一种基于分组管理的混合式P2P存储系统

杨 磊 黄 浩 李仁发 李肯立
(湖南大学计算机与通信学院 长沙 410082)

摘 要 利用P2P的方法建立了一个P2P存储系统。以预测的网络距离对参与节点进行分组,形成由超节点维护管理的覆盖网络。使用覆盖网络拓扑结构保持机制、DHT数据存储机制,依据数据访问率不同的数据备份机制和数据修复机制,提高了系统的可靠性和数据存储效率。在仿真实验基础上,验证了该存储系统的性能。

关键词 P2P存储系统,网络距离,分组,副本,带宽

Composite P2P Storage System Based on Group Management

YANG Lei HUANG Hao LI Ren-fa LI Ken-li
(College of Computer and Communication, Hu'nan University, Changsha 410082, China)

Abstract By using the way of P2P, a P2P storage system was built. The nodes involved in the system were grouped by the predicted network distance, and then an overlay network which is maintained by super nodes was formed. By using the overlay network topological structure mechanism of maintenance, DHT data storage mechanism, data replication mechanism based on different frequency of data access and data restoration mechanism, the reliability of the system and the efficiency of data storage in system were promoted. The performance of the storage system was testified on the basis of the results of simulation experiment.

Keywords P2P storage system, Network distance, Grouping, Replication, Bandwidth

P2P存储系统已成为当前一个重要的研究领域。在P2P网络上构建存储系统能有效利用节点间网络带宽、存储空间和数据资源,系统具有良好的可扩展性、数据可用性和可靠性。P2P存储系统的目标是有效组织系统中的节点来存储数据。因此,系统的拓扑结构及其存储数据的可用性和可靠性,与P2P存储系统中节点组织及数据存储备份方式有着极大的关系。

为了对系统节点进行有效的管理,提高系统数据存储的效率及可靠性,提出了一种以预测的网络距离对系统节点进行分组,再使用DHT方法存储数据,最终建立一个覆盖分层的混合式P2P存储系统的设计方案。

1 相关工作

当前基于P2P的存储系统节点组织主要分为两种方式:中央集中式(centrailized)和非中央集中式(decentrailized)。Napster[1]是典型的集中式系统,其利用中央服务器负责目录管理的服务常受服务器的限制,存在单点崩溃导致系统拓扑结构被破坏和由此引起的节点存储数据可用性不高、节点存储数据可靠性下降等问题;Gnutella[2]和Freenet[3]是典型的非集中式系统,由于没有中央服务器,查询数据时以flooding方式将消息传播到网络上,存在占用大量系统带宽、消息泛滥和系统可扩展性差等问题。

针对上述问题,很多研究者采用了对网络节点进行分组管理的方法以期取得更好的性能。文献[4]在网络坐标算法CNP[5]和Vivaldi[6]及DHT算法的基础上,利用等距同心圆簇,对节点二维网络坐标平面进行等面积划分,并根据节点所处区域进行多层命名空间中区间的一一映射,提出了一种分布式的拓扑感知节点聚集算法TANRA,从而有效地保持了P2P存储系统网络拓扑匹配,并降低了节点加入时延,但其应用大多针对流媒体数据。文献[7]和文献[8]分别采用用户行为语义和基于主题划分P2P存储系统的方式改进数据存储方式,提高了P2P存储系统搜索的效率。文献[9]提出了一种基于OPENDHT的节点聚集算法ReDIR,利用抽象出的原语接口put()/get()为上层应用在多层节点命名空间中进行节点分组,但算法没有考虑节点邻近度问题,导致节点间拓扑匹配较差。

2 P2P存储系统节点分组与分层算法

本文将P2P网络中网络距离相近的节点划分到同一分组,以实现对P2P存储系统节点的分组和有效管理。为此,先提出预测系统节点间网络距离的算法。

2.1 网络距离预测算法

P2P系统中两节点间网络距离的测算可通过计算系统中数据传输时延、网络最小带宽和数据传输所经路由跳数等方法获得。对于P2P存储系统中存储数据的可用性和可靠性来说,数据传输时延是一个很重要的因素,而数据传输时延最

