文章编号:1001-2486(2007)03-0061-04

P2P系统中数据复制算法研究*

王意洁,张小明,周 婧
(国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:比较分析了P2P系统中各种数据复制算法,并提出了一种基于LDPC编码的数据复制算法Dyre,数据块采用动态分配算法存储到节点中,在节点邻居中保存数据块的副本以提高数据块的有效性,数据块的数量过小时重建数据块。实验表明,即使节点的可靠性非常低,该算法也能够获得很高的数据可用性。

关键词:peer-to-peer;数据可用性;数据复制
中图分类号:TP393 **文献标识码**:A

Research of Data Replication Algorithm in P2P Systems

ZHANG Xiao-ming, WANG Yi-jie, ZHOU Jing
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: A comparative analysis of replication algorithms in P2P systems and a novel replication algorithm Dyre based on LDPC codes were presented. Dynamic replica placement was proposed. With this method, blocks are stored in predecessor and success nodes to improve the availability, and blocks are restored when the number of available blocks is small in this algorithm. Simulation results show that the algorithm can significantly enhance the data availability, even when the node reliability is low.

Key words: peer-to-peer; data availability; data replication

支持数据的可用性、持久性和完整性已成为P2P存储系统的一个关键问题,P2P系统主要采用数据复制技术来解决这些问题。目前P2P系统中的数据复制策略有全复制、分块复制和基于erasure[1]编码的复制三种。

在Oceanstore[2]中每个数据被分割成16块,然后利用erasure把这16块数据编码成32块数据。Dhash[3]用IDA加密算法把数据分割成的7块,编码成14块数据,同时采用局部和全局两个维护策略,局部的维护策略保证编码数据块不小于14,全局的维护策略保证数据块数目不大量超过14块。

在相同的存储开销情况下,利用erasure编码可以比传统的复制算法获得更高的数据可用性。但是在Oceanstore和Dhash中,编码块的位置信息都保存在一个节点上,数据块的访问都需先从该节点上获取位置信息,这种放置算法容易产生瓶颈问题,同时当数据丢失时没有考虑数据的重建。本文提出了一个基于LDPC(low-density parity-check)编码的数据复制算法Dyre,采用动态的数据块放置算法,数据的位置信息可以通过直接计算得到,不需要节点保存数据位置信息。

1 erasure数据编码

erasure数据编码利用数学的方法把 m 块数据变换成 n 块同样大小的数据,编码后的 n 块数据中的任意 m 块数据都可还原成原来的 m 块数据。因此一个 (m,n) 的MDS编码可以容忍 $n-m$ 个节点的失效。数据编码包括数据分割、编码、解码和还原数据等步骤。

在erasure编码中,一个数据被分割成 b 块,编码后的数据至少需要 b 块才能还原成原来的数据,存储开销为 S,相当于每个数据块有 S 个副本,因此系统中拥有 $S\times b$ 块数据。erasure编码通过利用数据块之间的相关性来提高数据的可用性,编码后的 $S\times b$ 块数据相互利用,并且这些数据块之间的任意 b

* 收稿日期:2007-01-06
基金项目:国家部委基金资助项目(2002CB312105);高等学校全国优秀博士学位论文作者专项资金项目(200141)
作者简介:王意洁(1971—),女,教授,博士生导师。

块数据都能够还原成原来的完整数据。因此利用 erasure 编码数据的可用性为：

$$A_b(b) = \sum_{i=b}^{Sb} \binom{Sb}{i} u^i (1-u)^{Sb-i} \tag{1}$$

前提条件相同情况下的全复制方法中数据的可用性为：

$$A_w(S) = \sum_{i=1}^{s} \binom{s}{i} u^i (1-u)^{s-i} \tag{2}$$

其中，u 表示节点的可靠性。传统复制算法读访问数据的带宽开销等于数据的大小。erasure 编码复制算法的读访问带宽等于恢复数据所需的编码块的大小，即 b 块编码数据，等于原始数据的大小，因此 erasure 编码复制算法的读访问带宽等于传统复制算法的读访问带宽。在副本的写访问当中，传统复制算法的写访问带宽等于数据的大小与副本数的乘积，erasure 编码复制算法的写访问开销等于编码后的数据的大小。如果要达到相同的数据可用性，则传统复制算法所需的副本数目要求大于 erasure 编码率，因此传统复制算法写访问的带宽开销大于 erasure 编码复制算法的写访问带宽开销。

