二元位运算P2P系统复制技术的研究

廖剑伟[1]，蔡洪斌[2]，熊海灵[1]，陈善雄[1]

(1. 西南大学计算机与信息科学学院 重庆 北碚区 400715; 2. 电子科技大学计算机科学与工程学院 成都 610054)

【摘要】提出了BitwiseTree，利用二元位运算确定复本放置，而不需要像其他基于日志的复制技术需要考虑客户访问日志；同时BitwiseTree提供的容错机制保证失效节点中的文件可以在其他复制节点中找到。仿真实验表明，BitwiseTree只需要使用较少的复本就能达到有效地缓解节点过载的目的，因此该模型非常适合无法获得客户访问日志的机密P2P系统或者对系统性能要求比较高的P2P系统。

关 键 词 均衡; 二元位运算; 容错; P2P系统; 复制树

中图分类号 TP 393 **文献标识码** A doi:10.3969/j.issn.1001-0548.2009.03.021

Replication in Peer to Peer System Based on Bitwise Operation

LIAO Jian-wei[1], CAI Hong-bin[2], XIONG Hai-ling[1], and CHEN Shan-xiong[1]

(1. College of Computer and Information Science, Southwest University Beibei Chongqing 400715;

2. School of Computer Science and Engineering, University of Electronic Science and Technology of China Chengdu 610054)

Abstract Replicating techniques are frequently used in a high performance distributed system to reduce the load of the overload nodes. Traditional file replication algorithms rely on the analysis of client-access logs to determine the location of the replicated nodes. This paper presents a fault-tolerant BitwiseTree model for peer to peer system. It constructs a replication tree for each node, and then uses bitwise operations to determine the location of the replicated node without any client-access history. In addition, each replication is guaranteed to reduce the workload of the replicating node by half. The experimental results show that BitwiseTree successfully and efficiently reduces the load of overloaded nodes.

Key words balancing; bitwise operations; fault tolerance; peer to peer system; replication tree

P2P系统面向广域网提供大规模网络共享服务，请求响应时间是P2P系统性能的重要表现[1]。当一个节点处理请求数目超过其所能处理上限时，导致响应时间的延长，该节点成为超载节点。复制技术被广泛用于减轻负荷超载节点和改进系统性能，传统的复制技术确定复本存放位置时，需要分析客户访问日志，而基于日志复制技术需要诸如磁盘和内存等系统资源。分析客户访问日志记录一方面增加了CPU的负担，另一方面增加了I/O和网络通信量，影响系统的性能[2-4]。

本文提出如何构建一个高性能、负载平衡、具有容错性的P2P系统，当节点过载时不需要获取以及分析客户的访问日志，而是通过二元位运算确定复本的存放。为了实现系统具有容错性的目标，将树型结构的P2P系统划分为2^b棵独立子树，$N \leqslant 2^m$，$0 \leqslant b \leqslant m$，其中$N$为P2P系统的节点数；$m$和$b$分别是满足条件的自然数。任何文件都在子树中存放有一个复本，使得系统容错度为2^b。

1 系统基本模型

BitwiseTree是一个建构在P2P系统查找协议之上的复制模型，P2P系统中的每个节点拥有唯一的物理标识PID，用$P(i)$表示PID=i的节点。在BitwiseTree算法中，利用P2P系统查找协议将信息从一个节点发送到其他节点，将需要发送的信息和目标节点PID作为输入信息，通过查询内部路由表将信息送达目标节点[3]。

模型中每个节点与其他节点通过复制树进行关联，在具有N个节点的P2P系统中的复制树是一棵具有N个节点的树。客户文件访问请求目标节点为复制树的根节点，如果该根节点过载时，将该文件访问请求定位到其他存放副本的节点。

1.1 层次结构的复制树

BitwiseTree方法为具有N个节点的P2P系统建构

收稿日期：2008 –03 – 15；修回日期：2008 – 10 – 15

基金项目：国家863计划(2006AA01Z335)

作者简介：廖剑伟(1981 –)，男，博士生，讲师，主要从事网络计算和分布式计算方面的研究.

