基于 DHT 的 P2P 系统中高可用数据冗余机制

陈贵海[1)] 吴 帆[2)] 李宏兴[1)] 邱彤庆[3)]

[1)](南京大学软件新技术国家重点实验室 南京 210093)

[2)](纽约州立大学计算机科学与工程系 纽约布法罗 14260)

[3)](佐治亚理工大学计算机学院 佐治亚亚特兰大 30332)

摘 要 在基于 DHT 的 P2P 系统中需要采用冗余机制以保证数据的高可用性.文中结合用户下载行为来衡量数据存储与共享系统中的不同冗余机制.此外,作者提出了一种混合式的数据冗余策略,它兼具传统的复制策略和分片冗余策略的优点.实验表明,复制策略虽然比分片冗余策略需要更多的存储空间,但当节点平均可用性高于 47% 时,更节省网络维护带宽.混合式冗余策略在各种网络环境中均能较传统冗余策略更节省网络带宽,并且冗余因子适中.

关键词 分布式哈希表;对等计算;可用性;冗余;分片冗余;复制

中图法分类号 TP311

Redundancy Schemes for High Availability in DHTs

CHEN Gui-Hai[1)] WU Fan[2)] LI Hong-Xing[1)] QIU Tong-Qing[3)]

[1)](*State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing* 210093)

[2)](*Department of Computer Science and Engineering, SUNY at Buffalo, Buffalo, NY* 14260)

[3)](*College of Computing, Georgia Institute of Technology, Atlanta, GA* 30332)

Abstract High availability in peer-to-peer DHTs requires data redundancy. This paper takes user download behavior into account to evaluate redundancy schemes in data storage and share systems. Furthermore, it proposes a hybrid redundancy scheme of replication and erasure coding. Experimental results show that replication scheme saves more bandwidth than erasure coding scheme, although it requires more storage space, when average node availability is higher than 47%. The hybrid scheme saves more maintenance bandwidth with acceptable redundancy factor.

Keywords DHT; peer-to-peer; availability; redundancy; erasure-coding; replication

1 引 言

随着互联网和网络计算技术的迅猛发展,一类基于分布式哈希表(DHT)的对等计算系统应运而生[1-3].DHT 能支持全局性的数据存储,提供确定性的定位服务,并已有诸多应用[4-7].在不能保证节点 100%可用的情况下,要实现数据文件的高可用性,即数据被成功访问到的概率,需要某种数据冗余策略.目前应用于基于 DHT 的 P2P 系统的数据冗余策略主要有两种:复制(replication)[4,7]和分片冗余(erasure coding)[4-6].

在数据存储方面,和复制相比,分片冗余既具有优势,也有不足.其优势在于要达到相同的可用性水平,分片冗余比复制需要的存储空间和网络带宽少得多,甚至相差一个数量级(换言之,若使用相同的

收稿日期:2006-10-06;最终修改稿收到日期:2008-06-06.本课题得到国家"九七三"重点基础研究发展项目基金(2006CB303000)、国家自然科学基金(60573131,60673154,60721002)和江苏省高技术研究项目(BG2007039)资助.**陈贵海**,男,1963 年生,博士,教授,博士生导师,主要研究领域为并行计算和无线网络等.E-mail:gchen@nju.edu.cn.**吴 帆**,男,1981 年生,博士研究生,主要研究方向为无线网络、数据挖掘和对等计算.**李宏兴**,男,1982 年生,硕士研究生,主要研究方向为对等网络、擦除编码和网络编码等.**邱彤庆**,男,1981 年生,博士研究生,主要研究方向为对等计算和分布式系统.

存储空间,或者通过网络传输的数据量相等的情况下,分片冗余比复制达到的可用性水平高得多)[5,8-9].但是分片冗余的优势有限,只有当节点平均可用性较低时分片冗余才明显优于复制[10-11].而当节点平均可用性较高时,引入分片冗余往往不足以弥补引入它的代价,如额外的系统复杂度、异构环境中下载的延迟和不支持关键词检索等.

一方面,观察目前的 P2P 文件共享系统发现,热点文件的下载次数相当多,无需维护,其可用性自动保持在较高水平.这里,我们将用户下载使用的带宽与维持文件可用性水平所需的带宽区别开来,我们称前者为用户带宽,而后者为维护带宽.这表明了使用混合式的数据冗余策略(即结合用户主动下载行为(复制)和分片冗余)来维持文件可用性水平的可行性.换言之,我们利用用户下载行为来节省维护带宽,再通过分片冗余使用较少量维护带宽使文件可用性维持在指定水平.另一方面,从计算机硬件的发展趋势看,存储器容量的发展速度远远高于网络接口带宽,相对于存储资源来说,带宽资源变得更加紧俏[11].在过去的 15 年中,存储器容量增长了近 8000 倍,而网络接口带宽却仅增长了 50 倍[10].这表明了研究更加有效的节省带宽的数据冗余策略的必要性.

本文提出了一种混合式的数据冗余策略,它既像复制策略那样共享用户下载的文件以供后续的下载,又利用分片冗余维持文件的可用性.实验表明,这种混合式的数据冗余策略较传统冗余策略(单一地使用复制或分片冗余)更节省网络带宽,并且冗余因子适中.

本文的主要贡献如下:(1)本文率先考虑用户下载行为因素(即用户下载频率)来衡量数据存储与共享系统的不同冗余机制;(2)本文指出复制策略虽然比分片冗余策略需要更多的存储空间,但当节点平均可用性高于 47%时,更节省网络维护带宽;(3)本文提出的混合式的数据冗余策略在各种网络环境中均较传统冗余策略更节省网络带宽,并且冗余因子适中(达到 0.999 的可用性所需冗余因子不超过 9.4).

