文章编号: 1673-0062(2009)04-0062-04

P2P存储系统节点分层管理策略研究

万亚平[1,2], 刘 立[1], 罗 扬[1], 肖建田[1]

(1.南华大学 计算机科学与技术学院, 湖南 衡阳 421001; 2.华中科技大学 计算机科学与技术学院, 湖北 武汉 430074)

摘 要: P2P存储系统具有自治性、容错性和可扩展性等特点. 但由于节点的高度动态特性导致 P2P存储系统的应用面临巨大挑战. 数据分发和拓扑结构的设计是其中的一个关键问题. 本文结合动态故障检测机制设计了一种分层的节点管理策略. 结果表明本设计方法具有较好的性能, 能高效的对节点进行管理, 减少了网络中由于故障检测而导致的大量的心跳数据包, 提高 P2P存储系统的可用性.

关键词: 可用性; 拓扑; 数据分发; 对等存储系统
中图分类号: TP393.02 文献标识码: A

Study on Node Management Strategy of P2P Storage System

WAN Ya-ping[1,2], LIU Li[1], LUO Yang[1], XIAO Jian-tian[1]

(1. School of Computer Science and Technology, University of South China, Hengyang, Hunan 421001, China;
2. Computer Department, HuaZhong University of Science and Technology, Wuhan, Hubei 430074, China)

Abstract: P2P storage system have a lot of attractive advantages, such as self-organization, scalability and fault tolerance. The applications of P2P storage system faces many challenges because of high dynamic characteristics of nodes. Data distribution and the design of underlying topology are the key. This paper design a hierarchical node management strategy combining dynamic failure detection method. The results show that this design can reduce network overhead and CPU occupancy rate and improve availability of P2P storage system.

Key words: availability; topology; data distribution; P2P storage systems

在 P2P(Peer-to-Peer)上构建存储系统能有效利用网络带宽、存储空间以及计算资源. 目前已经存在一些原型系统的设计, 如 OceanStore, CFS[1], BitVault, PAST[2]等. 本文设计了一种半结构化的多层树型管理策略, 其节点部分冗余、分级管理、各负其责, 有效利用了各个节点计算资源、存储资源以及系统的网络带宽. 同时, 为保证节点之间通信畅通, 采用了动态的故障检测机制, 它能

收稿日期: 2009-06-25
基金项目: 国家 973基础研究计划专项基金资助项目(2004CB318201); 湖南省教育厅基金资助项目(08C756)
作者简介: 万亚平(1973-), 男, 湖南株洲人, 南华大学计算机学院讲师, 博士研究生. 主要研究方向: 网络存储, 分布式系统, 故障检测技术.

根据网络状况的变化而动态的改变检测的时限.

1　背景及相关工作介绍

1.1　P2P拓扑结构

拓扑结构是指分布式系统中各个组成单元之间的物理或逻辑互联关系, 节点之间的拓扑结构是确定系统类型的重要依据. 当前 P2P系统主要有四种拓扑结构: 中心化拓扑; 全分布式非结构化拓扑; 全分布式结构化拓扑和半分布式拓扑.

1)中心化拓扑最大的优点是维护简单发现效率高. 问题是容易造成单点故障, 访问的“热点”现象和法律等相关问题, 如著名的 MP3共享软件 Napster

2)全分布非结构化网络在重叠网络 (overlay)采用了随机图的方式组织, 节点度数服从“Power- law”[a][b]规律. 支持复杂查询, 如带有规则表达式的多关键词查询, 模糊查询等, 最典型的代表是 Gnutella

3)结构化拓扑网络中节点间的逻辑拓扑关系通常由确定性的算法控制, 资源放置由确定性的算法精确发布到特定的节点上. 其通常采用分布哈希表技术 (DHT)构建. 资源定位准确且保证一定的效率, 具有良好的可扩展性和性能. 结构化拓扑的 P2P系统都是全分布式的, 如 Chord和 CAN 等.

4)半分布式结构吸取中心化结构和全分布式非结构化拓扑的优点, 选择性能较高 (处理、存储、带宽等方面性能)的节点作为超级节点 (SuperNodes), 在各个超级节点上存储了系统中其他部分节点的信息, 发现算法仅在超级节点之间转发, 超级节点再将查询请求转发给适当的叶子节点. 半分布式结构也是一个层次式结构, 超级节点之间构成一个高速转发层, 超级节点和所负责的普通节点构成若干层次. 最典型的案例就是 KaZaa

1.2　P2P存储系统

P2P存储系统, 也即对等存储系统, 是指存储节点以一种功能对等的方式组成的一个存储网络. 相比于传统的存储系统有如下优点: 系统自然具有高扩展性; 各个节点功能对等, 即有高容错性. 高扩展性和高容错性使得利用廉价机群搭建大规模高性能存储服务成为可能. 由于没有采用集中式控制, P2P存储系统能够极大的减少存储系统的总拥有开销 (TCO, Total Cost of Ownership)[3]

虽然 P2P系统有与生俱来的高容错潜力, 但 P2P系统中每个节点都可能随时暂时或永久的离开系统, 这使得构建 P2P存储系统极富挑战. 系统中的节点均负责存储数据, 一旦某节点暂时离开, 存在其上的数据就将暂时不可访问, 而节点的永久离开更会造成数据的丢失. 因此, 如何对节点进行管理, 提供数据的持久存储, 屏蔽这些系统错误成为近年来 P2P存储领域的研究热点.

