一种高可靠性的内容云的存储机制

张 瑞,林 闯,孟 坤,陈亚睿
(清华大学计算机科学与技术系,北京 100084)

摘 要: 云技术实现了异构硬件资源的有效整合,为向用户提供各种形式的服务奠定了技术基础.同时,由于使用的硬件资源性能、可靠性等差异巨大,所以如何使系统提供高效、高可靠性的服务是该领域研究的热点.本文对常用的可靠性和性能保证策略进行分析比较,探讨了系统能够容忍固定数目个硬件同时失效的必要条件,提出了一种既能保证服务效率,又能最大限度容忍硬件失效的内容存储机制(REST).该机制保证了在系统中同时出现故障的设备数不大于 M 时,系统仍能正常提供服务.最后,通过理论分析和仿真试验,我们验证了 REST 在存储效率、容失效能力等方面具有较大的优势.

关键词: 计算机网络; 内容云; 可靠性
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2014)04-0633-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.04.002

REST: A Reliable and Efficient Storage Technology for Content Cloud Service

ZHANG Rui, LIN Chuang, MENG Kun, CHEN Ya-rui
(*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)

Abstract: Cloud computing technology incorporates asymmetric hardware equipments and establishes the basis of providing constant and efficient service to the users. In this paper, intrigued from the redundancy backup strategy, we design a cloud storage mechanism which can tolerate hardware or software failures so as to ensure the reliability of the content cloud system while distributing the contents efficiently. Adopting this mechanism, the system can still operate smoothly in the presence of a fixed number of hardware or software failures, and if the number of failures is larger than that, the mechanism can protect the system from massive breakdowns. Afterwards, through experiments we compare several kinds of storage technologies, and validate that our technology takes advantages in storage efficiency and fault tolerance.

Key words: computer networks; content cloud; reliability

1 引言

云计算提供了一种整合相对廉价资源提供强大服务能力的计算模式,目的在于实现服务应用与硬件维护解耦,并分别由各层次的服务提供商维护管理,如 IaaS (Infrastructure as a Service)层服务提供商亚马逊(Amazon)通过封装底层硬件向用户提供计算或存储资源; PaaS (Platform as a Service)层服务提供商谷歌的产品 Google App Engine 建立了类操作系统环境平台,允许用户定制个性化的功能模块并开展二次开发; SaaS (Software as a Service)层服务提供商 Salesforce 整合了多种应用软件方便用户定制所需的软件系统等.通过层次化解耦,一方面各服务提供商可以屏蔽底层的管理,专心发挥其特长;另一方面用户只需关注自身需要的服务内容和服务质量[1].因而,云计算模式的采用不仅可以提高资源利用率、更好的保证服务质量,而且更重要的是,集中化管理降低了系统的综合维护成本.

根据云的部署方式和应用范围的不同,云可以划分为私有云、社区云、公共云和混杂云[2].私有云和社区云仅向特定的用户群体提供服务,而公共云和混杂云则服务更广泛的用户群体.目前,对外提供服务的云系统已达到了相当规模,用户在选择云服务时具有较大自由度,存在有多种服务租赁策略(或内容存储策略).云层次化的结构决定了云服务提供商不仅可以建立自己独立的服务系统,还可以被上层的内容提供商用来简化应用系统的搭建,如 SaaS 提供商可以租赁 PaaS 提供商的

收稿日期:2013-01-22;修回日期:2013-04-23;责任编辑:蓝红杰
基金项目:国家 973 重点基础研究规划(No.2010CB328105, No.2011CB302703);国家自然科学基金重点项目(No.60932003, No.61020106002);国家自然科学基金(No.61071065, No.11171368)

平台资源或 IaaS 层提供商的硬件资源.因此,资源利用策略的设计对于云计算系统具有普遍的意义,不仅有利于终端用户选择最佳的服务提供商组合以保证得到满意的服务质量,而且还能够使内容提供商选取优化的底层服务商组合来保证更好的提供服务.

以提供内容分发服务的运营商为例,考虑到传输成本、存储成本及存储效率等因素,他们可以通过选取合适的云存储提供商集合来实现服务成本限制下的最优服务.但是,由于云系统失效频率和持续时间的增加,分散租赁云服务成为了选取策略的基本指导思想[3],而关于是否能够通过分散存储来保证服务持续可用及如何分散存储的研究还相对匮乏.本文考虑了云计算环境下保证服务性能的容错(失效)存储策略,从存储的角度给出了云计算服务模型并得到了相应的优化模型,通过分析,给出了一种轻量级可靠性和可用性兼顾的存储策略.