到稿日期:2009-03-02 返修日期:2009-06-17 本文受国家自然科学研究基金项目(90715029)资助。

杨 磊(1976—),男,博士生,讲师,研究领域为分布式系统及网络;黄 浩(1981—),男,硕士研究生,研究领域为分布式存储;李仁发(1957—),男,教授,博士生导师,研究领域为嵌入式系统与网络;李肯立(1971—),男,博士后,教授,研究领域为并行处理、科学可视化。

主要的影响因子是传输两节点间的可用带宽。在大规模 P2P 网络中，由于网络节点频繁加入或退出系统导致的高动态性，使得综合考虑数据传输时延、网络最小带宽和数据传输所经路由跳数等因素时，测算网络距离代价过高且难以实现，并且计算系统所有节点的数据传输时延所需代价也是无法承受的。因此，本文决定采用数理统计中随机抽样的方法，通过采集的有限样本信息来预测网络数据传输时延，再利用预测到的较合理的网络数据传输时延对系统节点进行分组。

算法如下：在大规模网络中随机取一节点 k，判断与其相邻节点间网络数据传输时延。假设其有 n 个相邻节点，节点 k 与这 n 个相邻节点间网络可用带宽分别为 $BIT_1, BIT_2, \cdots, BIT_n$，假设传输数据块大小为 $DATA_k(kb)$，$r_i(i=1,2,\cdots,n)$ 为影响因子，可简单取 $r_i=1$。则可测得节点 k 与这 n 个相邻节点间网络数据传输时延分别为 $T_i=r_iDATA_k/BIT_i(i=1,2,\cdots,n)$，再取其网络数据传输时延平均值为

$$D_1=(\sum_{i=1}^{n}T_i)/n=(\sum_{i=1}^{n}r_iDATA_k/BIT_i)/n$$

依此类推，可取得 m 个随机节点的网络数据传输时延的平均值 $D_1, D_2, \cdots, D_m$，假定网络距离的预测值为将所得各采样节点网络数据传输时延的平均值再取平均值，即预测网络距离 $dis=(\sum_{i=1}^{m}D_i)/m$。

2.2 网络节点分组算法

该算法的思想是依次从初始网络中随机选取一个节点，以该节点为圆心，预测网络距离为半径，其内的所有节点都划入一个分组，再将获得分组的节点从初始网络中移除。以此方法循环划分，可将初始网络中大部分节点划入分组。此时，可在各分组节点中选取若干在线时间长、存储容量及网络可用带宽较大的节点作为超节点，考虑到在大规模网络中节点存储容量很难准确判断，故本文中超节点的选取主要考虑节点在线时间的长短及网络的可用带宽。具体方法如下：为确定某分组中超节点，在该分组中随机选取若干样本节点（不少于 50 个），采用定期心跳的方法（让该分组中选中的每个样本节点定期向其对应的目录节点（已指定好）报告自己的存在状态，如果对应的目录节点一段时间没有收到心跳，则认为该样本节点下线，并记录下该样本节点上次在线时的时间 $time_i$）确定节点在线时间长短；确定各样本节点的网络可用带宽的方法为：假设某样本节点有 n 个相邻节点，样本节点与这 n 个相邻节点间网络可用带宽分别为 $BIT_1, BIT_2, \cdots, BIT_n$，则取该样本节点的平均网络可用带宽 $B_i=(\sum_{i=1}^{n}BIT_i)/n$ 为该样本节点网络可用带宽。设节点在线时间长短及网络可用带宽对其被选为超节点的影响因子分别为 m_1 和 m_2（其中 $m_1+m_2=1, 0<m_1<1, 0<m_2<1$），则 $Snode_i=m_1\times time_i+m_2\times B_i$ 中 $Snode_i$ 较大即可确定其为该分组超节点。以此方法，在各个分组中选取若干次，可在每个分组确定若干个超节点，各分组中超节点负责将该分组中其附近普通节点组织起来。在初始化分组算法中，圆心节点选取的随机性必然导致少量剩余节点未划入分组，因此，考虑剩余节点到各分组超节点间的网络距离平均值，将取值最小的分组加入并连接到该分组网络距离最小的超节点下。至此，整个初始网络中全部节点分组完毕。