本文第 2 节介绍有关复制和分片冗余的背景知识;第 3 节介绍相关工作;第 4 节详细描述 3 种数据冗余策略:复制策略、分片冗余策略和混合式策略;在第 5 节通过实验来评估并分析 3 种数据冗余策略;最后第 6 节总结全文并指出今后的研究方向.

2 背 景

在数据存储和共享系统中有两种传统的实现数据高可用性的机制:复制(如 RAID 1)和奇偶校验冗余机制(parity scheme)(如 RAID 2～5).但是由于前者会引入极高的冗余因子,而后者适应环境动态性的能力又较差,所以它们在高动态性的网络环境中难以保证数据的可用性.

分片冗余的出现克服了复制机制冗余因子高的缺点.在实际运行中它首先将数据对象分割成 m 片数据分片,再将这 m 个数据分片编码成 n 片编码分片($n>m$).在此,我们将 $r=n/m$ 定义为编码冗余因子.以冗余因子 r 编码即意味着编码后所有编码分片占用的存储空间是原始数据的 r 倍.分片冗余最重要的特性是只要获得任意 m 个不同的编码分片就能重构原文件,而且这 m 个分片的体积之和与原文件大致相等.例如,若编码冗余因子为 $r=4$,原数据分割为 $m=16$ 片,则编码后为 $n=64$ 片,所需存储空间增大到原先的 4 倍.复制和 RAID 可以看作是分片冗余的特例.冗余因子为 4 的复制系统相当于($m=1$, $n=4$)的分片冗余;1、4 和 5 级的 RAID 用分片冗余分别表示为($m=1$, $n=2$)、($m=4$, $n=5$)和($m=4$, $n=5$).

3 相关工作

为克服 P2P 网络环境的异构性和节点的不可靠性带来的不利影响,几乎所有的 P2P 系统都提供了相应的确保数据可用性的机制.总体上来讲,可以采用的机制主要是复制技术和分片冗余技术两种.

使用复制技术维持文件可用性的系统较多.CFS[4]、PAST[7]和微软的 FARSITE[12]采用静态的复制系数,并用主动方式维护文件可用性.Ranganathan 等人[13]提出的方法中,每个节点根据自己建立的 P2P 存储系统的模型计算所需副本的数量和放置的位置.Bhagwan 等所给出的 Total Recall[8]则根据系统中节点的历史行为,推算出当前阶段合适的冗余策略.On 等人[14]提出合理安排副本放置的位置比盲目增加副本数量更能有效提高文件的可用性.

Oceanstore[6]同时使用了复制和分片冗余,对于经常读写的文件采用复制策略,对于不常用但需要长期保存的文件采用分片冗余.Cuenca-Acuna 等人[15]的工作与我们的可以说是最接近的,他们使用

复制和分片冗余相结合的方法维持文件可用性,并用一个全局目录管理副本和分片位置信息,但全局目录会导致单点失效问题.

关于复制技术和分片冗余技术的性能比较的研究由来已久.以往的众多研究成果[5,9]主张分片冗余优于复制,因为达到相同的可用性水平,分片冗余比复制需要的存储空间和网络带宽少得多,甚至相差一个数量级(换言之,若使用相同的存储空间,分片冗余比复制达到的可用性水平高得多).另一些文章[10-11]则认为只有当节点平均可用性较低时分片冗余才具有优势,并且分片冗余的优势有限,以至于往往不足以弥补引入它的代价,如额外的系统复杂度、异构环境中下载的延迟和不支持关键词检索等.

此外,除了具体的冗余算法,不同的副本维护策略也会给应用系统的性能带来不同的影响.目前,主要存在的是两种副本维护策略,一种是被动复制策略,另一种是主动复制策略.

对于被动复制策略[5,8,16-18],系统需要监控网络中相关文件的冗余因子,当该冗余因子低于某一个阈值时系统对相关文件进行复制以提高冗余因子.不过,这种策略也带来了一些问题,比如:需要对相关文件进行监控,提升了系统开销;系统的修复行为会给网络带来突发性的带宽消耗,有可能会影响到系统的正常应用.

最新的研究[19]所提出的主动复制策略则克服了被动复制策略所带来的弊端,同时只消耗了很少的网络资源.主动复制策略的主要思想是利用闲置的网络带宽资源,周期性地以一个确定的速率复制多余的副本以提高数据可用性.其实验结果表明该策略只消耗了相对较少的带宽资源而获得了较高的数据可用性和可靠性.

由于本文的主要工作和贡献集中于冗余算法和机制方面,所以将不再对被动和主动复制策略做更多描述.

4　冗余策略

这一节我们将首先详细阐述两种传统的保证数据高可用性的冗余策略:复制策略和分片冗余策略.在此之后我们提出了一套全新的混合式冗余策略.混合式冗余策略共享用户下载的文件作为后续的下载源,并利用分片冗余维持文件的可用性.这 3 种策略均使用一致性哈希函数(consistent hashing)[20],应用于数据/文件存储与共享系统,如 CFS[4].

简单起见,赋予每个文件一个唯一的标识符 d,即其文件名的一致性哈希值.保存文件 d 的完整副本或分片所在位置的节点称为 d 的索引节点.这里引入一个二元函数 $h(d,n)$,n 为索引编号($n\geq 1$).$h(\cdot,\cdot)$ 可定义为如下形式:$h(d,n)=H(d\,\|\,n)$,$H(\cdot)$ 为 DHT 中使用的哈希函数,$\|$ 为连接操作.图 1 是多重索引的一个示例.