2 相关工作

2.1 存储视角的云服务系统

云计算采用集群技术把各种资源整合,提供强大的计算和存储能力;采用互联网技术给用户提供便捷的接入方式,从而形成云提供商负责资源管理和维护,云用户按需请求响应资源的服务系统模式.在云服务系统中,云服务提供商按照与用户的服务协议,给用户定制的资源分配相应的存储空间,并把用户的请求指向给空间;用户则根据自身需求存取其定制的资源.因此,宏观上云服务系统可以等价地看作是一个分布式存储系统,云服务提供商负责给用户分配存储空间,用户通过到分配空间存取资源获得服务.由于分配空间的位置、网络状况及单价费用等因素使得用户的使用体验差别很大,因此关于存储策略的研究成为了该领域研究的热点之一,这方面的研究同时也是提高系统服务能力的主要途径.

关于优化存储结构,在内容分发网络(Content Delivery Network)领域已得到了广泛研究,存储资源部署方式可以分为中心式、层次式及 P2P 形式等.在中心式拓扑中,包括一个中心服务器集群,所有的用户都向该中心服务器集群提交服务请求,并由中心服务器统一分配和调度各种内容和资源以满足用户的请求;层次化拓扑由具有层次化管理关系的服务器组成,中心服务器存放所有的内容资源,上一级服务器随机地把资源部署在下一级服务器上,当请求不能被满足时,该请求被提交到上一级服务器,直到得到满足请求的内容,并返回给请求用户;P2P 式的拓扑,也被称为无中心的层次化拓扑结构,内容被部署在服务器中,服务器间的内容分享采用 P2P 方式,通过缓存相应的数据可以提高用户的访问效率.

对于上述方式,一般地,用户从距其最近的服务器上得到请求内容不仅可以保证延时较小,而且还能够避免由网络传输所产生的额外费用.因此,就近分配存储资源被认为是根本的优化策略.但是,由于存储开销、网络带宽等条件的限制,存储策略的选择受到诸多因素的影响,下面我们将对一些优化存储模型和策略进行比较分析.

2.2 优化存储策略

在文献[4]中,作者针对内容分发网络的数据部署和分发问题,通过把内容副本分发到网络边缘服务器,实现最小化带宽消耗.在边缘节点存储容量有限的情况下,该问题被转化为背包问题,作者给出了一种贪心算法,并设计出一种带宽消耗不超过某一阈值的最优策略.文献[5]以用户体验为优化目标,在以请求访问延时为衡量指标的情况下,提出了根据内容请求率的平方根对缓存空间进行分配的最小化搜索空间算法.文献[6]全面总结了内容分发网络中数据部署与分发的优化算法.

在文献[7]中,作者考虑了在线社交网络数据在服务器具有地理分布差异的云系统中存储的问题.对于一个特定的用户及其 QoS 指标(内容存储偏好),定义存储策略的 QoS 为一个向量$(q_1,q_2,\cdots,q_N)$,其中 q_k 表示该策略能满足用户请求的内容被分配在其偏好的前 k 个服务器中的比例.文章中以存储产生的费用(包括存储数据产生的费用、云系统之间维护数据一致性产生的费用、适应动态变化产生的维护费用和优化机制产生的重定向费用)为优化目标,在满足上述 QoS 要求的条件下,给出了一种基于角色转换的数据部署启发式算法.

文献[8]考虑了数据中心间的带宽和各个数据中心的存储容量这两项约束条件,给出了一个基于数据中心请求的数据动态迁移的系统,设计了基于用户请求的地理位置分布信息来实现数据的动态部署的算法,一定程度上减少了对存储空间和数据中心间流量的需求,提高了用户的使用体验.此外,文献[9]中把数据存储到异构的多个云服务系统中来提高服务的可用性,作者定义了存储内容的云服务商受困因子,给出了一种满足用户服务协议要求的动态存储策略.

但是在现有工作中,关于优化策略的研究往往仅关注实现一定优化目标的数据存放的策略,而关于同一位置数据的存放机制的研究相对匮乏(如在文献[9]中,作者指出采用 RAID5 的存放策略来提高数据的安全性).下面我们将根据已有的研究成果,给出基于存储视角的系统模型,在形式化描述系统的基础上,设计

一种能保证优化策略得以实施的高可靠性存储机制.

2.3　优化存储模型

根据上述分析,云系统可以抽象为图 1 所示的模型:系统由云控制中心处理用户请求及内容的分发存储,用户通过边缘服务器向系统请求需要的内容,若本地存在则直接读取,否则由边缘节点向控制中心转发请求,若系统中存在所需内容,控制中心向用户发送重定向指令以保证用户获取服务.显然,上述系统的集中式查询设计容易引起控制中心负载过重,成为系统瓶颈,减少查询耗时成为云存储领域关注的重点.一方面,通过采用与 Google App Engine 类似的基于域名系统(Domain Name System)的查询结构可以实现较高的查询效率;另一方面,可以通过维护尽量多的内容副本避免硬件失效对服务性能的影响,并通过由控制中心维护相关数据信息来提高访问效率.如何设计副本存储策略及如何存储副本的分布信息是本文讨论的重点.本文旨在给出一种能够容忍硬件失效、并能保持访问性能的内容存储机制.