算法如下：假设 S 为当前初始网络中所有节点的集合，dis 为预测的网络距离，g 为分组个数，$T(n,j)$ 为节点 n 和节点 j 之间的网络数据传输时延，$C[n]$ 为一个分组。

```
for(i=0;i<g;i++)
        //每次循环得到一个分组，共分 g 组
{ 从初始网络中选取一个节点 n;
        //以此节点为圆心进行分组
for(;j∈S,j≠n;;)
        //以预测的网络距离为半径分组
{ if(T(n,j)≤dis)
  { 添加节点 j 到分组 C[n];
    S=S-{j};
  }
}
}
在分组中选取超节点;
while(S 非空)
//对未加入分组的剩余节点添加到网络距离最近的分组
{
  for(k∈S,i=0;i<g;i++)
      {
    计算节点 k 到各分组超节点的网络距离;
  }
    计算节点 k 到各分组超节点的网络平均距离;
    添加节点 k 到网络平均距离最小的分组;
}
```

2.3 覆盖网络分层及拓扑结构保持机制

为更好地对已分组节点进行管理，利用了超节点来对已分组的系统进行分层管理。首先，对每个分组内超节点采用 chord[10] 方式组织，形成一个环形拓扑结构，各超节点附近的普通节点均与该超节点连接，各超节点所属普通节点之间采用随机组合方式连接。为确保各分组间的连通性，从各分组内的超节点中随机选取一个与其它分组的一个超节点连接，形成环形拓扑后，这些超节点又彼此交互连接，最终形成一个内部环结构。由此，将存储网络划分为一个三层覆盖网络，如图 1 所示。

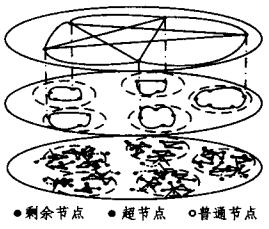

图 1 三层覆盖网络图

对于 P2P 存储覆盖网络，当网络拓扑结构发生变化时，其分层的拓扑结构的保持对存储系统性能优劣非常重要。为此，我们提出一种网络分层拓扑结构保持的策略。具体方法是：对分组中形成 chord 环路的第二层超节点保存至少一个备份节点（可从 2.2 节选取超节点方法中选取 $Snode_i$ 较大的节点作为备份节点）；对于形成第三层内部环结构的超节点，考虑到其对各个分组之间保持连通的重要性，为其保存至少两个备份节点，如图 2 所示。其中每个超节点和其备份节点均记录该超节点的 2 个前继超节点和后 2 个后继超节点的信息。这样一旦该超节点失效，可立即启用其备份节点替代其位置；一旦该超节点的前继或后继超节点失效，可立即使用失效超节点的前继或后继超节点作为自己现在的前继或后继节点，以保持环路拓扑结构的完整性。由理论分析可知，假设每

个节点失效概率为1/2，仅当连续两个超节点及其备份节点同时失效(概率至多为$1/2^4$)第二层拓扑结构才可能出现中断的情况；仅当超节点及其备份节点同时失效(概率至多为$1/2^3$)第三层拓扑结构才可能出现中断的情况。可见使用该方法后，覆盖网络拓扑结构完整性被破坏的概率很低。

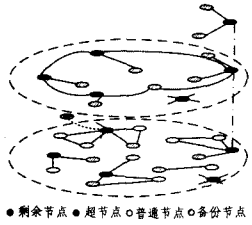

图2 分组内覆盖网络拓扑保持图

3 数据存储与备份机制

将网络距离相近的节点分组并利用超节点管理的方法层次化后，系统便可充分利用节点间网络距离的邻近性。为使P2P存储系统存储数据具有更好可用性和可靠性，提出了数据的存储和备份机制。