显然，在相同存储开销和带宽开销的情况下，erasure 编码可以比数据全复制获得更高的可用性。然而 erasure 编码复制算法的缺陷是：每个 erasure 编码块都可能不同，而在传统复制算法中每个不同副本中的同一个数据块都是一样的。

2 基于 LDPC 编码的数据复制算法

为了提高数据的恢复速度和位置信息的安全性，Dyre 采用动态的数据放置方法，即通过 Hash 函数决定数据块的存储位置，如果有 n 块数据，则数据的存储位置由 $h(key,r)(0 \leqslant r < n)$ 决定，r 为数据块的索引，数据块被存储到节点 ID 值离 $h(key,r)$ 值最近的节点上。存储 $h(key,0)$ 数据块的节点称为该数据的拥有者。当节点请求恢复数据时，可以根据节点的 key 值直接找到数据的存储位置，并且可以同时从 m（m 为数据分割成的块数）个节点下载数据，这种存储方法极大地提高了数据的恢复速度。所有存储这些数据块的节点称为这个数据的副本群，副本群的大小应大于 m。副本群的大小保证了数据能够被恢复。

2.1 数据的迁移

ID 值离 $h(key,r)$ 最近的节点称为该数据块的中心节点，为了保证存储在中心节点上的数据不会因为节点的失效而丢失，可以通过牺牲存储空间来换取数据的安全，即在节点 ID 值分别大于和小于中心节点 ID 值的最近的两个节点上保留中心节点所有数据的一个副本，因此每个数据块都有 3 个副本。这在当今用户的大部分存储区都为空闲的情况下是可行的。但是中心节点相邻的两个节点不算副本群中的节点，因此增加存放数据的节点不影响副本群的改变。

中心节点的任何数据都可以在相邻的节点上找到，当中心节点失效时，可以从相邻节点恢复该中心节点的数据。当新节点加入时，新节点的 ID 值可能比中心节点的 ID 更靠近 $h(key,r)$，因此新节点成为新的中心节点，或者新节点更靠近中心节点，在这种情况下，节点间应产生数据迁移，同时其中一个节点应释放某些数据的副本。

假设 F 为新节点，B 和 C 分别为 ID 值大于和小于 A 的 ID 值的节点。我们以 $A < F < B$ 为例来说明节点间的数据迁移。把节点 ID 值大于节点 A 的最近的节点称为节点 A 的后继节点，把节点 ID 值大于节点 A 的最近的节点称为该节点的后继节点。数据迁移算法的描述如图 1 所示。

由于数据迁移发生在节点 ID 值相邻的节点间，如果节点 ID 值相邻的节点也应为物理相邻的节点，则会大量减少数据迁移的时间等开销。有些 P2P 网络中节点 ID 空间大于 1 维，为了得到节点的前驱和后继节点，且节点与前驱和后继节点间互为后继和前驱节点，可利用 space-filling curves 把节点多维的 ID 空间映射为一维[4]。

```
R_A:节点 A 上的数据 ID 值;
D_I:ID 值为 I 的数据;
S:数据记录集
1. S = R_A ∩ R_B;
2. for(r∈S; S!=∅; S = S - r, r∈S) {
3. D_r 和 r 复制到 F 中;
4. if(|D_r - A| < |D_r - F|)
5. 删除 B 中的 D_r 和 r;
6. else if(|D_r - B| < |D_r - F|)
7. 删除 A 中的 D_r 和 r;
8. else{
9. if(|D_r - A| < |D_r - B|)
10. 删除 A 的前驱节点上的 D_r 和 r;
11. if(|D_r - A| > |D_r - B|)
12. 删除 B 的后继节点上的 D_r 和 r;}}
```

