文章编号: 1001- 2486(2008) 06- 0057- 06

基于 Erasure Code 的分割文件 P2P 存储结构设计*

陶 钧, 沙基昌, 王 晖
(国防科技大学 信息系统与管理学院, 湖南 长沙 410073)

摘 要: 在充分考虑 P2P 存储特点的前提下, 设计了一种新颖的基于 Erasure Code 的分割文件 P2P 存储方案, 构建 Chord 协议上的 P2P 文件存储结构。通过设计有效的文件存储、访问和更新功能, 克服传统 Erasure Code 冗余文件存储方案的不足, 使存储文件具有一定的数据修改能力和结构伸缩能力。理论分析和实验仿真验证了设计方案在 P2P 存储系统中的可行性和有效性。

关键词: P2P 存储; erasure code; 文件分割; 存储策略
中图分类号: TP393 文献标识码: A

A Design for the Erasure Code Based Segmented File P2P Storage Structure

TAO Jun, SHA Ji-chang, WANG Hui
(College of Information System and Management, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: In view of the characteristics of P2P storage, a novel Erasure Code based segmented file P2P storage strategy is designed, and the Chord protocol based P2P storage structure is constructed. The effective file storage, access and update function of the novel strategy overcome the deficiencies of the traditional Erasure Code P2P storage strategy, enabling the stored file to have a certain data revision capability and structure expansion-contraction capability. Theoretical analysis and experimental simulation verify that the novel strategy has reasonable feasibility and effectiveness.

Key words: P2P storage; erasure code; file segmentation; storage strategy

随着网络技术的发展, P2P 文件存储系统已成为解决海量数据存储的重要技术途径。它利用对等服务概念组织整个存储体系, 使结点规模的扩张突破传统瓶颈限制, 从而满足日益增长的各种数据存储需求。目前 P2P 文件存储系统主要通过数据冗余存储结构来解决系统规模引起的可靠性问题, 避免因单点失效而造成的文件数据丢失。但是完整复制的冗余模式会带来很大的存储资源开销, 许多著名的 P2P 文件存储系统, 例如 OceanStore[1]、Total recall[2]、Myriad[3] 等, 都采用了基于 Erasure Code[4-5] 的冗余存储方案, 利用具有纠删性质的文件编码块, 实现存储系统的高可靠、可扩展性。

1 相关工作分析

基于 Erasure Code 的冗余存储方案是提高 P2P 文件存储系统稳固性的实用技术, 著名的 P2P 文件存储系统 OceanStore[1]、Total recall[2]、Myriad[3] 等都采用了该冗余存储方案。Erasure Code 技术产生于磁盘阵列(RAID) 的数据存储研究领域[9], 通过将存储文件均分为 m 块并编码为 $n(n > m)$ 个编码块, 形成冗余度为 $\gamma = n/m$ 的文件存储模式, 使得任意 $r(r \geqslant m)$ 个编码块就可以解码重构出存储文件; 特别是符合 MDS 性质的 Erasure Code 方案(如 Cauchy Reed-Solomon Code) 满足所需重构编码块数量 $r = m$。但是在实际 P2P 文件存储系统中, 基于 Erasure Code 的文件冗余存储方案仍然存在着不足:

(1) 文件更新开销大。由于存储文件进行了 Erasure Code 编码, 因此文件数据的任何改动(特别是文件长度的变化) 都会影响到几乎所有的文件编码块。文献[1] 认为 Erasure Code 仅适合于文件归档存储,

* 收稿日期: 2007- 07- 12
基金项目: 国家部委基金资助项目
作者简介: 陶钧(1979—), 男, 博士生。

文件更新则通过版本更新的方式进行, 重新生成并存储所有的新版本文件编码块。

(2) 存储适应性不强。Erasure Code 是一类线性代数型编码, 其编解码开销符合如下关系式:

$$\mathrm{TIME}_{\mathrm{encoding}} = O(\gamma m S_F), \quad \mathrm{TIME}_{\mathrm{decoding}} = O(mS_F) \tag{1}$$

其中 S_F 表示存储文件的大小。为了提高编解码过程的执行效率, 通常选择较小的文件分块数量 m。因此在存储大型文件时, 单个文件编码块的尺寸仍然偏大。

(3) 访问性能受限制。P2P 存储文件的访问效率依赖于存储结点所提供的上行传输服务, 为获得较高的文件访问速率, 下载端必须以并行方式从各个结点同时下载数据。由于 Erasure Code 的文件分块数量小, 因此只有在高冗余度下才能满足大规模并行访问需求。