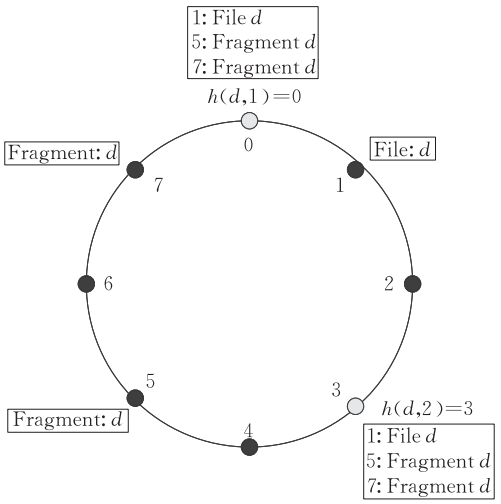


图 1　多重索引示例(假设 $M=2$,节点 1 存储了完整的文件 d,节点 5 和节点 7 分别存储了文件 d 的不同分片.由 $h(d,n)$确定的 2 个保存文件 d 索引的节点为 0 和 3)

各种冗余策略虽各具特点,但都由以下 3 部分组成:注册模块、请求模块和维护模块.

(1)注册模块.每个节点周期性地将其储存的文件和分片的标识符注册到 M 个分散且相互独立的索引节点.M 个索引节点的逻辑位置由前面定义的二元函数 $h(d,n)$,$n\in[1,M]$决定.如果 $h(d,n)$所指的节点不在线,则将其后继作为索引节点.索引节点赋给每个索引项一个计时器,当计时器超时,该项将被认为不可用并从索引中删除.

(2)请求模块.当某节点请求文件 d 时,它首先随机查询一个或多个 d 的索引节点.如果查询过的索引节点未能提供足够的文件或分片位置信息,则继续随机查询其他索引节点.如果所有索引节点都不能提供足够的文件或分片位置信息,该节点将等待一段时间后重做查询操作,直到查询超时.这种方法,可以平衡各索引节点的负载,并且减少得到不完整的索引的可能性.最后,若查询成功,节点根据得到的索引到注册的节点上下载文件或分片并重构原文件.

(3)维护模块.各索引节点周期性地评估在其

上注册的文件和分片的可用性，若可用性低于要求的水平，则启动程序修复文件和分片的可用性.

4.1 复制策略

复制是最简单的冗余策略. r 个副本被分别存放在相互独立的节点上. r 的值应该根据文件可用性 a（a 可表示为 9 的个数，例如 0.999）和节点的平均可用性 p 来确定. 文件可用性即是文件能够被成功访问到的概率，而节点可用性代表着节点正常开机的概率. 本文假设各节点是相互独立的. 所需的副本个数可由下式决定：

$$a=1-(1-p)^r \tag{1}$$

解 r 可得

$$r=\frac{\log(1-a)}{\log(1-p)} \tag{2}$$

复制策略的注册和查询算法与注册模块和请求模块中的描述一致. 若查询成功，该节点根据得到的索引到注册的节点上下载文件. 下载的文件被自动视为共享文件，以供随后的下载. 各索引节点周期性地维护在其上注册的文件的可用性，若低于要求水平，则安排必要数量的文件复制操作，副本被分别发送到随机选择的节点上.

4.2 分片冗余策略

分片冗余（例如 Reed-Solomon[21] 和 Tornado[22]）将数据对象分割成 m 片，再编码成 n 片（$n>m$）. 就是说分片冗余的冗余因子为 $r=n/m$. 分片冗余最重要的特性是只要获得任意 m 个不同的分片就能重构原文件，而且这 m 个分片的体积之和与原文件大致相等.

本文假设一个节点只存放同一文件的一个分片，但可以存放不同文件的分片，并且系统中不存在重复的分片. 文件的可用性可根据概率计算出来，至少获得 n 个分片中的任意 m 个的概率为

$$a=\sum_{i=m}^{n}\binom{n}{i}p^i(1-p)^{n-i} \tag{3}$$

这里，p 是节点平均可用性.

各节点上的文件数量大致符合泊松分布. 由于计算泊松分布比较困难，这里用正态分布近似泊松分布. 如果随机地将文件放置在各节点上，则各节点上的文件数量将大致符合正态分布. 通过代数化简和正态逼近，可得到下面的分片冗余的冗余因子表达式，具体推导过程可参见文献[23].

$$r_c=\frac{n}{m}=\left(\frac{\sigma_a\sqrt{\frac{p(1-p)}{m}}+\sqrt{\frac{\sigma_a^2p(1-p)}{m}+4p}}{2p}\right)^2 \tag{4}$$

这里，σ_a 是所需的可用性对应的正态分布中的标准差. 表 1 列出了不同的文件可用性对应的正态分布标准差. 例如 $\sigma_a=3.1$ 对应 3 个 9 的可用性.

表 1 文件可用性和正态分布标准差对照表

a	σ_a
0.800	0.84
0.900	1.28
0.990	2.48
0.995	2.81
0.998	2.88
0.999	3.10

从式(4)看出冗余因子与 n 无关，换句话说，n 与维持文件可用性水平所需的存储空间的大小无关. p 是节点平均可用性，是 P2P 网络的属性. 唯一决定冗余因子的因素就是 m. 当 m 增大，冗余因子降低，但系统的复杂度增加且下载延迟增大，在异构的环境中，下载延迟的增大尤为明显.

在使用分片冗余策略时，产生新的分片需要首先获得完整的原文件. 如果每次生成新分片时都要下载必要的分片重构文件，将消耗巨大的网络带宽. 生成 1 个新分片却需要下载 m 个分片，这样维持可用性消耗的网络带宽将是丢失冗余数据体积的 m 倍. 一个有效的办法是给每个文件指定一个主节点（home peer），主节点定义为节点 ID 最接近于该文件标识符的节点. 主节点负责保存完整的文件并生成新分片. 如果某一主节点失效，则离它最近的节点接替它的任务成为主节点. 如图 2 所示. 这相当于给冗余因子加 1.

