文章编号:1001 - 2486(2004)03 - 0039 - 04

基于 Tornado 码的复制算法*

王意洁,卢锡城
(国防科技大学计算机学院,湖南 长沙 410073)

摘 要:面向 Internet 的分布存储系统具有数据种类多、数据量大、分布广泛等特点,为了提高分布存储系统的数据访问效率,提出了一种基于 Tornado 码的复制算法。与传统的复制算法相比,基于 Tornado 码的复制算法能够提供更高的可用性、持久性和安全性,并且具有更低的存储开销和带宽开销。

关键词:分布存储;Internet;复制;编码;数据可用性;存储开销;带宽开销
中图分类号:TP393 文献标识码:A

The Tornado-code-based Replication Algorithm

WANG Yi-jie, LU Xi-cheng
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: In the Internet-based distributed storage system, the mass heterogeneous data are widely distributed. In order to improve the efficiency of data access, the Tornado-code-based replication algorithm is proposed. Compared with the traditional replication algorithms, the Tornado-code-based replication algorithm can provide better availability, durability and security, and has less storage cost and bandwidth cost.

Key words: distributed storage; Internet; replication; coding; data availability; storage cost; bandwidth cost

随着网络技术和计算机技术的迅速发展,Internet 已经逐步延伸到世界的每一个角落,人们可以随时随地登录 Internet 查找需要的各种数据,越来越多的企业、研究机构、政府部门和商家将自己的数据资源与 Internet 相连,从而构成庞大的分布存储系统,以此提高数据资源的利用率。

复制技术是提高分布存储系统数据访问效率的有效途径之一。目前,许多研究计划都利用传统的复制算法实现分布存储系统的高可用性和持久性[1-5]。传统的复制算法创建与数据对象大小相同的副本,随着系统规模的增加,需要不断增加副本数目才能保证系统的可用性和持久性;但是,副本数目的增加在某种程度上也增加了对网络带宽和系统存储能力的需求,特别是当数据对象较大时,这种需求更加迫切;另外,副本数目的增加也会对数据安全性产生影响。为了克服传统复制技术的不足之处,我们提出了基于 Tornado 码的复制算法,它先将数据对象分割成若干个数据块,然后对分割后的数据块进行 Tornado 编码,使数据块之间存在一定的数据冗余,从而有效实现系统的高可用性和持久性,并在一定程度上保证了数据的安全性。

1 基本思想

面向 Internet 的分布存储系统中的数据具有数据量大、数据对象规模大等特点。传统的复制算法[6]以数据对象为基本单位进行复制,其研究重点在于副本数目的设置、副本的放置、副本的数据一致性维护等,处理大规模数据对象时将会引起大量的带宽开销和存储开销,并对数据安全性产生影响。因此,传统的复制算法难以很好满足面向 Internet 的分布存储系统的需求。

针对大规模数据对象的特点,将编码理论应用于分布存储系统中,我们提出了基于 Tornado 码的复

* 收稿日期:2004 - 02 - 01
基金项目:高等学校全国优秀博士学位论文作者专项资金资助项目(200141);国家重点基础研究专项 973 计划资助项目(2002CB312105);国家自然科学基金项目(69933030,69903011)资助
作者简介:王意洁(1971—),女,博士,副教授。

制算法，它首先将需要复制的数据 D 分割为 m 个大小为 b 的数据块，然后对分割后的数据块进行 Tornado 编码[7]，将编码产生的 n 个大小为 b 的数据块分布到多个结点上；需要数据访问时，从相关结点读取 r 个数据块（$r > m$），对它们进行解码，即可恢复出原数据 D。这里，$e_r = m/n$ 称为编码率。Tornado 码是系统（systematic）码，所以编码后的前 m 个数据块就是原来的 m 个数据块，前 m 个数据块称作信息块，其余的 $n - m$ 个数据块称作校验块。与传统的复制算法相比，基于 Tornado 码的复制算法的复制粒度小，带宽开销和存储开销小，能够提供高可用性和持久性，并在一定程度上保证数据安全性。