1.3　相关研究工作

早期 P2P存储网络的研究主要集中于路由算法和一致性维护等. “Total Recall”项目研究了副本冗余和纠删码冗余两种方式提高数据的可用性. 所有这些研究都建立在基础覆盖网络的基础之上, 因此, 合理的设计 P2P存储网络的拓扑结构是关键.

文献[4]提出了一种混合型 P2P网络结构, 它上层采用 Chord方法, 下层网络采用集中式方法. 充分发挥网络节点邻近性优势, 提高了用户之间的数据传输速度.

文献[5]通过聚集半分布式 P2P网络中物理位置相近的节点, 提出基于区域划分的超级节点选取机制, 该机制将网络中的节点按照物理位置的远近关系划分成若干区域, 保证区域内节点在物理位置上是相近的. 其降低了半分布式 P2P网络的信息检索延迟, 有效的提高了检索的效率, 并且具有较好的可扩展性.

文献[6]针对层次式 P2P系统中恶意超级节点频繁离开网络导致系统不稳定甚至崩溃的问题, 提出一种新的基于信誉的超级节点选择算法, 选择信誉高的节点为超级节点, 该算法有效地提高了系统的稳定性. Larrea在文献[7]中提出一种所谓“通信有效”的算法, 其设定仅仅只有超级节点发送信息.

Schiper和 Toueg[8]针对动态系统实现了三种健壮的、轻量级的超级节点选举服务. 其主要包括三个部分: 组管理模块、故障检测模块、节点选举算法模块. 三种服务各有侧重点, 针对不同的应用提供不同的服务质量.

2　多层节点管理模型

本文针对 P2P存储系统的特点, 提出了一种三层节点管理模型. 其主要包括中心管理节点 (CN)、超级节点 (SN)和存储资源提供节点 (RN), 如图 1所示:

CN: 系统的中心管理节点负责和超级节点之

间的联系,其记录了所有超级节点的相关信息,包括每个超级节点的标识符、信誉感知度评价、实际管理的资源提供节点的数目以及最大的节点管理数目. 为了防止单点失效的问题,系统中心管理节点的数目至少为 2个. 多个中心节点之间互为备份. 当某个中心管理节点失效,其它的中心节点能迅速接管其工作直到失效中心节点恢复为止.

SN: 超级节点负责管理系统中的存储资源提供节点 RN,其不但提供存储资源服务,而且可以用来检索、查找和定位资源,并记录每个 RN 节点的节点带宽、在线时间、提供的存储空间大小,并结合一定的权值设置给以评分,并由此评分值对其管理的所有 RN 节点进行排队,当 SN失效的时候,由中心节点重新选举一个综合评价值最高的 RN 节点升级为 SN.

RN: 系统自由加入的资源提供节点保存了自己注册的所有的 SN 的信息.

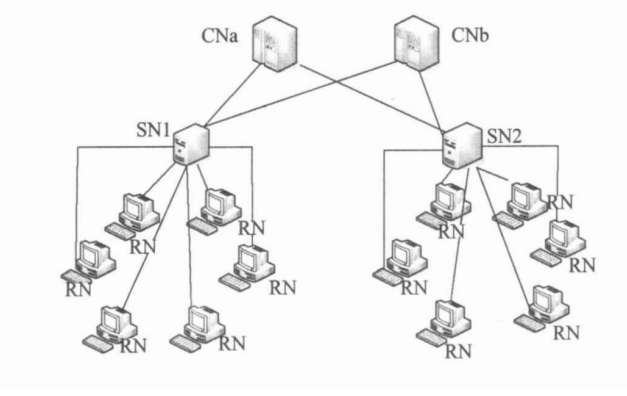

图 1 三层节点模型

Fig. 1 Three- layer node model

节点的加入: 先发送注册消息给 CN,再发送探测消息给各超级节点,根据消息的往返时间 RTT 的值来估计与超级节点之间的距离,选择距离最小的超级节点加入. 然后发送消息给该超级节点,一旦该超级节点所管理的节点数没有超过最大值则加入. 否则继续选择别的超级节点加入,直到找到为止. 如果一直没有找到合适的超级节点,则发送无法加入信息给中心管理节点. 并由中心管理节点选择一个性能最佳的资源提供节点 RN 升级为超级节点,并把刚提出申请的节点交给此超级节点管理.

节点的离开: 本文只考虑节点正常离开的情况,若离开节点是 RN,则该 RN 会向区域中有联系的 SN 发送离开消息,收到消息的 SN 需要更新自己的节点列表信息. 若离开节点是 SN,则除了要进行上述处理外,该 SN需要发送离开消息给所管辖的 RN,并且通知 CN 节点重新选举新的超级节点. 若离开的是 CN 节点,则其要发送消息给备份的中心管理节点,并把自己管理的超级节点信息发送给备份中心管理节点,由备份节点接管离开节点的工作.