有两种新分片再生策略[15]：

(1) 最常用的方法是生成所有 n 个分片，一段时间以后，检测并重新生成丢失的分片. 但这种方法在高度动态的环境中存在两点不足：① 必须准确记录各节点上保存分片的情况，以确定当某节点失效时或分片被替换出缓存时应重新生成哪个分片. ② 必须能够区别节点暂时下线和永久离开，以免引入重复的分片. 这两个点降低了分片冗余策略的效率.

(2) 设置 $n\gg m$，但不生成所有的分片. 当提高某文件的可用性时，相应的主节点只需随机产生一个或多个分片即可. 当 n 足够大时，随机产生的分片与系统中已存在的分片重复的概率足够小，这样即使没有节点之间的协调，也能保证新生成分片具有较高的有效性. 当 $n-m\geq m$ 时，编码和解码一个随机分片的复杂度为 $\theta(m)$. 所以，大 n 并不会提高编

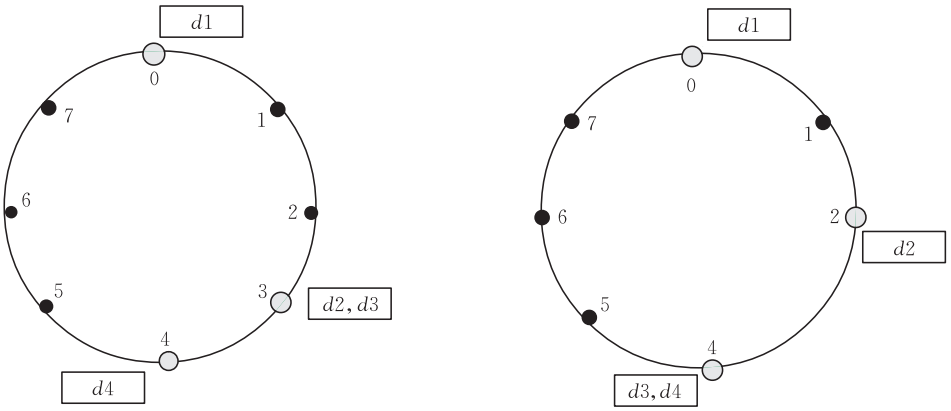


图 2　主节点示例(在图(a)中,节点 0 是文件 $d1$ 的主节点,节点 3 是文件 $d2$ 和 $d3$ 的主节点,节点 4 是文件 $d4$ 的主节点. 节点 3 失效后的情况如图(b)所示,$d2$ 离节点 2 的 ID 较近,则节点 2 成为 $d2$ 的主节点;$d3$ 离节点 4 的 ID 较近,则节点 4 成为 $d3$ 的主节点)

码和解码的复杂度. 式(3)和式(4)用于这种方法时,式中的 n 应重定义为当前系统中的分片数目.

分片冗余策略并不要求共享用户下载的文件,而是只共享缓存在各节点上的文件分片. 每个节点周期性的将其缓存的分片的标识符注册到 M 个分散且相互独立的索引节点. 当某节点请求文件 d 时,它首先随机查询一个或多个 d 的索引节点. 如果查询过的索引节点未能提供足够的分片位置信息,则继续随机查询其他索引节点. 如果所有索引节点都不能提供足够的分片位置信息,该节点将等待一段时间后重做查询操作,直到查询超时. 若查询成功,节点根据得到的索引到注册的节点上下载分片并重构原文件,然后随机生成一个分片放进自己的缓存中. 各索引节点周期性地维护在其上注册的分片的可用性,若低于要求水平,则通知相应的主节点生成必要数量的新分片,新分片将被分别发送到随机选择的节点上.

4.3　混合式策略

复制策略通过共享用户下载的文件以供后续的下载来节省网络带宽. 虽然它的冗余因子明显高于分片冗余策略,但当节点平均可用性较高时,即动态性较低的网络环境中,它比分片冗余策略更节省网络带宽. 另一方面,分片冗余策略和复制策略相比,达到相同可用性水平所需的存储空间较少;在高度动态性的网络环境中,分片冗余策略更节省网络带宽,但维护开销始终过大. 我们提出了一种混合式的数据冗余策略,它兼具复制策略和分片冗余策略的优点,在各种网络环境中都有较好的性能,更加节省网络带宽,冗余因子也适中.

混合式策略既像复制策略那样共享用户下载的文件以供后续的下载,又利用分片冗余维持文件的可用性. 混合式策略自动将下载的文件视为共享文件,以供随后的下载;当修复某文件可用性水平时,指定任何一个储存该文件完整副本的节点生成并分发新分片即可. 一方面,混合式策略像复制策略那样共享用户下载的文件以供后续的下载,节省了网络带宽;另一方面,增加相同的可用性,分片冗余策略产生的新数据的体积比复制策略少,即分片冗余策略需要传送的数据较少,混合式策略利用这一特性进一步节省了网络带宽.

使用混合式策略时的文件可用性和冗余因子可由如下方法推出. 首先,我们假设同一节点不存放同一文件的副本和分片,可以存放多个不同文件的副本和分片,并且系统中不存在重复的分片. 文件的可用性 a 可根据概率计算出来,即至少获得一个完整的副本或 n 个分片中的任意 m 个的概率. 或者通过 1 减去所有副本都无法得到且没有足够的分片重构原文件(可用分片数不足 m 个)的概率:

$$a=1-(1-p)^h\left(1-\sum_{i=m}^{n}\binom{n}{i}p^i(1-p)^{n-i}\right) \quad (5)$$

这里,h 为完整副本的个数.