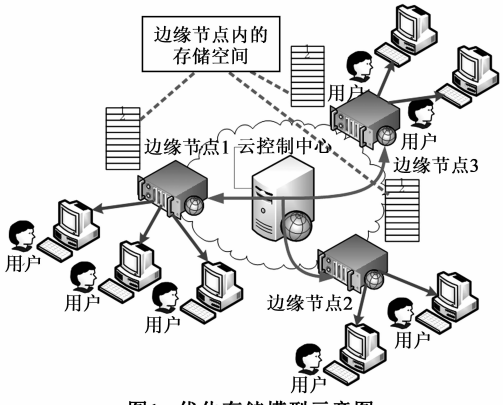

图1　优化存储模型示意图

在设计高可靠性的存储策略时,我们采用下列模型来描述本文要研究的问题:假设需要存储的资源集合为 $S=(S_1,S_2,\cdots,S_N)$,存储服务器集合为 $T=(T_1,T_2,\cdots,T_M)$,且对应的存储空间分别为 $B_1,B_2,\cdots,B_M$,存储服务器失效的概率分别为 $Pr_1,Pr_2,\cdots,Pr_M$.我们将针对上述模型,分别从选取合适的存储内容、容错存储策略等方面讨论高可靠性的内容云的存储机制.

3　边缘服务器的内容存储策略

云系统承载着越来越多的应用,如在线社交网络、在线网络存储等,这些应用的服务质量需求各不相同,因此需要采用的存储策略也不尽相同.根据应用的不同,边缘服务器上需要存储的内容可以分为基于用户请求的内容和基于服务推送的内容.例如,在线社交网络中生成和传输的数据就是基于用户请求的内容[10,11],而线上应用商店 Appstore 或 Android 中推广的应用软件可以看做是基于服务推送的内容.

3.1　基于用户请求的内容存储

由于用户使用偏好及其地理位置的不同,需要针对用户需求,选择合适的内容存储在可用的服务器资源上.对于上述为满足用户需求而存储的数据,我们可以称之为基于用户请求的数据.

我们用以下模型来描述这类数据的存储原则:对于存储服务器 N,假设其存储容量为 V,通过该节点接入实现请求的内容集合为 C_N,那么对于内容 $c\in C_N$,设该内容需要占用的存储空间大小为 V_c,$\Pr(c)$是某一内容请求是对内容 c 的请求的概率,且 $\sum_{c\in C_N}\Pr(c)=1$.设 $C_N^C\subseteq C_N$ 是存储在节点 N 上的数据集合,那么 C_N^C 可以通过求解下面的规划问题得到.

$$\max\sum_{c\in C_N^C}\Pr(c),\quad \text{s.t.}\sum_{c\in C_N^C}V_c\leqslant V$$

由于 M 是边缘节点数,B 是每个边缘节点的缓存大小,所以 MB 就是所有边缘节点的缓存大小之和.我们根据存储容量和待存储内容数量 N 间的关系针对下面几种情况给出选取条件:当 $N\leqslant B$ 时,每个边缘节点备份所有的内容,问题很容易得到解决;当 $MB\geqslant N\geqslant B$ 时,存储空间较充足,可以实现所有内容在所有边缘节点中至少有一个备份,那么此时的待存储内容为所有内容;当 $MB<N$ 时,不存在使每项内容都至少存在一个备份的缓存分配策略.考虑所有的组合,确定使待存储集合被访问概率最大的集合为备选集合,即

$$N=\{S\mid S\subseteq P,\Pr(S)=\max_{T\subseteq P}\{\Pr(T)\}\}.$$

3.2　基于服务推送的内容存储

为了让更多用户方便发现和使用有关服务内容,内容提供商往往选择把内容部署在合适的云服务系统上,我们把这类内容称之为基于服务推送的内容.对于这类内容,内容提供商为了提高内容的覆盖范围或针对性,一般会选择一个或多个云服务系统部署内容.通常这类问题中往往存在部署成本和存储空间的限制,因此可以把问题抽象化为优化问题来求解.

网络内容提供商通过把内容推送到云存储节点上可以实现更多用户对其内容的高效访问,提升内容的影响度.对为满足推广需求而产生的存储数据,我们称之为基于服务推送的数据.内容提供商根据对用户可能访问内容的预测确定最佳的存储范围,用 S 表示存储有某一项内容 C 的云存储节点的集合,如果某个云存储节点 $N\in S$,那么数据 C 是节点 N 的待存储数据.