2 文件存储结构的方案设计

针对上述传统 Erasure Code 的 P2P 文件存储方案分析, 可以看出 Erasure Code 在提高存储可靠性的同时也会影响系统的综合性能。文件分割是并行处理领域的重要技术手段, 文献[6] 中所设计的 CFS 文件存储系统就是以文件分割段为单位实现大规模的网络存储, 但是文件分割存储会造成完整数据的访问可靠性降低; 文献[7] 则设计了文件分割与 Erasure Code 编码相结合的 Reperasure 数据复制方法, 但直接将文件存储在 DHT 上的方式无法适应动态 P2P 环境。本文结合文献[6-7] 提出了一种改进的 Erasure Code 分割文件 P2P 存储方案, 如图 1 所示, 其存储文件的构造方法为:

步骤 1 根据存储文件 F 的三元组< 文件名称、类型、所有者> 进行哈希运算, 得到全局唯一的文件标识 FileHash-ID; 生成名为 FileHash-ID 的元文件 F_{Meta} 记录文件的 FileName、Type、Owner 信息。

步骤 2 对存储文件 F 进行文件分割, 文件分割段的大小为 S_{Seg}, 对不能完全分割的情况允许最后分割段大于 S_{Seg}。记分割段为 $F_1, F_2, \cdots, F_L$, 分割段都有 Segment-ID 序号。在元文件 F_{Meta} 中记录文件分割段的版本(SegVersion-ID)、后继分割块(SegNext-ID)、一致性检验的内容哈希值(SegHash-ID)。

步骤 3 将分割段 F_i 均分为 m 个数据块, 记作 $F_{i,1}, F_{i,2}, \cdots, F_{i,m}$, 对其进行 (m, n) 模式的 Erasure Code 编码, 形成 n 个具有 MDS 性质的文件编码块, 记作 $S_{i,1}, S_{i,2}, \cdots, S_{i,n}$。每个编码块按 Segment-ID 编号, 并附带文件标识 FileHash-ID 和所属分割段的 SegVersion-ID、SegNext-ID 、SegHash-ID。

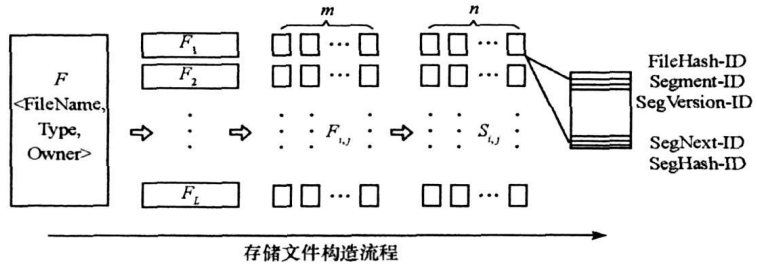

图 1 存储文件的构造流程图

Fig. 1 The flowchart of the file storage scheme

改进的 Erasure Code 分割文件存储方案是一种混合模式的文件存储结构, 相当于采用 Erasure Code 编码来存储一组串联关系(Tandem Relation) 的小型文件。这种改进方案在不增加文件 Erasure Code 编码开销的前提下, 加大了存储文件的数据离散化程度, 并且保证解码重构后的独立自检验能力。

3 文件存储结构的功能体系

3.1 文件数据的存储定位机制

改进后的 Erasure Code 分割文件编码块集合为 $\Omega = \{S_{i,j} \mid L \geqslant i \geqslant 1,\ n \geqslant j \geqslant 1\}$, 底层采用基于 Chord[9] 的结构化 P2P 网络, 每个结点都有唯一的编号(Node-ID), 可形成文件 FileHash-ID 到 Chord 空间上的存储映射。以标识 FileHash-ID 为关键字定位到结点 A, 从结点 A 开始连续定位 $K-1$ 个后继结点[9], 将编码块集合 Ω 存储到选定的 K 个结点上, 前 $n/2$ 个结点同时保存元文件 F_{Meta}(如图 2 所示)。

编码块集合 Ω 在结点上采用不同的分布规则, 从而实现不同的文件存储策略。

策略 1　紧密型存储(Strategy-C), 将编码块集合 Ω 存储到 $K = n$ 个结点。从初始结点 A 开始, 第 $i-1$ 个后继结点依次存放 L 个编码块的集合 $\Omega_i = \{S_{1,i}, S_{2,i}, \cdots, S_{L,i}\}$。