混合式策略的冗余因子可由叠加复制和分片冗余的冗余因子得到

$$r=h+\frac{n}{m}$$

$$=h+\left(\frac{\sigma_a(h)\sqrt{\frac{p(1-p)}{m}}+\sqrt{\frac{\sigma_a^2(h)p(1-p)}{m}+4p}}{2p}\right)^2 \quad (6)$$

这里，$\sigma_a(h)$是关于 h 的函数，其值和分片冗余需要达到的可用性水平 a'的对应关系如表 1 所示. a'可由式(5)推出：

$$a'=\sum_{i=m}^{n}\binom{n}{i}p^i(1-p)^{n-i}=1-\frac{1-a}{(1-p)^h} \quad (7)$$

从式(6)看出冗余因子 r 与 m 和完整副本的个数 h 有关，而与 n 无关. r 随 h 的增大而增大，随 m 的增大而减小. 给定 h，当 m 增大时，冗余因子降低，但系统的复杂度增加、下载延迟增大；反之，当 m 减小时，系统的复杂度降低、下载延迟减小，但冗余因子增大、消耗存储空间增加. 分片冗余策略的两个新分片再生策略依然适用于混合式策略，并且第二条策略在混合式策略中优越性更加显著.

图 3 展示了复制策略、分片冗余策略和混合式策略为达到文件可用性为 0.999 所需冗余因子理论值随节点平均可用性变化的情况，其曲线分别由式(2)、式(4)和式(6)决定. 混合式策略的冗余因子曲线有两个因素决定：节点平均可用性 p 和完整副本的个数 h. 对任意给定 h 值，都有一条与之相对应的曲线. 由图可见，当节点平均可用性一定时，分片冗余策略的冗余因子最小；除了当节点平均可用性极高时，混合式策略的冗余因子略高于分片冗余策略，但大大低于复制策略.

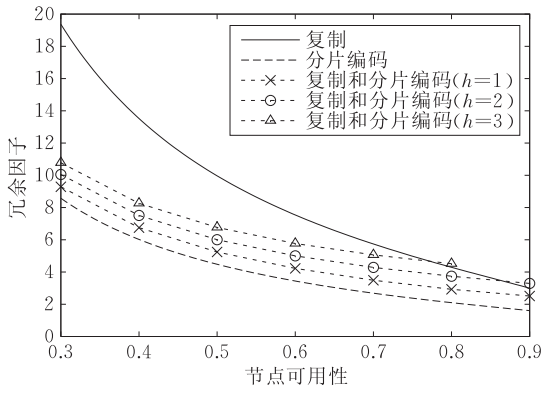


图 3 文件可用性为 0.999 时，3 种冗余策略所需冗余因子理论值随节点平均可用性变化的情况(复制策略、分片冗余策略和混合式策略的曲线分别由式(2)、(4)和式(6)决定($m=7$))

每个节点周期性地将其储存的文件和分片的标识符注册到 M 个分散且相互独立的索引节点. 节点通过两种索引查找文件：文件副本索引和文件分片索引. 当某节点请求文件 d 时，它首先随机查询一个或多个 d 的索引节点. 如果查询过的索引节点未能提供足够的文件副本和分片位置信息，则继续随机查询其他索引节点. 如果所有索引节点都不能提供足够的文件副本和分片位置信息，该节点将等待一段时间后重做查询操作，直到查询超时. 若找到完整文件副本，则下载该副本；否则，节点转而下载足够的分片并重构原文件. 用户下载的文件自动作为共享文件以供后续的下载.

各索引节点周期性地维护在其上注册的文件和分片的可用性. 若文件 d 的可用性低于要求水平，则指定某一个储存该文件完整副本的节点生成并分发新分片到随机选择的节点. 若当前系统中没有完整的副本，首先等待一段时间，若有节点下载分片并重构原文件，则指定该节点修复丢失的冗余因子；否则，索引节点需要自己下载分片，重构原文件，再生成足够的新分片. 实验表明几乎所有文件都至少有一个副本存在于系统中，所以混合式策略没有必要像分片冗余策略那样指定主节点保存文件的永久副本. 这样，混合式策略节省了维护主节点所需的网络带宽.

5 实验评估

我们在 P2Psim① 上实现了上述 3 种数据冗余策略. 模拟的网络环境包含 1024 个节点. 各节点交替离开和重新加入网络，相邻事件的时间间隔呈指数分布. 当节点失效或离开，其上的所有文件、分片和索引均被丢弃. 每次有节点加入网络，都会使用不同的 IP 和 DHT 标识符. 对文件的访问请求按照 Zipf-like 分布，将文件按访问频率从高到低排列，第 i 个文件的访问频率和 $1/i^{\alpha}$ 成正比，这里设置 α 为 0.74(即文献[24]中 6 个不同网络的 α 平均值). 我们设计了两套实验：变化节点平均可用性和变化用户请求频率. 在变化节点平均可用性实验中，节点平均可用性变化范围为 30%～90%，其它条件不变. 在变化用户请求频率实验中，节点生命期内平均文件请求数变化范围为 2～20，其它条件不变. 每组实验运行模拟器时间 6h，只收集后半段模拟时间的数据. 每组实验重复 5 次，结果取平均值. 设置 $m=7$，这是 CFS[15] 中使用的重构原文件所需分片数. 设置文件可用性为 0.999，这是目前大多数网络服务所能达到的可用性[25].

① Gil T, Kaashoek F, Li J, Morris R, Stribling J. P2Psim: A simulator for peer-to-peer protocols. http://www.pdos.lcs.mit.edu/p2psim/

我们主要采用以下两种冗余策略评价指标:

(1)冗余因子(redundancy factor). 冗余因子定义为为达到文件可用性所需要的存储空间与储存原文件一个副本所需存储空间之比.