4　REST:容忍节点失效的存储策略

若内容 i 被同时存储在两个不同的服务器上,那么

即使其中一个失效,仍不影响用户对该内容的访问.因此,如果内容被同时存放在 $r+1$ 个服务器上,那么在最坏的情形下该内容可以容忍 r 个服务器同时失效.一般来说,内容被尽量多的存储在不同的服务器上则对应着该策略具有较高的容失效能力.这样,问题的目标就转换为了在有限的存储空间中实现所有待存储内容尽可能多的被异构存储,即对于所有内容 $\{s_1, s_2, \cdots, s_v\}$,$M$ 个服务器 $\{server_1, server_2, \cdots, server_M\}$,我们需要采用恰当的分组设计方法使所有内容均衡的分布于各个分组中.下面我们首先给出区组设计的相关概念.

定义 1 对于集合 $S=\{s_1, s_2, \cdots, s_v\}$,$2^S$ 为 S 的幂集,(S, B)为 S 上的一个设计(design),其中 $B=\{B_1, B_2, \cdots, B_M\}$,$B_i \in 2^S$ 对应存储在设备 i 上的内容集合,我们称 B 为区组集合.如果至少有一个区组没有完全包含 S 中的所有元素,称该设计为不完全设计(Incomplete);如果一个设计中的所有区组都具有相同的容量,则称该设计为区组设计(Block design).当任意一对元素在设计中相遇的次数 λ_m 也相同,称该设计为均衡的(Balanced).

假设云系统中有 M 个存储设备,且对应的存储容量分别为 $B_i, i=1,2,\cdots,M$;N 是待存储的内容数,我们需要确定能容忍同时失效存储设备的最大个数 r,及相应的优化存储策略.由于存储容量的限制($|B_i| < N$),使得可实施的存储策略往往对应不完全设计,因此我们可以采用区组设计的方法逐步构造最优的分组;同时均衡的区组设计可以保证关联度较高的数据能够联合存储,提高数据的存取效率.

一个简单的存储分配机制是按照每个内容项的顺序在边缘节点中依次备份,当一个边缘节点的存储空间不足以备份所有内容项时,在下一边缘节点继续顺序备份内容项,直到所有内容项依次得到备份,再重新从第一个内容项开始重复备份过程.该算法实现的就是存储空间的平均分配策略,即在不考虑内容利用率情况下的最优策略.但该分配方式明显的缺点是各个边缘节点存储的内容显著不同,即某几个特定的边缘节点如果发生错误时,会发生某些内容同时失效的情况,因此我们要在满足存储需求的情况下,设计出负载均衡的存储分配机制.

我们说一个存储机制是均衡的,通常包括两个方面,一方面是内容之间均衡,即某项内容与其他各项内容同时出现的概率相同;另一方面是内容在边缘节点处的分配均衡,即某项内容在各个边缘节点处存储的概率相同.我们称同时满足两方面均衡条件的存储分配机制是对称均衡的存储机制.我们用 λ_e 表示两项内容同时出现在一个边缘节点处的次数,用 λ_b 表示任意两个边缘节点存储有相同内容的项数,那么对称均衡机制可以用数学表达式表示为 $\lambda_e=\lambda_b=\lambda$.

设 $N, M, r\text{-}1, B, \lambda_e$ 分别表示待存储资源的个数,可存储的服务器个数,最大允许服务器失效的个数,所有服务器中最小的容量,以及任何两个资源被分配在同一个服务器上的次数.那么可以得到以下均衡存储分配机制存在的必要条件.

引理 设每项内容的备份数为 r,则对称均衡机制存在的必要条件是

$$\begin{cases} MB = Nr \\ r(B-1) = \lambda_e(N-1) \\ B(r-1) = \lambda_b(M-1) \end{cases} \tag{1}$$

证明 计算一个存储分配机制中所包含的所有内容数有两种计算方法,一种是将边缘节点数 M 乘以每个边缘节点的存储空间大小;另一种是将内容数 N 乘以每项内容的备份数 r,两种计算方法所得结果相同,这样条件 1 得证.与证明条件 1 类似,对于条件 2,等式两边都表示与某一特定内容同时在边缘节点处存储的其他内容的总项数.对于条件 3,等式两边都表示选定边缘节点的内容在其他边缘节点处存储的次数.这样条件 2 和条件 3 不难得到证明.

根据以上的三个必要条件,我们可以进一步推导出对称均衡机制存在的充分必要条件.

定理 1 当且仅当式(1)中的三个等式成立且 $N=M$ 时,对称均衡机制存在.

证明 由引理中的第二项和第三项等式,我们有

$$\begin{cases} \lambda_e = \dfrac{r(B-1)}{(N-1)} \\ \lambda_b = \dfrac{B(r-1)}{(M-1)} \end{cases}$$

因为在对称均衡存储机制中 $\lambda_e=\lambda_b=\lambda$,所以 $r(B-1)(M-1)=B(r-1)(N-1)$.经过化简后得到 $(N-M)(MB-(N+M)+1)=0$.因为上式中的第二个乘数不等于 0,所以 $N-M=0$,即 $N=M$ 成立.