策略 2　分散型存储(Strategy-D), 将编码块集合 Ω 存储到 $K = n \times L$ 个结点。从初始结点 A 开始, 后继结点存放分割段 F_1 至 F_L 的编码块, 顺序为 $G = \{\Omega_1 \rightarrow \Omega_2 \rightarrow \cdots \rightarrow \Omega_L\} = \{S_{1,1} \rightarrow \cdots \rightarrow S_{L,1} \rightarrow \cdots \rightarrow S_{1,n} \rightarrow \cdots \rightarrow S_{L,n}\}$。

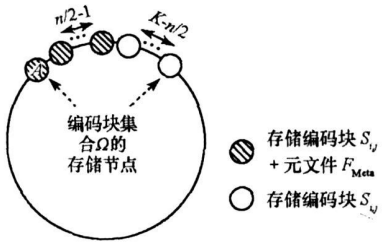

图 2　编码块存储结点选择方式
Fig. 2　The decision for storage nodes of the coded chunks

策略 3　随机型存储(Strategy-R), 将编码块集合 Ω 存储到 $n < K < n \times L$ 个结点。编码块分布规则介于存储策略 Strategy-D 与存储策略 Strategy-C 之间, 允许编码块非均匀地存放到各结点上。

比较分析: 存储策略 Strategy-C 与传统 Erasure Code 具有相同的逻辑存储结构, 但是数据的存储颗粒度更小; 存储策略 Strategy-D 与传统 Erasure Code 相比逻辑存储结构更加分散, 达到最大限度的分散存储; 存储策略 Strategy-R 与传统 Erasure Code 相比逻辑存储结构灵活, 能够适合于异构化的 P2P 文件存储环境。表 1 所示的伪代码实现了分割化 Erasure Code 文件存储方案下的存储结构伸缩性。

表 1　文件存储定位算法的伪代码描述

Tab. 1　The parallel download reconstruction method for coded chunks

```
[Parameter]
FileHash-ID: 文件标识; L: 文件分割段数; n: 文件分割段内的 Erasure Code 参数;
Ω: 存储编码块集合; S: 文件存储策略
[Algorithm]
StoreFile(FileHash-ID, L, n, Ω, S)
  Node= null
  if S= Strategy-C then
    for t = 1 to n do
      if Node= null then
        Node= DHTLookUp(FileHash-ID)
      else
        Node= Node.successor
      Node --store--> Ω_t= {S_{1,t}, S_{2,t}, ···, S_{L,t}}
      if t ≤ n/2 then Node --store--> F_Meta
  else if S= Strategy-D then
    for t= 1 to n×L do
      if Node= null then
        Node= DHTLookUp(FileHash-ID)
      else
        Node= Node.successor
      Node --store--> G_t= {S_{i,j} | t= j×L+ i}
      if t ≤ n/2 then Node --store--> F_Meta
  else if S= Strategy-R then
    t= k= r= 0
    while t< n×L do
      if Node= null then
        Node= DHTLookUp(FileHash-ID)
      else
        Node= Node.successor
      r= Random(1, L)
      if t+ r> n×L then r= n×L- t
      Node --store--> {G_{t+1}, G_{t+2}, ···, G_{t+Num}}
      t= t+ r  k= k+ 1
      if k ≤ n/2 then Node --store--> F_Meta
    loop
```

3.2　文件数据的并行访问方式

改进后的 Erasure Code 分割文件集合 Ω 可以实现编码数据的联合下载重构。当保证每个文件分割段至少有 m 个可访问编码块时, 就可以完成文件数据的解码重构。编码块采用条带传输(Stripe-trans)

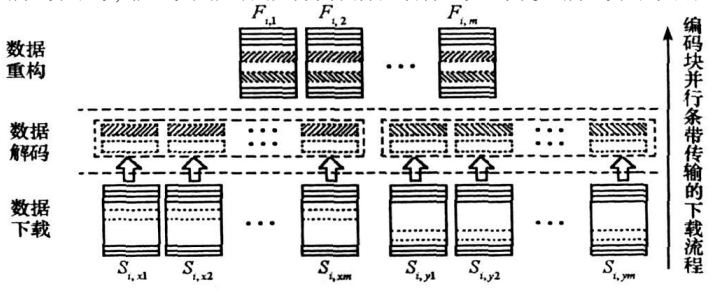

图 3　文件编码块的联合并行重构方式

Fig. 3　The parallel reconstruction for coded chunks

技术,对分割段 F_i 实现多编码块的联合访问,图3显示了编码块的联合并行下载重构方式。在数据下载前应检查各编码块的 Segment-ID、SegVersion-ID、SegHash-ID 是否相同,在解码重构后应检验分割段 F_i 的内容哈希值是否与 SegHash-ID 相一致,从而保证下载文件数据的正确性和完整性。