N棵具有不同根节点的复制树，复制树中的每个节点标识符为VID。为了进行区分，VID用二进制形式表示，而PID用十进制表示，如图1所示。

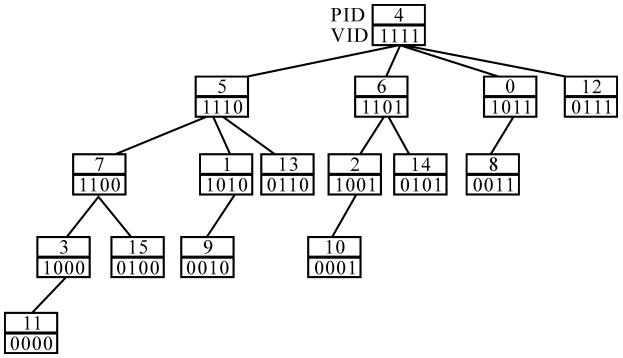

图1　$P(4)$为根的16个节点复制树

复制树的相关性质为：

(1) 节点的VID最左边连续比特位为1的个数就是该节点拥有的子节点数，各子节点的VID是将其父节点的VID值中最左边(高位)连续为1的比特位由1变为0而得到的。

(2) 将给定节点的VID左边(高位)第1个值为0的比特位置为1，即得到其父节点的VID。

(3) 节点i比节点j拥有更多的子节点，当且仅当i节点的VID值比节点j的VID值大，根据$A\oplus B=C$可以推导出$A\oplus C=B$，得到性质(4)。

(4) 给定复制树根节点的PID时，可以实现复制树中任意节点PID值与VID值的转换。

构建$P(k)$的复制树首先对k的二进制位取反，用k表示；通过$k\oplus$节点VID得到节点的PID，其中$\oplus$表示异或操作。每个VID可以映射为不同复制树中PID不同的节点。为了描述方便，定义ψ为哈希函数，输入参数是一个字符串，可以得到输出结果为0～(2^m-1)的值。

1.2　文件操作

每次请求访问文件操作只需请求节点PID和接收请求节点(存放文件节点)PID，根据$r=\psi(f)$确定目标节点的PIDr，其中f是表示请求文件的唯一标识(如URL)。本文分别介绍文件的插入、读取和复制操作。

1.2.1　插入文件

当节点$P(k)$接收到将文件插入到目标节点$P(r)$的请求后，根据P2P系统提供的查询协议确定节点$P(r)$是否活跃。如果是活跃节点，$P(r)$通过P2P系统本身所提供的消息传递机制，将该请求传送到该目标节点$P(r)$；接着由$P(r)$节点利用AdvancedInsertFile算法完成文件的本地存储操作；如果节点$P(r)$已经失效，$P(k)$将该请求转发给$P(r)$的复制树中某活跃节点，再完成文件插入。

插入文件算法AdvancedInsertFile(f)的部分伪代码为：

(1) $r=\psi(f)$

(2) FindLiveNode(r,r).CreateFile(f)

选择活跃节点算法FindLiveNode(s,r)的部分伪代码为：

(1) if $P(s)$ is alive

(2) then return $P(s)$

(3) else s.VID $\leftarrow r\oplus s$

(4) for $i\leftarrow$ s.VID − 1 to 0

(5) do $p\leftarrow r\oplus i$

(6) if $P(p)$ is alive

(7) then return $P(p)$

(8) return false

1.2.2　读取文件

节点$P(k)$接收到读取节点$P(r)$中文件f 的请求，首先检查自身有无文件拷贝，如果有则直接返回文件f 复本给客户；否则计算$P(k)$在$P(r)$复制树中其父节点的PID，并通过消息传递机制将该请求发送给父节点，重复该过程直至找到请求文件或者到达节点$P(r)$。计算节点$P(k)$在$P(r)$复制树中父节点PID的算法如下：