我们用决策变量 x_{in} 来表示第 n 项内容在第 i 个边缘节点是否存在备份,即节点 i 存有内容 n,则变量 $x_{in}=1$;节点 i 内没有内容 n,则变量 $x_{in}=0$.所有决策变量 x_{in} 组成了一个 $M\times N$ 维矩阵,该矩阵称为存储空间分配的关联矩阵.如果一个内容云系统的存储分配机制是对称均衡的,那么该分配机制的关联矩阵要满足定理 2 中描述的必要条件.

定理 2 设 $\boldsymbol{A}$ 是存储空间分配的关联矩阵,则对于对称均衡机制来说,它的关联矩阵要满足下列等式:

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} = (r-\lambda_e)\boldsymbol{I}_N + \lambda_e \boldsymbol{J}_N.$$

其中,r 是每项内容的备份数,$\boldsymbol{I}_N$ 为 N 阶单位矩阵,$\boldsymbol{J}_N$ 表示元素全为 1 的 N 阶方阵.

证明　令 $\boldsymbol{H}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=(h_{ij})_{N\times N}$,则 h_{ij}为矩阵 $\boldsymbol{A}^{\mathrm{T}}$ 的第 i 行与矩阵 $\boldsymbol{A}$ 第 j 列的内积,也就是矩阵 $\boldsymbol{A}$ 的第 i 列与第 j 列的内积. h_{ii}为矩阵 $\boldsymbol{A}$ 第 i 列中非零元素的个数,所以 $h_{ii}=r$. 当 $i\neq j$ 时,$h_{ij}=\lambda_e$,即在第 i 列与第 j 列中对应元素都等于 1 的个数为 λ_e. 因此,$\boldsymbol{H}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=(r-\lambda_e)I_N+\lambda_e J_N$.

定理 2 说明了对称均衡存储机制的关联矩阵的转置与矩阵本身的乘积是一个对称矩阵,且该对称矩阵的主对角线元素全部为 r,非主对角线元素全部为 λ_e. 该定理从直观上不难理解.

假设内容项 n 在云系统中存在的备份数是 s_n. 在随机搜索策略下,找到某个内容项 n 需要搜索的边缘节点数是一个随机变量 q_n,且 q_n 服从几何分布. 因为 q_n 服从几何分布,所以我们计算随机变量 q_n 的期望值 $E(q_n)=\dfrac{N}{s_n}$,即内容项 n 的平均搜索空间大小是$\dfrac{N}{s_n}$,其中 N 是云系统中的内容数. 我们对每项内容的 q_n 求平均,得到搜索所有内容的平均搜索次数为 $\sum_n(p(n)\cdot E(q_n))=N\sum_n p(n)/s_n$.

为了使搜索空间最小,必须有若 $p(n_1)>p(n_2)$,则云系统中内容 n_1 的备份数大于等于内容 n_2 的备份数. 否则如果内容 n_1 的备份数小于内容 n_2 的备份数,那么将原来备份 n_1 的存储空间用来备份 n_2,将原来备份 n_2 的存储空间用来备份 n_1,所得备份方案的搜索空间更小. 假设边缘节点 i 在一段时间内收到对内容 n 的请求数量为 k_{in},我们用 $c_{jn}(j=1,2,\cdots,k_{in})$ 表示这 k_{in}个请求的搜索空间的大小. 令 $C_{in}=\sum_{j=1}^{k_{in}}c_{jn}$,则每个边缘节点处都对内容 n 储存(k_{in},C_{in}). 我们用 V 表示对内容 n 的搜索空间中边缘节点的集合. 系统中对内容 n 的请求的到达速率 $\mu_n=\sum_{i\in V}k_{in}/|V|$,搜索空间的大小 $A_n=2\sum_{i\in V}C_{in}/\sum_{i\in V}k_{in}$.

在接受内容请求的边缘节点不存在所需内容时,该边缘节点从其他边缘节点或中心服务器获取请求的内容,并需要将该内容放入自身的缓存中,这就涉及到边缘节点缓存的内容替换算法(缓存内容是动态的,在动态的情况下要考虑内容替换算法). 两种通用的内容替换算法是 LRU(Least Recently Used)和随机替换算法. 在某一项内容被请求后,系统通常会增加该内容的备份(至少该内容的备份数保持不变). 而内容的某个备份也不会永久地存在于云系统中. 通常来说,一个备份在系统中存在的时间越长,那么该备份被新备份替换的概率也就越大,即若系统中存在两个内容的备份,如果备份一的产生时间早于备份二,那么在某一时刻,备份一在系统中仍然存在的概率要小于备份二.