3.1 数据存储机制

本文采用DHT方式进行数据存储。具体方法是：首先将网络中各节点根据其标志(IP地址或域名)按规定的Hash算法映射成一长度为M位(bit)的全球唯一标志符NodeID，并将其唯一分配给该节点；对系统中需要存储的数据根据其标志(文件名)按规定的Hash算法映射成一个长度为M位(bit)的全球唯一标志符DataID，再将每个数据对象都存储在与该对象DataID相近的NodeID节点上。为此，分组内各超节点与其备份节点维持着一张与其相连普通节点NodeID相关的信息索引表(为降低超节点负担，各超节点与其备份节点不存储数据，仅维持一张普通节点NodeID信息索引表，当超节点负担过重时，可使用备份节点作agent)，第三层超节点NodeID设置为网络中已知成员节点，可为数据存储查询提供其所知的信息。在数据存储查询时，只要知道所查询数据的DataID及第三层超节点NodeID的位置，查询超节点中与其相连节点NodeID信息索引表就可找到存储该数据的节点。具体方法是：先在该第三层超节点所在分组内进行路由查询，如相近NodeID不在本分组内，此时可以通过第三层超节点内部环结构将此数据DataID广播到其它分组超节点之上，再采用相同的组内查询方法，最后总能为需要存储的数据找到相近NodeID的普通节点(假设网络节点总数为N，网络被划分为g个组，一个分组中节点数量大致为$N/g(N/g<<N)$，且分组内各超节点均维持着各自相连普通节点NodeID，其数据存储位置查询时延代价将非常小)。

考虑到各节点存储空间有限，当某节点存储的数据超过设定的阈值$TH1$时，将其最近最少访问的数据转存到其它存储空间未超过阈值$TH1$的相连普通节点上，并将此消息通知与其相连的超节点。分组内，各超节点采用定期心跳机制确认其所连普通节点是否失效，当某普通节点失效时，删除其在超节点和备份节点中的NodeID索引项；如超节点失效，备份节点将立刻取代其位置。当有新节点需加入系统时，先为其确定一个全球唯一标志符NodeID，再按剩余节点选择分组的方法确认其加入的分组及所连超节点，加入成功后，在所属超节点信息索引表中注册其NodeID即可。

3.2 数据备份机制

数据备份的目的是为了提高存储系统数据的可用性和可靠性，使系统不因某些节点失效而丢失其存储的数据。文献[11]的统计分析表明，在特定分组子网中，数据查询过程的重复率很高，这是因为网络存在一些存储着热点数据的高访问率节点。这些高访问率的热点数据的备份，对整个系统数据可用性和可靠性的提高起着重要作用。因此，本文按数据访问率的高低采用不同的数据备份机制。当系统节点上数据存储成功且运行一段时间后，对于访问率低于设定阈值$TH2$的数据，考虑到备份数据时所需带宽和路由开销对整个系统数据可用性和可靠性的影响，采用组内备份的方式。具体方法是：需备份数据节点随机均匀的选择n个(该参数为创建副本的数目，需用户预先设定)相连组内节点，在这些节点上创建该数据的副本，并将此信息写入与其相连的超节点信息索引表中；对访问率高于设定阈值$TH2$的数据，采用组内备份和组间备份的方式，即对访问率高的数据有选择的备份到其它的分组节点中，再将其备份信息写入与其相连的超节点信息索引表。由于存储网络采用的是覆盖分层的超节点管理拓扑结构，当有节点存储数据失效时，系统可通过查询信息索引表的方式在组内或组间快速找到所需备份数据信息，且组间备份方式较组内备份方式容灾性更强，更适合于大规模存储系统中重要数据信息的备份。

3.3 数据修复机制

由于P2P存储网络的高动态性，当系统节点频繁加入和退出时，节点存储数据及其副本可能部分丢失甚至因受损而无法还原，从而降低系统存储数据的可用性及可靠性。为保证数据的可靠性并使其始终存在一定数目的副本，我们设计的P2P存储系统将周期性地检测并修复丢失的数据及其副本。检测过程由系统各分组的超节点周期性地检测其所属普通节点所存数据及其备份的副本数目(为降低超节点负担，也可考虑由各普通节点周期性地读取保存其数据副本的节点中其备份的副本，看其是否存在)。修复分为两种情况：存储数据的修复和备份副本的修复。存储数据的修复是当检测出某普通节点的存储数据丢失或受损后，立即启用其副本备份节点来修复其丢失或受损的数据；为降低因立即修复副本而增大网络流量使系统开销增加，当节点存储数据保存完好，其备份副本因受损减少的修复，仅当备份副本数目减少到一半时才进行副本的修复。