图 1 数据迁移过程

Fig.1 Process of data translation

2.2 数据的恢复

为了保证节点失效时,节点上的数据仍然可用,每个节点都和邻居节点周期性地交换信息,信息包括该节点的前驱和后继节点的地址和 ID 值以及该节点所在的副本群等信息,当一个节点通知邻居节点或者邻居节点检测到该节点失效时,邻居节点应启用恢复算法恢复该失效节点中的数据。

```
B:为节点 A 的后继节点;
1. S = R_B ∩ R_C;
2. for(r∈S; S!=∅; S = S - r, r∈S) {
3. if(|D_r - B| < |D_r - C|)
4. 复制 D_r 和 r 到 B 的后继节点上;
5. B 通知成为 D_r 副本群的一员; }
```

图 2 数据恢复过程

Fig.2 Process of renewing data

每个失效节点的数据都在邻居节点上保留副本,因此失效节点的数据可以直接从邻居节点上找到。节点失效时由该节点的前驱和后继节点启动恢复算法恢复该节点中的数据。例如节点 A 的前驱和后继节点为 B 和 C,当 B 或 C 检测到或者被通知 A 节点失效时,以 B 节点为例,数据恢复过程如图 2 所示。

2.3 数据的重建

由于采用(n,m)的 erasure 编码时,至少需要 m 块编码才能恢复原始数据,因此系统中应保证至少 m 块编码有效,即数据副本群的有效节点的数量应大于 m,系统中引入一个阈值 $f(f>m)$,当副本群的大小小于 f 时,该群启动重建算法重建群中的数据。

在副本群中需要一个节点监视群的大小,以便群的大小小于 f 时能够启动数据重建算法,监视群的大小的任务由副本的拥有者即存储 $h(key,0)$编码块的节点承担。群中每个节点周期性的发送消息给副本拥有者,副本拥有者根据确认消息计算群的大小。副本拥有者节点失效时,存储 $h(key,r+1)$(r 为拥有者存储的编码块索引号)编码块的节点称为副本拥有者。

在数据重建的过程中,群中的某些节点需要存储的新编码块与原来的编码块不一样,当用户需要访问编码块时,可能从一部分节点上提取了旧的编码块而从另一部分节点上提取了新的编码块,这样引起了数据还原错误。因此在编码块中增加版本信息位(取值为 0 或者 1),当用户读取的版本信息不一样时,用户丢弃已提取的编码块,然后经过一段随机时间后重新产生访问数据的请求。数据重建后,群中每个节点还应更新前驱和后继节点中的编码块副本。数据重建算法如图 3 所示。

```
R_A:节点 A 上的数据 ID 值;
D_I:ID 值为 I 的数据;
S:数据记录集;
1. for(i = 0; i < n; i++) {  //广播消息
2.  if(N_i 有效){
3.  VD_i = (VD_i + 1) mod 2;
4.  VD_0 = VD_i;
5.  S = S + D_i;}}
6. 利用 S 还原数据 D;
7. 编码 D 为 D_1, ..., D_n;
8. for(i = 0; i < n; i++) {
9. h(key, i)节点存储 D_i 且 VD_i = VD_0;
10. 前驱与后继节点更新数据 D_i; }
```

图 3 数据重建算法

Fig.3 Process of restoring data

3 性能分析

通过模拟实验来分析比较算法 Dyre 与全复制和块复制算法的性能，数据复制算法基于 CAN 上，网络拓扑由 BRITE 产生。测试平台为 P4 微机(CPU 为 P4 2.2GHz，内存为 512MB)。实验主要验证不同的算法中数据在不同的环境下的有效性以及复制算法的时间开销。

1.数据可用性的比较

在第一实验中 u 为 0.5，第二个试验中 S 为 2。所有实验中数据编码都为(2,4)编码，节点数量为 1000，分块复制算法中数据分块的数目为 2 块，数据数量为 1500 个，数据分布请求随机分布。