5　仿真与数字试验

我们通过仿真试验来对我们设计的内容云系统存储机制的性能进行分析比较. 假设我们要设计一个简单的云系统,其中内容数有 16 个,每个边缘节点的存储空间可以容纳 6 项内容(下文还分析了内容数更多,存储空间更大情况下的系统性能),那么是否存在对称均衡的存储分配机制?

实例中 $N=16$, $B=6$,取均衡数 $\lambda=2$,我们设计得到的存储分配机制有以下三种形式.

表 1　$\lambda=2$ 时的存储分配机制

Ⅰ	B_1\{1 2 3 4 5 6\}	B_9\{2 5 8 9 15 16\}
	B_2\{1 2 7 8 11 12\}	B_{10}\{2 6 10 12 14 16\}
	B_3\{1 3 9 11 14 15\}	B_{11}\{3 4 8 12 13 15\}
	B_4\{1 4 7 10 15 16\}	B_{12}\{3 5 7 12 14 16\}
	B_5\{1 5 8 10 13 14\}	B_{13}\{3 6 7 8 9 10\}
	B_6\{1 6 9 12 13 16\}	B_{14}\{4 5 9 10 11 12\}
	B_7\{2 3 10 11 13 16\}	B_{15}\{4 6 8 11 14 16\}
	B_8\{2 4 7 9 13 14\}	B_{16}\{5 6 7 11 13 15\}
Ⅱ	B_1\{1 2 3 4 5 6\}	B_9\{2 5 8 9 15 16\}
	B_2\{1 2 7 8 11 12\}	B_{10}\{2 6 10 12 14 15\}
	B_3\{1 3 9 12 13 15\}	B_{11}\{3 4 8 11 14 15\}
	B_4\{1 4 7 10 15 16\}	B_{12}\{3 5 7 12 14 16\}
	B_5\{1 5 8 10 13 14\}	B_{13}\{3 6 7 8 9 10\}
	B_6\{1 6 9 11 14 16\}	B_{14}\{4 5 9 10 11 12\}
	B_7\{2 3 10 11 13 16\}	B_{15}\{4 6 8 12 13 16\}
	B_8\{2 4 7 9 13 14\}	B_{16}\{5 6 7 11 13 15\}
Ⅲ	B_1\{1 2 3 4 5 6\}	B_9\{2 5 8 9 14 16\}
	B_2\{1 2 7 8 11 12\}	B_{10}\{2 6 10 12 13 16\}
	B_3\{1 3 9 12 13 14\}	B_{11}\{3 4 8 11 13 16\}
	B_4\{1 4 7 10 14 16\}	B_{12}\{3 5 7 12 15 16\}
	B_5\{1 5 8 10 13 15\}	B_{13}\{3 6 7 8 9 10\}
	B_6\{1 6 9 11 15 16\}	B_{14}\{4 5 9 10 11 12\}
	B_7\{2 3 10 11 14 15\}	B_{15}\{4 6 8 12 14 15\}
	B_8\{2 4 7 9 13 15\}	B_{16}\{5 6 7 11 13 14\}

表 1 中我们用 B_i 表示第 i 个边缘节点的存储空间,用 B_i 后面的集合表示存储空间内储存的内容. 集合里的数字表示内容项的编号,即 B_1\{1 2 3 4 5 6\}表示第一个边缘节点内存储有前 6 项内容. 因此,在云系统内容数为 16,每个边缘节点能够存储 6 项内容时,存在对称均衡的存储机制,使用上面的三种分配方案可以很好地实现内容云系统的功能. 在设计边缘节点的存储空间为 6 项内容的存储机制之后,我们对该机制的性能和可靠性进行分析,对照的实验对象是一般的存储分配机制即顺序分配机制.

从图 2(a)可以看出,均衡分配机制在容失效能力方面要远好于一般的存储分配机制,特别是在有多个

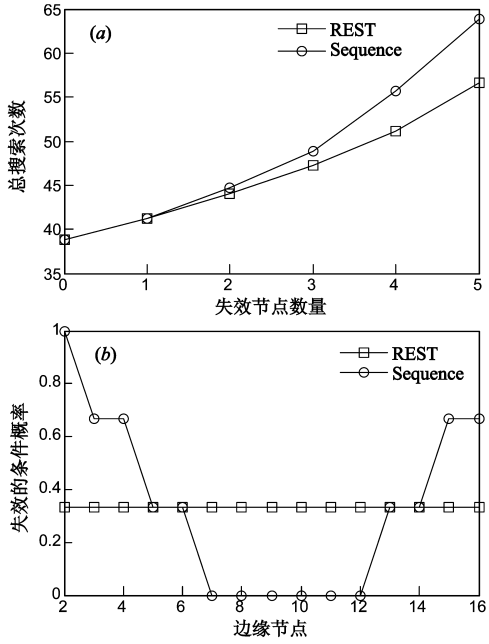

图2 B=6时云存储系统性能和可靠性的试验分析

边缘节点同时失效的情况下,均衡分配机制由于节点失效导致的性能损失较小,从而可以提供较高可靠性的内容云服务.