4 仿真实验和性能分析

MIT的chord算法采用一维环形空间结构组织节点，该算法思路简洁清晰，性能较好，具有很高的应用价值。但在chord系统中，逻辑相邻的节点物理位置未必相邻(逻辑相邻节点之间物理位置可能很远)，从而导致在大规模网络系统中，当有节点频繁加入退出系统时，系统路由效率降低，节点存储数据的一致性和可靠性下降，节点单位时间内处理消息数过多等问题。本文设计算法将系统中网络距离相近的节点划分到同一分组由超节点管理，使逻辑位置相邻的节点物理位置(节点网络距离)也相近。因此，对chord系统与本文算法从数据存储定位的路由查询跳数和节点加入退出系统时系统平均处理消息数这两个指标进行对比分析。

在 Intel Core 2 Duo 1.73G 处理器和 Linux Redhat 9.0 环境下，使用 NS-2 网络仿真器对实际网络情况进行模拟，通过 Otcl 和 C++程序设计分别实现了 Chord 和本文算法，并使用 GT-ITM 拓扑发生器生成了网络拓扑图。为了尽可能模拟实际网络情况，结合文献[12,13]结论和当今网络实际情况将节点所拥有带宽设置为 4 个等级。其中约 5%的节点模拟高级用户拥有 100Level 带宽，15%的节点拥有 10Level 带宽，60%的节点模拟大部分用户使用的 DSL 上网方式拥有 1Level 带宽，20%的节点模拟拨号上网的用户拥有 0.1Level 带宽；各等级节点在线时间随机取值，其在线时间单位为小时。

分别考虑 1000，2000，3000，4000，5000，6000，7000，8000，9000，10000 个节点的网络，取影响因子和数据块大小 $r_i=1$，$DATA_k=10000(kb)$进行分组，分组后，每个分组取 50 个节点按影响因子 $m_1=0.5$，$m_2=0.5$ 按公式 $Snode_i=m_1\times time_i+m_2\times B_i$（$B_i$取 Level 值）选取超节点及其备份节点。为得到更准确的仿真结果，对不同节点个数的系统各重复进行 20 次实验再取平均值。

本文算法与 chord 系统数据存储定位路由查询跳数实验结果如图 3 所示。由图 3 可以看出，本文算法对数据存储定位的路由查询跳数比 chord 算法有了较大改进。分析认为，本文设计的系统中，各分组内节点个数远小于 chord 系统中节点个数，且节点间网络距离较近，在超节点管理下进行查找必然能有效提高找到目标节点的效率。