(2)维护带宽占用率(bandwidth ratio). 维护带宽占用率定义为维持文件可用性和分片冗余策略中维护主节点所消耗的网络带宽,与用户下载服务所用带宽之比. 相比于上述带宽消耗来说,维护路由表和查询文件占用的带宽可以忽略不计. 维护带宽占用率为 0.1 代表维护用带宽占系统日常服务使用带宽的 10%.

因为目前带宽资源较存储资源相对不足,故本文把维护带宽占用率作为更加重要的评价指标.

5.1　冗余因子

图 4 展示了复制策略、分片冗余策略和混合式策略 3 种冗余策略在达到文件可用性 0.999 的情况下,冗余因子随节点平均可用性变化的曲线. 图 4 中分片冗余策略的冗余因子曲线与图 3 中的大体一致. 但图 4 中复制策略的冗余因子曲线却与图 3 中的相差甚远,尤其是当节点平均可用性大于 0.6 时. 这是因为分片冗余策略不共享用户下载的文件,仅在缓存中保留一个分片,受用户下载行为的影响小;但复制策略完全共享用户下载的文件,网络中文件副本的数量随用户的下载而增多. 同时,节点平均可用性越高,系统中文件副本的丢失率就越低. 过多的副本残留在系统中,造成了复制策略的冗余因子没有随节点平均可用性提高而明显下降.

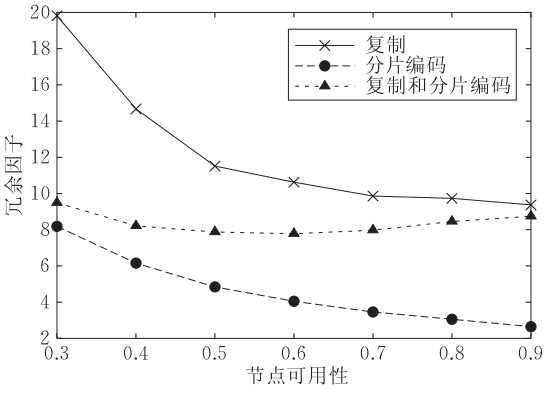


图 4　文件可用性为 0.999 时,复制策略、分片冗余策略和混合式策略的冗余因子随节点平均可用性变化的情况

由图 4 可见,虽然节点平均可用性由 0.3 变化到 0.9,但混合式策略的冗余因子变化并不明显,徘徊在 8.5～9.4 之间. 当节点的扰动强烈时,即节点平均可用性低时,系统中的文件副本较少,混合式策略主要利用分片冗余来节省存储空间. 当节点平均可用性高时,混合式策略的冗余因子不降且升,其原因与复制策略相似:过多的副本残留在系统中. 这些多余的冗余文件或分片不会对系统的稳定性造成危害,随着时间的推移,多余的文件和分片会被节点逐渐置换出缓存.

5.2　维护带宽占用率

由图 5 可见,当节点平均可用性高于 0.47 时,复制策略比分片冗余策略更节省维护带宽;当节点平均可用性低于 0.47 时,分片冗余策略消耗的维护带宽较复制策略少. 复制策略通过共享用户下载的文件来减轻维护操作的负担. 当节点平均可用性较高时,复制策略在节约维护带宽方面非常有效,原因在于,依赖于用户的下载行为大部分文件的可用性被自动维持在目标水平. 但在高度动态的网络环境中,节点频繁地加入和离开,使用户下载的速度无法弥补文件副本和分片的丢失. 在这种情况下,分片冗余策略的优势就显现出来:消耗同样的网络带宽,分片冗余策略比复制策略更能提高文件可用性水平;换言之,提高同样的文件可用性水平,分片冗余策略比复制策略需要更少的带宽.

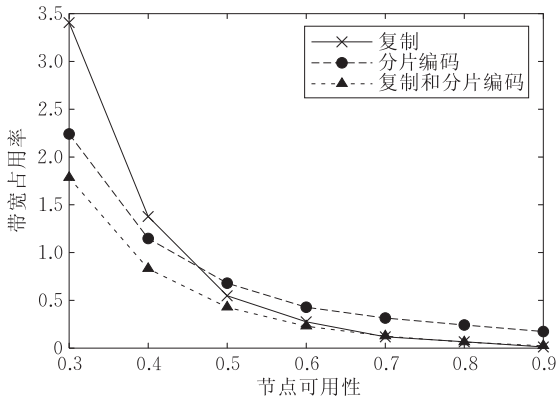
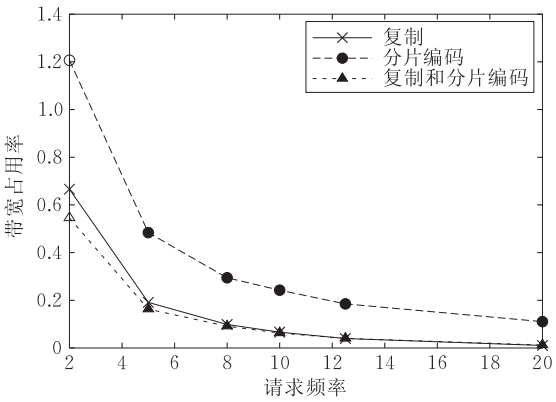