我们进一步对有更大的边缘节点数和内容数的云系统进行分析.当 $N=79$, $B=13$, $\lambda=2$ 时,比较均衡设计和一般的存储机制之间的性能区别如图3所示(假设超过一定搜索次数后不再继续搜索,而是向中心服务器发送请求).该试验是从最坏情况的角度进行分析,即计算存储有相似内容的几个边缘节点同时失效的情况,这时对云存储系统带来的性能损失最大.图3(a)中横坐标是失效边缘节点的数量,纵坐标是搜索各项内容的总次数,圆点连线构成的曲线表示一般分配机制,即我们在前文中提到的顺序分配机制.显然,总搜索次数随着失效节点数量的增多是会显著增加的.

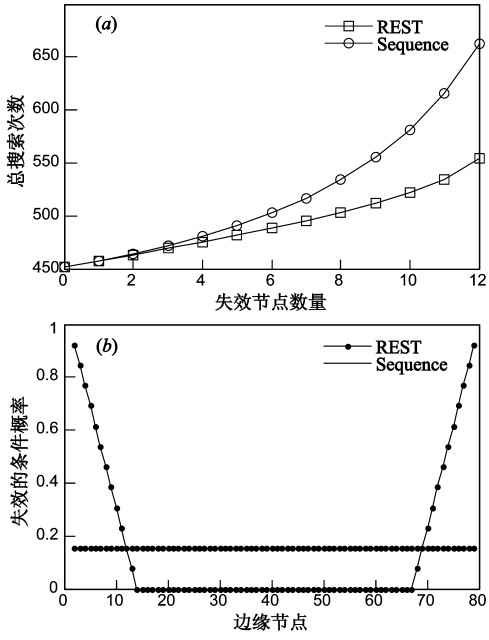

图3 B=13时云存储系统性能和可靠性的试验分析

图2(b)和图3(b)分别表示系统有16个节点和79个节点的情况下出现一个边缘节点失效时其他节点失效的条件概率.从两图中可以看出,均衡分配机制下节点失效的条件概率各个节点间是相同的,而一般分配机制各个节点的失效概率差别很大,这进一步证明了我们设计的内容云存储机制的高可靠性.

6 结论与下一步研究工作

本文对基于内容服务的云系统的性能进行了定量分析,设计了一种能够及时应对硬件和软件故障的云存储机制,以实现在不多于固定个故障出现的情况下,系统仍能正常提供高效服务.该机制采用容错设计的相关技巧,分别针对内容需求度不相关和相关的情形,设计了存储策略,并给出了最优性证明,得到了在硬件限制条件下最优的高可靠性内容云存储机制.未来工作可能包括考虑多目标的优化,同时优化延时和带宽消耗等.在我们通常的分析中,云存储系统的边缘节点是无差别的,但在一些实际的网络中,存在边缘节点倾向于向其他某些边缘节点传输数据的情况,这需要进一步的理论分析.

参考文献

[1] A Dhamdhere, C Dovrolis. Twelve years in the evolution of the internet ecosystem[J]. IEEE/ACM Transactions on Networking, 2011, 19(5): 1420 – 1433.

[2] P Mell, T Grance. The NIST definition of cloud computing (draft)[R]. NIST Special Publication, 2011.

[3] IBM Technique Report. Cloud computing for the media and entertainment industry[R/OL]. http://www.wired.com/insights/2012/11/amazon-google-outages, 2012.

[4] S C Borst, V Gupta, et al. Distributed caching algorithms for content distribution networks[A]. IEEE Conference on Computer Communications[C]. San Diego, USA: IEEE, 2010. 1 – 9.

[5] E Cohen, S Shenker. Replication strategies in unstructured peer-to-peer networks[A]. SIGCOMM Computer Communication Review[C]. Pittsburgh, USA: ACM, 2002. 32(4): 177 – 190.