第一个实验中数据可用性分布如图 4 所示，实验中 Dyre 算法中的数据有效性为 0.99 以上，第二个实验中数据可用性如图 5 所示。由图 4 和图 5 可知(4,2)编码的 Dyre 算法中数据的可用性大于拥有 6 个副本的全复制和块复制算法中数据的可用性，这是由于同时采用了数据迁移和数据重建的算法克服节点失效对数据产生的影响，在相同情况下 Dyre 比 Oceanstore 中的复制算法获得更高的数据可用性，这是由于数据重建算法克服了节点失效对数据还原带来的影响，算法在大量节点失效前还原和重新分配数据，提高了数据的可用性。在(n,m)编码的情况下，Dyre 算法的空间开销为数据大小的 $3(n/m)$倍。

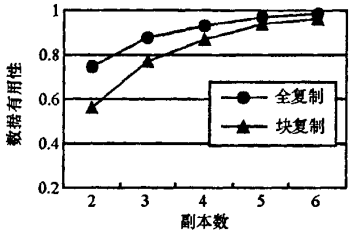

图 4 节点有效性相同的数据可用性比较

Fig.4 Comparison of data availability with same node reliability

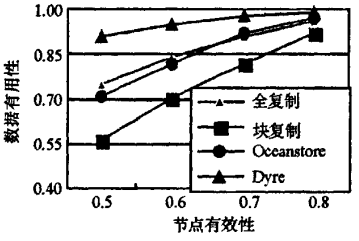

图 5 复制算法数据可用性比较

Fig.5 Comparison of data availability

2.时间开销的测试

在本实验中节点数和数据量为 100，每次实验中请求查找的数据的大小不一样，数据块的大小为 1MB 和 2MB，比例为 8:2，节点间的网络带宽设置为 10MB/s，S 为 2。用 Tc 表示数据查找时间，Ts 表示传输一个数据块的时间，Td 表示数据恢复时间。

图 6 所示为各种数据复制算法中每个数据访问时间开销的比较。从图中可以看出，块复制算法具有最小的时间开销，数据访问的开销为 $Tc + \mathrm{Max}(Ts)$。Dyre 和 Oceanstore 中数据访问所需的时间为 $Tc + \mathrm{Max}(Ts) + Td$，全复制算法中数据访问的时间开销为 $Tc + b \times Ts$。虽然 Dyre 比全复制算法要增加数据解码的时间，但 Dyre 采用 LDPC 编码，解码开销很小，Dyre 中各个数据块可以并行传输，而全复制算法中数据传输时间等于整个数据的传输时间。因此当额外数据传输时间大于数据解码时间时，总的数据访问时间是 Dyre 比全复制算法小。由于 Dyre 采用动态数据数据放置算法和编码，比 Oceanstore 中数据的查找时间和数据解码时间都小，因此 Dyre 比 Oceanstore 的数据访问时间小。

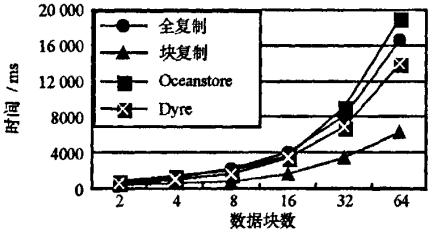

图 6 复制算法开销比较

Fig.6 Comparison of replication time

(下转第 70 页)

参考文献:

[1] Vapnik V N. The Nature of Statistical Learning Theory[M]. New York: Springer, 1995.

[2] Zhang L, Zhou W D, Jiao L C. Radar Target Recognition Based on Support Vector Machine[C]//Proceedings of the 2000 International Conference on Signal Processing, 2000, 3: 1453 - 1456.

[3] Li Y, Ren Y, Shan X M. Radar HRRP Classification with Support Vector Machines[C]//Proceedings of the 2001 International Conference on Info-tech and Info-net, 2001, 1: 218 - 222.

[4] Wang X D, Wang J Q. Support Vector Machine for HRRP Classification[C]//Proceedings of the 7th International Symposium on Signal Processing and Its Applications, 2003, 1: 337 - 340.