图 5　文件可用性为 0.999 时,复制策略、分片冗余策略和混合式策略的维护带宽占用率随节点平均可用性变化的情况

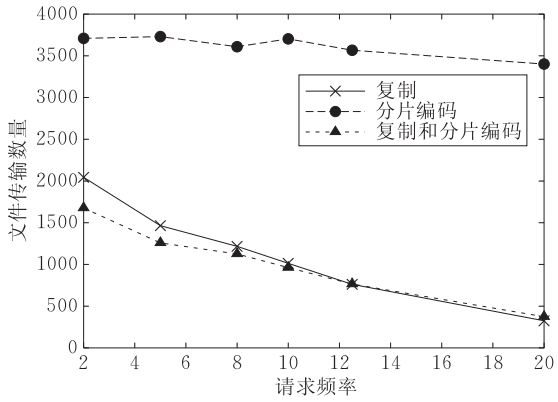
图 5 表明,在节省维护带宽方面混合式策略具有最好的整体效果. 混合式策略既像复制策略那样共享用户下载的文件以供后续的下载,又利用分片冗余维持文件的可用性. 当节点平均可用性高于 0.7 时,由于绝大部分文件的可用性被自动维持在目标水平,故复制策略和混合式策略消耗的维护带宽大致相等. 当节点平均可用性低于 0.7 时,混合式策略的优越性就显现出来了. 混合式策略优于分片冗余策略的原因在于:(1)共享用户下载的文件以

节省维护带宽;(2)无需维护主节点.

图 6 所示的节点平均可用性为 0.807,这是在一般的公司和高校中我们能够观察到的网络状况[26].由图 6(a)可见,文件请求率越高,维护带宽占用率越低.维护带宽占用率是一个关于维护带宽和文件下载服务带宽的相对指标,图 6(b)展示了维护带宽的绝对值①.由图 6(b)可见,随着平均文件请求率的提高,复制策略和混合式策略用于维持文件可用性传输的文件数目明显减少,但分片冗余策略的曲线却仅在小范围内波动.这一现象证明了共享用户下载的文件以供后续的下载能够有效降低维护带宽开销.



(a)维护带宽占用率



(b)维护中文件传输数量

图 6 文件可用性为 0.999,节点平均可用性为 0.807 时,复制策略、分片冗余策略和混合式策略的维护带宽占用率和维护中文件传输数量随节点生命期内平均文件请求率变化的情况

图 6(b)还表明,当文件请求率较低时,混合式策略在节省维护带宽方面优于其他两种冗余策略.当文件请求率大于 10 时,复制策略和混合式策略维护带宽占用率非常接近,并且贴近水平坐标轴,这是因为依赖于用户的大量下载行为使大部分文件的可用性自动维持在目标水平.

6 结论与展望

本文考虑用户下载请求频率来衡量数据存储与共享系统中的不同冗余机制.实验结果表明:复制策略虽然比分片冗余策略占用更大的存储空间,但当节点平均可用性高于 0.47 时,达到同样的文件可用性,复制策略更节省网络带宽.当节点平均可用性高于 0.7 时,复制策略和混合式策略的维护带宽占用率非常接近.此外,复制策略是 3 种冗余策略中最简单的,维护的复杂度最小.所以,在节点平均可用性较高的网络环境中,如在企业或高校的网络环境,复制策略是一个非常好的选择.

分片冗余策略和复制策略相比,达到相同可用性水平所需的存储空间较少;而且,在高度动态性的网络环境中,分片冗余策略消耗的网络带宽较少.分片冗余策略的不足之处在于:当节点平均可用性高于 0.47 时,维护带宽占用率较复制策略高;给系统引入的复杂度不仅包括编码和解码分片的复杂度,还包括系统设计的复杂度.

本文提出的混合式策略既像复制策略那样共享用户下载的文件以供后续的下载使用,又利用分片冗余维持文件的可用性.这种混合式的数据冗余策略在各种网络环境中均较传统冗余策略更节省网络带宽,并且冗余因子适中(不高于 9.4).实验结果表明:当节点平均可用性低于 0.7 和文件请求率相对于节点扰动频率较低时,混合式策略的优越性尤为突出.混合式策略不仅在动态性低的网络环境中表现良好,而且在高度动态的网络中性能卓越.混合式策略的缺点是给系统增加了复杂度.

使用混合式策略在节省网络带宽方面虽然较传统的方法能够取得更佳的效果,但在节点平均可用性低于 0.5 的高度动态网络环境中,它的维护带宽占用率仍然很高,系统的可扩展性差.考虑到网络带宽相对存储空间较缺乏,近期的改进重点将放在如何进一步节省维护带宽上.设计新型的编码方法和进一步利用用户下载的文件是将来的两个研究方向.

此外,在本文中为了便于将更多的精力放在冗余算法和机制的研究上,我们并没有把网络中各个

① 分片冗余策略和混合式策略的维护带宽绝对值是按传输的分片数统计的,为方便 3 种冗余策略的比较,将分片数折算为文件数.

结点的存储限制列入考虑范围.而在实际应用中由于对等网络中结点的异构性,不同结点的存储能力也存在着差异.同时我们还需要考虑到存储文件的更新和替换的问题.在一些国际最新的研究成果[26]中,由于应用特性、复制策略和结点可靠性差异所带来的存储差异及限制更加引起了我们的关注和重视.我们将会在进一步的工作中对以上问题进行研究并期望得到解决.