3 故障检测机制

故障检测是保证节点高效管理的一个重要前提,它检测节点在线状态. 三层节点之间互相保持联系需要一个可靠的故障检测机制.

由于 P2P存储系统的高度动态性特征,本文设计了一种自调整(Self- Regulatory)的故障检测机制,它能适应实际系统由于网络状况的变化而存在的不稳定性,并且能够根据性能需求的变化重新配置故障检测的修正值,满足故障检测的 Qos评价指标要求.

由无偏灰色预测模型[9]产生的心跳到达时间预测值,结合心跳到达时间修正值 σ,生成 $N+1$ 次心跳消息到达时间预测值作为判断故障的时限(Timeout):

$$\mathrm{Timeout} = t^{(0)}(k) + \sigma,\ k = N+1$$

心跳到达时间修正值 σ:

$$\sigma = i \times \frac{\sum_{k=1}^{N} (t^{(0)}(k) - \hat{t}^{(0)}(k))}{N},$$

其中 i 为权值且 $0 < i < 2$, $t^{(0)}(k)$ 为每次原始心跳到达时间, $\hat{t}^{(0)}(k)$ 为每次预测的心跳到达时间, N 为时间窗口的大小.

另外,为了反映系统的实时性,达到指定大小的数据后,最新的心跳到达时间数据总是作为最后一个数据加入到时间窗中,并且删除第一个数据.

本故障检测方法具有较高的故障检测正确判断的结果,因此还可以用来对系统进行事先恢复: 即根据故障检测的精度值预先修复一些备份数据的副本.

4 实验结果及其分析

本文使用上述的故障检测机制对节点进行管理. 通过实验对本文设计的节点管理策略进行了验证,实验环境由 10台工作站组成,其位于武汉光电国家实验室. CPU 为 Intel Pentium dual CPU 1. 80 GHz,操作系统使用的是 linux2 4及以上内核,表 1显示了本方法对 CPU 负载率和网络带宽

的影响. 简单起见, 本实验假设消息的丢失率为 0, 网络延迟为 0.020 ms(其它的取值并不会影响分析), 所有检测消息通过 UDP协议传输, 检测周期设定为 200 ms.

表 1　节点管理对 CPU和网络带宽的损耗

Table 1　Overhead of CPU and network bandwidth to node management

节点数	平均 CPU占用率 Node/%	平均占用带宽 Node/(KB• s^{-1})
3	0.1	10
6	0.15	30
10	0.22	55

由于对节点的管理需要在节点之间频繁的进行消息的交互, 所以导致对 CPU和网络带宽有一定的损耗. CPU占用率是系统性能的主要参考指标, 如果在其他条件相同的情况下, CPU占用率越低, 表明系统的性能越好, 反之就越差. 测试数据表明, CPU占用率保持在 1%以下是合理的. 同样, 几十 kB的网络带宽占用也对当前的网络状况产生很小的影响.

5　结束语

P2P存储系统面临着数据不可访问和安全性这两个最大的挑战. 面对系统中节点的高度动态特征, 更需要设计一种更好的节点管理维护以及容错方法去适应企业的持续数据存储需求. 实际系统的运行面临许多问题, 比如说数据冗余和分发方案、数据维护策略, 敏感数据的私密性和完整性以及资源提供节点的激励机制, 这些都需要我们在下一步工作中完善.

参考文献:

[1] Dabek F, Kaashoek M, Karger D, et al. Wide-area cooperative storage with CFS[C]//18th ACM Symposium on Operating Systems Principles (SOSP'01). October, 2001.

[2] Rowstron A, Druschel P. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility[C]//Proceedings of the eighteenth ACM symposium on Operating systems principles, 2001: 188-201.

[3] Zhang Z, Lin S, Lian Q, et al. RepStore: a self-managing and self-tuning storage backend with smart bricks [C]//Proc of International Conference on Autonomic Computing 2004: 122-129.

[4] 杜吉辉. 混合型 P2P网络及应用[J]. 电信快报, 2008 (12): 25-28.

[5] 郭良敏, 杨寿保, 郭磊涛, 等. P2P网络中基于区域划分的超级节点选取机制[J]. 小型微型计算机系统, 2008, 29(2): 208-212.

[6] 刘玉枚, 杨寿保, 陈万明, 等. P2P系统中基于信誉感知的超级节点选择算法研究[J]. 中国科学院研究生院学报, 2008, 25(2): 197-203.

[7] Larrea M, Fernandez A, Arevalo S. Optimal implementation of the weakest failure detector for solving consensus (brief announcement)[C]//proceedings of PODC'00, ACM Press, 2000: 334.

[8] Bhagwan R. Total Recall: System Support for Automated Availability Management[C]//Proc. of the First ACM/Usenix Symposium on Networked Systems Design and Implementation(NSDI), 2004.

[9] Yaping Wan, Yang Luo, Li Liu, Dan Feng. A Dynamic Failure Detector for P2P Storage System[C]//proceedings of International Conference on New Trend in Information and Service Science, June, 2009.