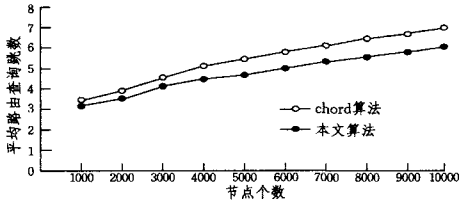

图 3　平均路由查询跳数对比

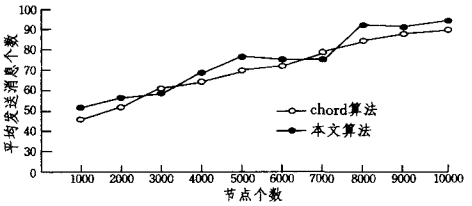

图 4　节点加入系统平均发送消息个数对比

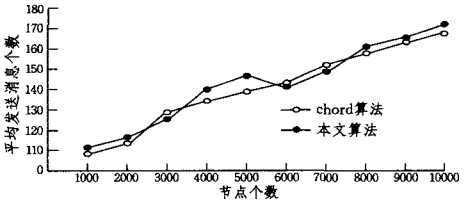

图 5　节点退出系统平均发送消息个数对比

本文算法与 chord 系统节点加入和退出系统时系统平均处理消息数实验的结果如图 4 和图 5 所示。由此可见，随着网络节点数量的增加，本文设计的算法在节点加入和退出系统时系统平均处理消息数有时较 chord 系统略大，特别是节点规模为 5000 和 9000 时，节点加入系统时平均处理消息个数增幅较大。分析认为，这是由于本文算法较 chord 算法在维护系统拓扑结构时更加严密，节点加入系统时在超节点的注册及其备份比 chord 系统更加频繁且复杂而导致的结果。在节点规模为 3000 和 8000 时，本文算法较 chord 系统平均处理消息数略低。分析认为，这是因为在系统分组和选取各分组的超节点阶段，较好的样本节点采样，能更真实地反映系统的实际情况，因此构建的混合系统在处理节点加入或退出系统时的开销降低。

总之，上述实验结果有效证实了随着系统规模的增大，本文设计系统的数据存储定位的路由查询跳数较 chord 系统有了较大改进，而本文算法所带来的系统维护开销却与 chord 系统维护开销相差不大，在可接收范围之内。

结束语　本文详细描述了一种建立在 P2P 上的分布式存储系统设计方案。提出了一种按预测网络距离对节点进行分组，在超节点管理下建立三层覆盖网络的机制，并给出了有效保持这种拓扑结构的方法。在此基础上，给出了一种按 DHT 方式的数据存储机制，并为之设计了详尽的数据副本备份和修复方法，为存储数据的可用性和可靠性提供了有效保障。在后续工作中，我们将对 P2P 存储系统中副本和纠删码[14]相结合的备份机制的运用、节点间 Byzantine[15]错误的处理及系统各影响因子、阈值的更优化组合做进一步研究。

参 考 文 献

[1] C-NET NEWS. Napster among fastest - growing Net technologies. 2000

[2] Gnutella. http://www.gnutella.com/ 2003

[3] Clarke I, Sandberg O, Wiley B, et al. Freenet: A distributed anonymous information storage and retrieval system[C]// Workshop on Design Issues in Anonymity and Unobservability. 2000: 25-31

[4] 段翰聪，卢显良，唐晖，等. 基于 DHT 的拓扑感知节点聚集算法[J]. 计算机研究与发展，2007，44(9)：1557-1565

[5] Ceccanti A, Jesi G P. Building latency-aware overlay topologies with quickPeer[C]// Joint Int'l Conf on Autonomic and Autonomous Systems and International Conference on Networking Services. Papeete, Tahiti, 2005

[6] Nakao A, Peterson L, Bavier A. A routing underlay for overlay networks[C]// IEEE SIGCOMM. karlsruhe, Germany, 2003

[7] 邱志欢，肖明忠，代亚非. 一种基于 P2P 环境下的基于用户行为的语义检索方案[J]. 软件学报，2007，18(9)

[8] 傅向华，冯博琴，马兆丰，等. 基于主题划分的有组织 P2P 搜索算法[J]. 西安交通大学学报，2005，39(12)

[9] Rhea S, Godfrey B, Karp B, et al. OpenDHT : A public DHT service and its uses[C]// IEEE SIGCOMM. Philadelphia, USA, 2005

[10] Stoica I, Morris R, Karger D, et al. Chord: A scalable peer-to-peer lookup service for Internet a pplications[C]// Annual Conf. of the Special Interest Group on Data Communication (SIGCOMM 2001). 2001: 124-137

[11] Ratnasamy S, Shenker S, Stoica I. Routing algorithms for dhts: Some open questions[C]// Proc. of the IPTPS02. Cambridge, 2002

[12] Chawathe Y, Ratnasamy S, Breslau L, et al. Making gnutella-like P2P system s scalable[J]. ACM SIGCOMM, 2003

[13] Saroiu S, Gummadi P K, Gribble S D. A Measurement Study of Peer-to-Peer File Sharing Systems[C]// Proceedings of Multimedia Computing and Networking 2002 (MMCN'02). San Jose, CA, Jan. 2002

[14] 田敬，代亚非. P2P 持久存储研究综述[J]. 软件学报，2007，18(6)：1379-1399

[15] 杨磊，黄浩，李仁发，等. P2P 存储系统拜占庭容错机制研究[J]. 计算机应用研究，2009，26(1)