(1) 通过性质(4)在复制树中找到$P(r)$的VID；

(2) 通过性质(2)得到其父节点的VID；

(3) 通过性质(4)将父节点的VID转换为PID；

读取文件算法ReadFile(f)的部分伪代码为：

(1) $k\leftarrow$this.PID

(2) if this.FileExisted(f)=true

(3) then this.ReturnFile(f)

(4) else FP_k^r.ReadFile(f)

算法中，this表示本节点； FP_k^r 表示在$P(r)$的复制树中节点$P(k)$的父节点的PID。

1.2.3　复制文件

复本技术可以缓和过载节点负载问题，因此，系统中失效节点以及决定失效节点中文件复本应该存放位置也是必须考虑的一个问题。

考虑如图2所示的一个例子：节点$P(4)$的孩子列表为{$P(6)$,$P(7)$,$P(1)$,$P(12)$,$P(13)$,$P(8)$}，根据节点的VID排序，其中虚线节点表示失效节点。通过调用 $V_r^r(f)$ 将文件f复制到$P(r)$孩子列表中的第一个没有该文件复本的节点中。

节点$P(k)$由于过多操作文件f的请求导致过载时，$r=\psi(f)$，且$k\neq r$。首先通过FindLiveNode(r,r)算法在$P(k)$的复制树中查找是否存在某个节点的VID

比$P(k)$的VID大。如果存在这样的节点，则说明节点$P(k)$的过载主要是由于其子孙节点都将操作文件f的请求转发给其自身引起的，通过调用$C_k^r(f)$表示把文件f复制到$P(k)$的孩子节点列表中的某个节点，便可以有效地解决$P(k)$的过载问题；如果没有找到这样的节点，则说明节点$P(k)$的过载主要是由于所有操作文件f的请求转发给它而引起的，那么同样可以通过调用$C_k^r(f)$表示把文件f复制到$P(k)$的孩子节点列表中的某个节点，便可以有效地解决$P(k)$过载问题。因为没有客户访问日志，所以不能准确地确定存放复本的孩子节点，在BitwiseTree中是根据合并节点r和节点k的孩子列表并依照VID排序，将文件拷贝到孩子列表中没有文件f复本的第1个节点。

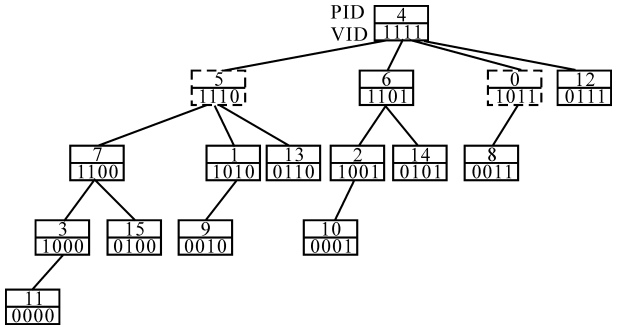

图2 具有14个节点的$P(4)$的复制树

2 容错模型

容错模型利用具有m比特的节点VID中的b位来实现容错，每个节点VID使用1个0～(2^m-1)标识符表示，然后将该生成的树结构划分为b棵不相交的子复制树，子树中节点的VID的低b位是子树标识，而高$m-b$比特表示节点在某棵子树中的标识。具有16个节点且b=2的容错复制树模型如图3所示。