[6] 尹浩,袁小群,等.内容网络服务节点部署理论综述[J].计算机学报,2010,33(9):1611 – 1620.
Yin Hao, Yuan Xiao-qun, et al. The survey of service nodes placement theories for content networks[J]. Chinese Journal of

Computers, 2010, 33(9):1611 – 1620.(in Chinese)
[7] L Jiao, J Li, et al. Cost optimization for online social networks on geo-distributed clouds[A]. The 20th IEEE International Conference on Network Protocols[C]. Austin, USA: IEEE, 2012.1 – 10.
[8] S Agarwal, J Dunagan, et al. Volley: Automated data placement for geo-distributed cloud services[A]. The 7th USENIX Symposium on Networked Systems Design and Implementation[C]. San Jose, USA: USENIX Association, 2010.17 – 32.
[9] T G Papaioannou, N Bonvin, K Aberer. Scalia: An adaptive scheme for efficient multi-cloud storage[A]. International Conference on High Performance Computing, Networking, Storage and Analysis[C]. Salt Lake City, USA: IEEE Computer Society Press, 2012.1 – 10.
[10] P Krishnan, D Raz, et al. The cache location problem[J]. IEEE Transactions on Networking, 2000, 8(5):568 – 592.
[11] J M Almeida, D L Eager. Minimizing delivery cost in scalable streaming content distribution systems[J]. IEEE Transactions on Multimedia, 2004, 6(2):356 – 365.
[12] J Kangasharju, K W Ross, et al. Optimizing file availability in peer-to-peer content distribution[A]. IEEE Conference on Computer Communications[C]. Anchorage, USA: IEEE, 2007.1973 – 1981.
[13] 杨戈,廖建新,等.流媒体分发系统关键技术综述[J].电子学报,2009,37(1):137 – 145.
Yang Ge, Liao Jian-xin, et al. Survey of key technologies of the distribution system for streaming media[J]. Acta Electronica Sinica, 2009, 37(1):137 – 145.(in Chinese)
[14] Mathias Bjorkqvist, Lydia Y Chen, et al. Minimizing retrieval latency for content cloud[A]. IEEE Conference on Computer Communications[C]. Shanghai: IEEE, 2011.1080 – 1088.
[15] Michael Armbrust, Armando Fox, et al. Above the clouds: A Berkeley view of cloud computing[R]. University of California, Berkeley, 2009.
[16] S Tewari, L Kleinrock. Proportional replication in peer-to-peer networks[A]. IEEE Conference on Computer Communications[C]. Barcelona, ESP: IEEE, 2006.
[17] 李建敦,彭建杰,等.云存储中一种基于布局的虚拟磁盘节能调度方法[J].电子学报,2012, 40(11):2247 – 2254.
Li Jian-dun, Peng Jian-jie, et al. A layout-based energy-aware approach for virtual disk scheduling in cloud storage[J]. Acta Electronica Sinica, 2012, 40(11):2247 – 2254.(in Chinese)
[18] S Buchholz, T Buchholz. Replica placement in adaptive content distribution networks[A]. ACM Symposium on Applied Computing[C]. Nicosia, Cyprus: ACM, 2004.1705 – 1710.
[19] S Zhou, J Katto, et al. Replication algorithms to retrieve scalable streaming media over content delivery networks[A]. The 5th ACM SIGMM International Workshop on Multimedia Information Retrieval[C]. Berkeley, USA: ACM, 2003. 255 – 261.
[20] Q Lv, P Cao, et al. Search and replication in unstructured peer-to-peer networks[A]. International Conference on Supercomputing[C]. New York: ACM, 2002.84 – 95.
[21] B-G Chun, K Chaudhuri, et al. Selfish caching in distributed systems: a game-theoretic analysis[A]. The Twenty-third Annual ACM Symposium on Principles of Distributed Computing[C]. St Johns, Canada: ACM, 2004.21 – 30.
[22] Xiangzhen Kong, Jiwei Huang, et al. Performance, fault-tolerance and scalability analysis of virtual infrastructure management system[A]. IEEE International Symposium on Parallel and Distributed Processing with Applications[C]. Chengdu, China: IEEE, 2009.282 – 289.
[23] 王丽娜,任正伟,等.一种适于云存储的数据确定性删除方法[J].电子学报,2012,40(2):266 – 272.
Wang Li-na, Ren Zheng-wei, et al. A data assured deletion approach adapted for cloud storage[J]. Acta Electronica Sinica, 2012, 40(2):266 – 272.(in Chinese)
[24] 吴吉义,傅建庆,等.一种对等结构的云存储系统研究[J].电子学报,2011,39(5):1100 – 1107.
Wu Ji-yi, Fu Jian-qing, et al. Study on the P2P cloud storage system[J]. Acta Electronica Sinica, 2011, 39(5): 1100 – 1107.(in Chinese)

作者简介

张　瑞　男,1986 年出生于山东东营,清华大学计算机系博士研究生.研究方向为网络安全性分析、云存储服务系统的性能评价.
E-mail: zhangrui@csnet1.cs.tsinghua.edu.cn

林　闯　男,清华大学计算机系教授.研究方向包括计算机网络、性能评价、网络安全性分析以及 Petri 网理论及其应用.

孟　坤　男,清华大学计算机系博士后.研究方向为计算机网络的性能评价、网络安全性分析、服务计算和随机模型.

陈亚睿　女,清华大学计算机系博士后.研究方向为计算机网络、性能评价、用户行为分析和网络管理.