applications [C] // Proc. of ACM SIGCOMM. San Diego, California, August 2001

[4] Ramasamy S, Francis P, Handley M, et al. A scalable content-taddressable network [C] // Proc. of ACM SIGCOMM. New York, 2001

[5] Rowston A, Druschel P. Pastry: scalable distributed object location and routing for large-scale peer-to-peer systems [C] // Proc. of the 18th IFIP/ACM International Conference on Distributed System Platforms. Germany, 2001

[6] Zhao B, Kubiatowicz J, Joseph A. Tapestry: an infrastructure for fault-resilient wide-area location and routing [R]. U. C. Berkeley, Tech. Rep. UCB // CSD-01-1141, April 2001

[7] Saroiu S, Gummadi P K, Gribble S D. A measurement study of Peer-to-Peer file sharing systems // Proc. of the Multimedia Computing and Networking Conference. San Jose, California, USA, 2002

[8] Sen Shubho, Wang Jia. Analyzing P 2 P traffic across large networks // Proc. of the ACM SIGCOMM Internet Measurement Workshop (IMW). Marseilles, France, 2002

[9] Ratnasamy S, Handley M, Karp R, et al. Topologically-aware overlay construction and server selection [C] // Proc. of INFOCOM 2002. 2002

[10] Zegura EW, Calvert KL, Bhattacharjee S. How to model an internetwork // Proc. of the INFOCOM'96. New York: Institute of Electrical and Electronics Engineers, Inc. 1996: 594-602