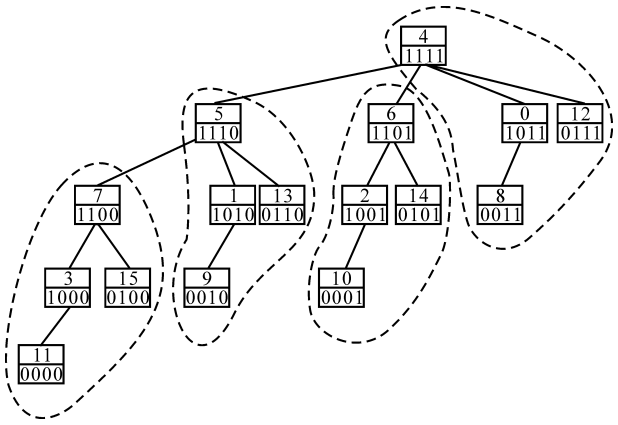

图3 具有16个节点且当b=2时的容错模型

具有N节点的系统中节点$P(k)$接收到将文件f插入请求，其中$N\leqslant 2^m$且要求系统具有度为2^b容错能力。假设$r=\psi(f)$，首先将$P(r)$的复制树划分为2^b棵子树，然后通过修改FindLiveNode算法确定在每棵子树中的目标节点，利用性质(4)获得每个目标节点的PID，最后将文件插入请求转发到各个目标节点，使整个系统具有2^b的相同拷贝。当某个节点接收到读取文件请求，那么该节点首先在其子树中进行查找，如果该请求无法满足要求(即产生错误，在该子树中没有发现所请求文件的拷贝)，那么该节点将读取文件请求转发到其他子树中，直到找到文件复本或者所有的2^b子树都没有找到文件复本，系统提供了度为2^b的容错能力。

3 仿真和比较

为了更好地说明BitwiseTree的性能，本文利用GnuSim[5]构建了模拟器。GnuSim是一个通用Gnutella和非结构化P2P网络仿真器，该仿真器可以验证在P2P网络中使用的各种模式，并评估其性能和价值，以及P2P网络中非可控负载的重要性。另外，GnuSim还提供了一些变量来仿真故障率现象。在仿真实验中，设置每个节点最大负载为每秒处理100个请求，如果超过该请求数则认为该节点是过载节点，可以新建复本节点以减少其负载。只有当系统中没有过载节点时，该系统才能称为负载均衡的系统。实验中设定表示节点VID的参数m=10，b=0，测试在一台酷睿双核T2300(1.66 GHz)、内存512 MB的主机上完成。

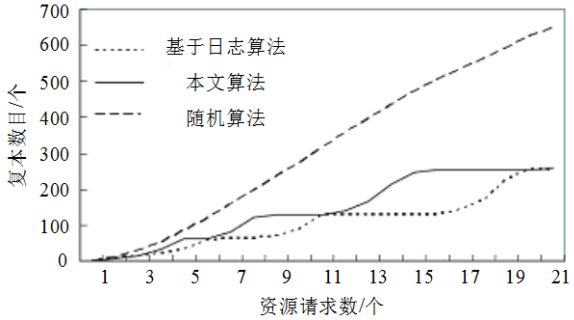

图4 3类方法的仿真比较

BitwiseTree、随机复制方法和基本日志方法的比较如图4所示[6-9]。

3种方法都利用树形结构解析查询请求，系统初始化时所有文件都只有1个原本，系统中的随机节点对文件的请求为1 000～20 000 次/s。随机复制方法是指当1个节点过载时，随机地将其上的文件复制到其他节点中；基于日志方法通过分析客户访问日志，判断将文件复制到哪个节点中。BitwiseTree使用较少的复本就可以达到负载均衡的要求，并根据计数机制删除一些访问量较少的复本。