参 考 文 献

[1] Ratnasamy S, Francis P, Handley M, Karp R, Shenker S. A scalable content addressable network//Proceedings of the ACM SIGCOMM. San Diego, USA, 2001: 161-172

[2] Rowstron A, Druschel P. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems//Proceedings of the Middleware. Heidelberg, Germany, 2001: 1611-3349

[3] Stoica I, Morris R, Karger D, Kaashoek M F, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for internet applications//Proceedings of the ACM SIGCOMM. San Diego, USA, 2001: 149-160

[4] Dabek F, Kaashoek M F, Karger D, Morris R, Stoica I. Wide-area cooperative storage with CFS//Proceedings of the ACM SOSP. Banff, Canda, 2001: 202-215

[5] Dabek F, Li J, Sit E, Robertson J, Kaashoek F, Morris R. Designing a DHT for low latency and high throughput//Proceedings of NSDI. San Francisco, USA, 2004: 85-98

[6] Kubiatowicz J, Bindel D, Chen Y, Czerwinski S, Eaton P, Geels D, Gummadi R, Rhea S, Weatherspoon H, Weimer W, Wells C, Zhao B. Oceanstore: An architecture for globalscale persistent storage//Proceedings of the ASPLOS. Cambridge, MA, USA, 2000: 190-201

[7] Rowstron A, Druschel P. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility//Proceedings of the SOSP. Banff, Canada, 2001: 188-201

[8] Bhagwan R, Tati K, Cheng Y, Savage S, Voelker G. Total Recall: System support for automated availability management//Proceedings of the NSDI. San Francisco, USA, 2004: 337-350

[9] Weatherspoon H, Kubiatowicz J. Erasure coding vs. replication: A quantitative comparison//Proceedings of the IPTPS. Cambridge, MA, USA, 2002: 328-338

[10] Blake C, Rodrigues R. High availability, scalable storage, dynamic peer networks: Pick two//Proceedings of the HotOS. Hawaii, USA, 2003: 1-6

[11] Rodrigues R, Liskov B. High availability in DHTs: Erasure coding vs. replication//Proceedings of the IPTPS. Ithaca, USA, 2005: 226-239

[12] Bolosky W J, Douceur J R, Ely D, Theimer M. Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs//Proceedings of the SIGMETRICS. Santa Clara, USA, 2000: 34-43

[13] Ranganathan K, Iamnitchi A, Foster I. Improving data availability through dynamic model-driven replication in large peer-to-peer communities//Proceedings of the CCGRID. Berlin, Germany, 2002: 376-381

[14] On G, Schmitt J, Steinmetz R. The effectiveness of realistic replication strategies on quality of availability for peer-to-peer systems//Proceedings of the P2P. Linköping, Sweden, 2003: 57-65

[15] Cuenca-Acuna F M, Martin R P, Nguyen T D. Autonomous replication for high availability in unstructured P2P systems//Proceedings of the SRDS. Florence, Italy, 2003: 99-108

[16] Chun B, Dabek F, Haeberlen A, Sit E, Weatherspoon H, Kaashoek M F, Kubiatowicz J, Morris R. Efficient replica maintenance for distributed storage systems//Proceedings of the NSDI. San Jose, Canada, 2006: 45-58

[17] Haeberlen A, Mislove A, Druschel P. Glacier: Highly durable, decentralized storage despite massive correlated failures//Proceedings of NSDI. Boston, USA, 2005: 143-158

[18] Rhea S, Godfery B, Karp B, Kubiatowicz J, Ratnasamy S, Shenker S, Stoica I, Yu H. OpenDHT: A public DHT service and its users//Proceedings of the ACM SIGCOMM. Philadelphia, USA, 2005: 73-84

[19] Sit E, Haeberlen A, Dabek F, Chun B, Weatherspoon H, Morris R, Kaashoek M F, Kuiatowicz J. Proactive replication for data durability//Proceedings of the IPTPS. Santa Barbara, USA, 2006

[20] Karger D, Lehman E, Leighton F, Levine M, Lewin D, Panigrahy R. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide Web//Proceedings of STC. Salt Lake City, USA, 1997: 654-663

[21] Reed S, Solomon G. Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics, 1960, 8(2): 300-304

[22] Byers J W, Luby M, Mitzenmacher M, Rege A. A digital fountain approach to reliable distribution of bulk data//Proceedings of the ACM SIGCOMM. Vancouver, Canada, 1998: 56-67

[23] Bhagwan R, Savage S, Voelker G. Replication strategies for highly available peer-to-peer storage systems. San Diego, USA, UCSD: Technical Report CS2002-0726, 2002

[24] Breslau L, Cao P, Fan L, Phillips G, Schenker S. Web-caching and Zipf-like distributions: Evidence and implications//Proceedings of the IEEE INFOCOM. New York, USA, 1999: 126-134

[25] Merzbacher M, Patterson D. Measuring end-user availability on the Web: Practical experience//Proceedings of the IPDS. Washington DC, USA, 2002: 473-477

[26] Tati K, Voelker G M. On object maintenance in peer-to-peer systems//Proceedings of the IPTPS. Santa Barbara, USA, 2006

CHEN Gui-Hai, born in 1963, Ph. D., professor and Ph. D. supervisor. His research interests include parallel computing and wireless networks.

WU Fan, born in 1981, Ph. D. candidate. His research interests focus on wireless networks, data mining, and peer-to-peer computing.

LI Hong-Xing, born in 1982, M. S. candidate. His research interests include peer-to-peer computing, wireless networks, erasure coding and network coding.

QIU Tong-Qing, born in 1981, Ph. D. candidate. His research interests are in the areas of peer-to-peer computing and distributed systems.

Background

This work is supported by many research funding agencies both from China and from the US, including China NSF grants (No. 60573131, No. 60673154), China 973 projects (No. 2002CB312002, No. 2006CB303000), U. S. NSF grant ACI-0203592 and U. S. NSF grant CCF-0524030. The purpose of the work aims to establish scalable techniques for large-scale structured P2P systems. The Sino-US joint research team have kept friendly collaboration for more than 4 years and produced fruitful results in the related fields. For example, the recently-proposed constant-degree P2P overlay network (Cycloid) was published in Chinese Journal of Computer in July 2005. This paper presents the authors' recent achievement in the study of DHT overlay networks for highly available file systems. A preliminary version of this paper was once presented on 3rd International Symposium on Parallel and Distributed Processing and Applications in 2005.