2 算法设计

基于 Tornado 码的复制算法主要包括以下步骤：(1)数据分割：将数据 D 分割为 m 个大小为 b 的数据块，即，$D = [d_1, d_2, \cdots, d_m]$；(2)编码：对 m 个数据块进行编码，产生出 n 个数据块，这些数据块分布地存储在 Internet 上的多个结点上；(3)解码：从当前可用的结点上获取 r 个数据块，对它们进行解码得到原来的 m 个数据块；(4)恢复数据：由解码得到的 m 个数据块恢复数据 D。下面，详细介绍算法的核心步骤：编码和解码。

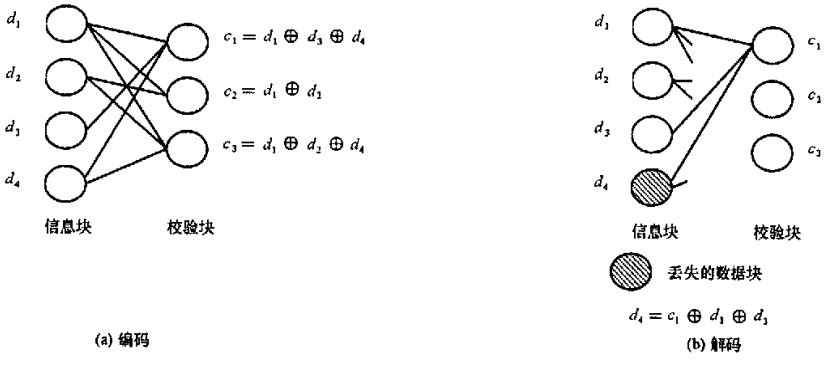

图1 二分图
Fig.1 Bipartite graph

我们利用二分图 B 对数据块 $d_1, d_2, \cdots, d_m$ 进行编码，产生包括 $d_1, d_2, \cdots, d_m$ 在内的 $(1+\beta)m$ 个数据块，即产生 m 个信息块和 βm 个校验块。二分图 B 定义了信息块与校验块之间的映射关系，每个校验块可以由其相邻的信息块进行异或运算得到（如图 1(a)）。因此，编码的计算开销与二分图 B 中的边数成正比。如果与某一校验块相邻的所有信息块中存在一个丢失的信息块，那么可以通过该校验块和其它信息块的异或运算得到丢失的信息块（如图 1(b)），这就是解码的基本思想。

显然，我们可以利用所有校验块和部分信息块得到丢失的信息块，从而恢复原数据 D。但是，如果某些校验块丢失，那么有可能影响到原数据的恢复。为了解决这个问题，可以利用级联二分图实现编码。级联二分图（如图 2）由若干个二分图 $B_0, B_1, \cdots, B_k, B_{k+1}$ 连接而成，其中，B_k 和 B_{k+1} 的左结点是相同的，也就是说，对相同的数据块计算两次校验块，其目的是提高解码概率。利用 B_0 可以由 m 个信息块产生 βm（$\beta < 1$）个校验块，这 βm 个校验块就是 B_1 的信息块，利用 B_1 又可以产生 $\beta^2 m$ 个校验块，依此类推，利用 B_k 可以由 $\beta^k m$ 个信息块产生 $\beta^{k+1} m$ 个校验块；利用 B_{k+1} 可以由 $\beta^k m$ 个信息块产生 $\beta^{k+1} m$ 个校验块。因此，利用编码 $C(B_0, B_1, \cdots, B_k, B_{k+1})$ 产生的校验块的数目为 $\sum_{i=1}^{k+1} \beta^i m + \beta^{k+1} m$。

通常，解码从最高级二分图开始，首先利用 B_k 和 B_{k+1} 得到它们的信息块，即 B_{k-1} 的校验块；然后，依次得到 $B_{k-2}, \cdots, B_1, B_0$ 的校验块；最后，得到 B_0 的 m 个信息块。当然，如果在解码开始时就已经得到了 $B_i (0 \leqslant i < k)$ 的所有校验块，那么，可以直接从 B_i 开始解码。