[5] Syed N A, Liu H, Sung K K. Incremental Learning with Support Vector Machines[C]//Proceedings of Workshop on Support Vector Machines at the International Joint Conference on Artificial Intelligence, Sweden: Stockholm, 1999: 272 - 276.

[6] 萧嵘,王继成,孙正兴,等. 一种SVM增量学习算法[J].南京大学学报(自然科学版),2002,38(2):152 - 157.

[7] 曾文华,马健.一种新的支持向量机增量学习算法[J].厦门大学学报(自然科学版),2002,41(6):687 - 691.

[8] An J L, Wang Z G, Ma Z P. An Incremental Learning Algorithm for Support Vector Machine[C]//Proceedings of the 2th International Conference on Machine Learning and Cybernetics, Xi'an, 2003(2): 1153 - 1156.

[9] Li Z W, Zhang J P, Yang J. A Heuristic Algorithm to Incremental Support Vector Machine Learning[C]//Proceedings of the 3th International Conference on Machine Learning and Cybernetics, Shanghai, 2004(3): 1764 - 1767.

[10] Mitra P, Murthy C A, Pal S K. Data Condensation in Large Databases by Incremental Learning with Support Vector Machines[C]//Proceedings of the 15th International Conference on Pattern Recognition, Spain, 2000(2): 708 - 711.

[11] Xiao R, Wang J C, Zhang F Y. An Approach to Incremental SVM Learning Algorithm[C]//Proceedings of the 12th International Conference on Tools with Artificial Intelligence, 2000(1): 268 - 273.

[12] 萧嵘,王继成,孙正兴,等. 一种SVM增量学习算法——ISVM[J].软件学报,2001,12(12):1818 - 1824.

[13] Cauwenberghs G, Poggio T. Incremental and Decremental Support Vector Machine Learning[C]//Advances in Neural Information Processing Systems, Cambridge MA: MIT Press, 2001(13): 409 - 415.

[14] Burges C J C. A Tutorial on Support Vector Machines for Pattern Recognition[M]. Boston: Kluwer Academic Publishers, 1998.

[15] 周伟达,张莉,焦李成. 支撑矢量机推广能力分析[J].电子学报,2001,29(5):590 - 594.

[16] Zhang L, Zhou W D, Jiao L C. Pre-extracting Support Vectors for Support Vector Machine[C]//Proceedings of the 5th International Conference on Signal Processing, 2000(3): 1432 - 1435.

[17] Ding A L, Liu F, Li Y. Pre-extracting Support Vector by Adaptive Projective Algorithm[C]//Proceedings of the 6th International Conference on Signal Processing, 2002(1): 21 - 24.

(上接第64页)

4 结论

本文提出了一个基于erasure的数据复制算法Dyre,编码块采用动态分配算法存储到不同的节点中,在每个节点的前驱和后继节点中保存编码块副本,利用数据迁移和恢复算法保证数据在节点失效情况下可用,采用重建算法保证编码数据块的数量大于数据还原所需的块数。模拟实验表明该算法能够获得很高数据的可用性,在相同的环境当中获得的数据可用性高于全复制和块复制算法。

参考文献:

[1] MacWilliams F J, Sloane N J A. The Theory of Error-correcting Codes, Part I[M]. North-Holland Publishing Company, Amsterdam, New York, Oxford, 1977.

[2] Kubiatowicz J, et al. Oceanstore: An Architecture for Global-scale Persistent Storage[C]//Proceedings of ASPLOS 2000, Cambridge, Massachusetts, Nov. 2000.

[3] Brunskill E. Building Peer-to-peer Systems with Chord, a Distributed Lookup Service[C]//HOTOS'01: Proceedings of the Eighth Workshop on Hot Topics in Operating Systems, 2001.

[4] Xu Z C, Mahalingam M, Karlsson M. Turning Heterogeneity into an Advantage in Overlay Routing[R]. Infocom'03, 2003.