BitwiseTree系统中有10%、20%和30%的死亡节点时，达到负载均衡分别所需要的复本数目如图5所示。

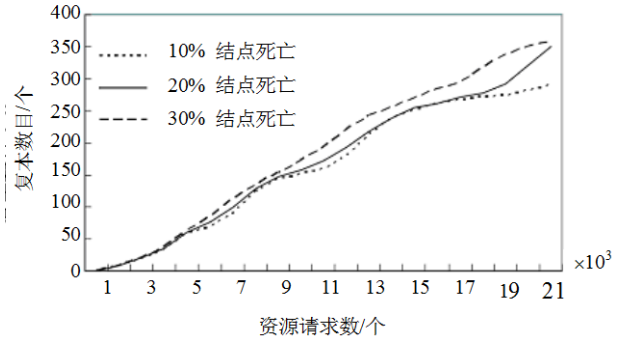

图5　BitwiseTree方法中出现死亡节点的仿真结果

系统中80%的节点用来发送查询请求，而20%的节点用来接收和处理这些请求的确定性。模型中3种不同算方法的比较如图6所示。

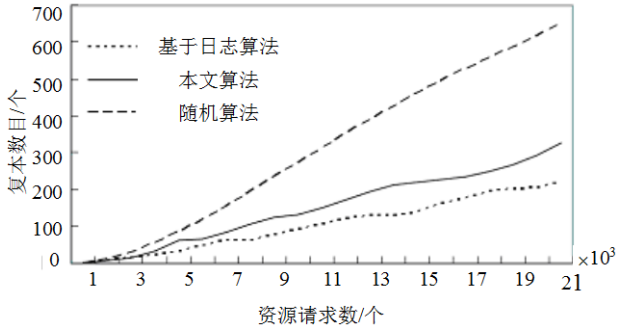

图6　确定性模型中3类方法的仿真比较

通过仿真实验中的数据可以说明，BitwiseTree不需要获取和分析客户访问日志，通过二元位运算确定复本存放的位置即可达到较为理想的负载均衡的要求。该方法在无法获得客户访问日志的机密P2P系统中或者对系统性能要求比较高的情况时是非常有效的。

4　总　结

本文提出了不需要分析客户端日志进行复本存放的BitwiseTree算法，通过二元位运算来确定复本应该存放位置，提供容错机制保证用户需要存取在已离开节点中的复本也能够在其他节点中找到。仿真实验表明，与传统简单的复本存放和需要分析客户端日志记录的复本存放技术比较，BitwiseTree只需要使用较少的复本就能成功有效地减少原本负载过重的节点的工作量，达到负载均衡的目标，从而提高系统的性能。

参　考　文　献

[1] ABERER K, DATTA A, HAUSW1RTH M. Efficient, self-contained handling of identity in peer-to-peer systems[J]. IEEE Transaction on Knowledge and Data Engineering, 2004, 16(7): 858-869.

[2] KALOGERAKI V, ZEINALIPOUR Y, Gunopulos D, et al . Distributed middleware architectures for scalable media services[J]. Journal of Network and Computer Applications, 2007, 30(1): 209-243.

[3] CHOU J, HUANG Tai-yi. A scalable and load-balanced lookup protocol for high-performance peer-to-peer distributed systems[C]//Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA). Las Vegas, Nevada, USA: CSREA Press, 2003: 708-712.

[4] 侯孟书, 卢显良, 周　旭, 等. 非结构化P2P系统复本研究[J]. 小型微型计算机系统, 2005, 26(11): 1903-1906.
HOU Meng-shu, LU Xian-liang, ZHOU Xu, et al. Study on replication in unstructured P2P system[J]. Mini-micro Systems, 2005, 26(11): 1903-1906.

[5] KARAKAYA M, IBRAHIM K L, ULUSOY O Z. GnuSim: a general purpose simulator for GNUTELLA and unstructured P2P networks[R]. Bilkent University, 2005.

[6] SAVITHA K, KARTHIKEYAN V, MARIO L. DiST: a scalable, efficient P2P lookup protocol[J]. Lecture Notes in Computer Science, 2005, 3601: 40-53.

[7] LU Xi-cheng, ZHENG Qian-bing, ZHU Pei-dong, et al. An efficient long query path driven replication strategy in unstructured P2P network[C]//Proceedings of 4th International Conference on Networking (LNCS). [S. l.]: Springer, 2005: 793-799.

[8] WANG Yun, WANG Jun-ling. Load balancing framework for actively replicated servers[J]. Journal of Southeast University, 2005, 21(4): 419-426.

[9] 张　虎, 董小社, 伍卫国, 等. 一种基于日志合并优化的数据同步机制[J]. 小型微型计算机系统, 2006, 27(12): 2183-2188.
ZHANG Hu, DONG Xiao-she, WU Wei-guo, et al. A data synchronization mechanism based on log-merging optimization[J]. Journal of Chinese Computer Systems, 2006, 27(12): 2183-2188.

编　辑　熊思亮