基于 $B_0, B_1, \cdots, B_k, B_{k+1}$ 的编码和解码的计算开销与二分图中的边数成正比，基于 Tornado 码的复

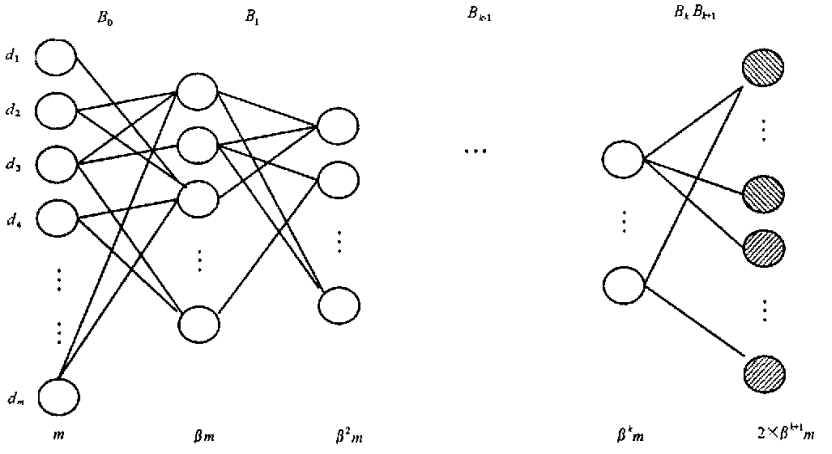

图 2 级联二分图
Fig.2 Cascade bipartite graphs

制算法的编码和解码的计算复杂度均为 $O(mb)$。

3 性能评价

3.1 基于 Tornado 码的复制算法与传统复制算法的对比分析

1. 数据可用性

设数据 D 分割后被编码为 n 个数据块,需要其中 r 个数据块才能恢复数据 D,分布存储系统中共有 N 个结点,当前不可用的结点数为 M,数据 D 的可用性 P_D 可以表示为:

$$P_D = \sum_{i=0}^{n-r}\left[\binom{M}{i}\binom{N-M}{n-i}\Big/\binom{N}{n}\right] \tag{1}$$

如果分布存储系统由 1 000 000 个结点构成,其中 10% 的结点不可用,那么利用传统复制算法存储数据 D 的两个副本可获得 0.99 的可用性;利用编码率为 0.5 的基于 Tornado 码的复制算法对数据 D 的 32 个数据块进行编码可获得 0.999999998 的可用性。显然,在相同存储开销和带宽开销的情况下,基于 Tornado 码的复制算法可以获得更高的可用性。

2. 存储开销

在传统复制算法中,关于数据 D 的存储开销为数据大小与副本数目的乘积,即 $S_t = RD$,其中,R 为数据 D 的副本数目;在编码率为 e_r 的基于 Tornado 码的复制算法中,关于数据 D 的存储开销为 $S_n = D/e_r$。所以,$S_t/S_n = (RD)/(D/e_r) = R \cdot e_r$。

由(1)式的分析可知,如果要求达到相同的数据可用性,那么,$R > 1/e_r$,所以,$S_t/S_n > 1$,即传统复制算法的存储开销大于基于 Tornado 码的复制算法的存储开销。

3. 带宽开销

复制算法的带宽开销包括读访问的带宽开销和写访问的带宽开销。

传统复制算法的读访问带宽开销为数据大小,即 $W_r = D$;基于 Tornado 码的复制算法的读访问带宽开销为用于恢复数据 D 的数据块的大小,即 $W_{tr} = (r/m)D$,其中,数据 D 被分割为 m 个数据块,r 为用于恢复数据 D 的数据块的数目。$W_r/W_{tr} = D/[(r/m)D] = m/r$,对于基于 Tornado 码的复制算法而言,$r$ 略大于 m,所以,W_r/W_{tr} 略小于 1,即传统复制算法的读访问带宽开销略小于基于 Tornado 码的复制算法的读访问带宽开销。

传统复制算法的写访问带宽开销为数据大小与副本数目的乘积,即 $W_{tw} = RD$,其中,R 为数据 D 的

副本数目;编码率为 e_r 的基于 Tornado 码的复制算法的写访问带宽开销为数据 D 编码后的数据块的大小,即 $W_{tw}=D/e_r$。$W_{tw}/W_{nc}=RD/(D/e_r)=R\cdot e_r$,由(1)式的分析可知,如果要求达到相同的数据可用性,那么,$R>1/e_r$,所以,$W_{tw}/W_{nw}>1$,即传统复制算法的写访问带宽开销大于基于 Tornado 码的复制算法的写访问带宽开销。

3.2 数据大小对基于 Tornado 码的复制算法效率的影响

数据 D 被分割为 m 个数据块,编码后产生 n 个数据块,基于 Tornado 码的复制算法需要 $(1+\alpha)m$ 个数据块用于恢复数据 D,这里,我们取 $\alpha=0.05$。利用 C 语言实现基于 Tornado 码的复制算法,测试平台为 P4 1.5G 微机(CPU 为 P4 1.5GHz,内存为 256MB)。选取数据块大小为 1KB,编码率为 0.5。调整数据大小,观察编码时间和解码时间的变化(如表 1)。

由第 2 节可知,基于 Tornado 码的复制算法的编码和解码的计算复杂度均为 $O(mb)$,即,与数据大小成正比。表 1 的测试结果与理论分析基本一致。

表 1 数据大小对算法效率的影响

Tab.1 Influence of data size on efficiency of indirect replication algorithm

数据大小(B)	编码时间(s)	解码时间(s)
250K	0.005	0.004
500K	0.011	0.007
1M	0.023	0.013
2M	0.049	0.021
4M	0.116	0.037
8M	0.253	0.082

4 结论

基于 Tornado 码的复制算法对分割后的数据块进行编码,使数据块之间存在一定的数据冗余,与传统复制算法相比,基于 Tornado 码的复制算法能够提供更高的可用性、持久性和安全性,有效降低系统开销。对基于 Tornado 码的复制算法的测试结果表明,基于 Tornado 码的复制算法的实用性较高,其编码时间和解码时间与数据大小成正比,用于恢复数据的数据块数目越多,数据恢复的成功率越高。

参考文献:

[1] Kubiatowicz J, Bindel D, Chen Y, Czerwinski S, et al. OceanStore: An Architecture for Global-scale Persistent Storage[C]. Proc. Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX), ACM Press, New York, 2000:190-201.

[2] Stoica I, Morris M, Karger D, Kaashoek M F. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications[C]. ACM SIGCOMM 2001, San Deigo, CA, 2001:160-177.

[3] Druschel P, Rowstron A. PAST: A Large-scale, Persistent Peer-to-peer Storage Utility[C]. In: Proc of HotOS Ⅷ, Schloss Elmau, Germany, 2001: 75-80.

[4] Cohen E, Shenker S. Replication Strategies in Unstructured Peer-to-peer Networks[C]. In the ACM SIGCOMM'02 Conference, Pittsburgh, USA, August 2002:308-321.

[5] Kangasharju J, Roberts J, Ross K W. Object Replication Strategies in Content Distribution Networks[C], In Proceedings of WCW'01: Web Caching and Content Distribution Workshop, Boston, USA, June 2001:252-201.

[6] Plaxton C G, Rajaraman R, Richa A W. Accessing Nearby Copies of Replicated Objects in a Distributed Environment[C]. In Proc. 9th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'97), Newport, RI, USA, 1997:311-320.

[7] Luby M G, Mitzenmacher M, Shokrollahi M A, Spielman D A, Stema V. Practical Loss-resilient Codes[C]. In Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing, El Paso, Texas, USA, May 1997:150-159.