比较分析:与传统 Erasure Code 和非编码数据的传输不同,分割化 Erasure Code 编码块的条带传输是同时作用于 m 个重构块 $F_{i,1}$, $F_{i,2}$, …, $F_{i,m}$ 上的,条带传输单元都保留所属分割段的编码系数,具有独立解码能力。分割化 Erasure Code 的数据重构充分利用分散存储的规模优势,获得比传统 Erasure Code 更高的数据访问性能。

3.3 文件数据的分段更新流程

改进后的 Erasure Code 分割文件集合 Ω 的数据更新可以划分到一个或多个文件分割段 F_i 上。记所需修改的文件分割段集合为 $\Delta\{F_i\}$,文件更新过程中始终保持有 $\Delta\{F_i\}$ 的旧/新版本的完整编码块,使文件 F 始终保持数据的完整性[10]。那么文件更新的具体流程为:

步骤1 计算集合 $\Delta\{F_i\}$ 更新后的内容哈希值 $\mathrm{SegHash}_{new}$,提升 $\Delta\{F_i\}$ 的版本编号 $\mathrm{SegVersion\text{-}ID}_{new} = \mathrm{SegVersion\text{-}ID}_{old} + 1$,并重新生成 $\Delta\{F_i\}$ 的 (m, n) 模式 Erasure Code 编码块集合 $\Delta\{\Omega_i\}_{new}$。

步骤2 将 $\Delta\{\Omega_i\}_{new}$ 保存到相应的 $\Delta\{\Omega_i\}_{old}$ 位置,保存过程中保留 $\Delta\{\Omega_i\}_{old}$ 数据。在集合 $\Delta\{\Omega_i\}_{new}$ 完全存储后,修改存储元文件 F_{Meta} 中与 $\Delta\{F_i\}$ 相关的 $\mathrm{SegVersion\text{-}ID}_{new}$ 和 $\mathrm{SegHash}_{new}$ 信息。

步骤3 文件更新完成后删除 $\Delta\{F_i\}$ 所对应的 $\Delta\{\Omega_i\}_{old}$ 数据,释放结点上的存储空间。

比较分析:分割化 Erasure Code 对于小范围的文件修改来说,仅需更新部分文件分割段所对应的编码块,比传统 Erasure Code 的数据更新量显著降低,提高了存储文件的更新效率和可修改能力。

4 文件存储结构的综合性能

4.1 文件数据的存储可靠性分析

P2P 存储系统的结点会因故障造成存储数据失效,假设各结点故障概率均为 P_f 且互相独立,可以将完整副本、传统 Erasure Code 和分割化 Erasure Code(存储策略 Strategy-C、Strategy-D 和 Strategy-R)的存储可靠度进行比较,分别记作 R_1、R_2、$R_{3\text{-}C}$、$R_{3\text{-}D}$ 和 $R_{3\text{-}R}$,则当冗余度为 γ 时有

$$\begin{cases} R_1 = 1-(1-P_f)^{\gamma}, \quad R_2 = R_{3\text{-}C} = \displaystyle\sum_{i=0}^{(\gamma-1)m}\binom{\gamma m}{i} P_f^{\gamma m-i}(1-P_f)^{i} \\ R_{3\text{-}D} = \left[\displaystyle\sum_{i=0}^{(\gamma-1)m}\binom{\gamma m}{i} P_f^{\gamma m-i}(1-P_f)^{i}\right]^{L}, \quad R_{3\text{-}C} \leqslant R_{3\text{-}R} \leqslant R_{3\text{-}D} \end{cases} \tag{2}$$

图4显示了 $P_f = 0.1$ 和 $P_f = 0.5$ 时存储可靠度随存储冗余度的变化情况,其中 Erasure Code 编码的文件分块数 $m = 8$、分割段数 $L = 8$。从变化曲线的对比可以看出:分割化 Erasure Code 与完整副本冗余相比具有更好的可靠性效果;与传统 Erasure Code 相比则具有一定伸缩性,其中存储策略 Strategy-C 的可

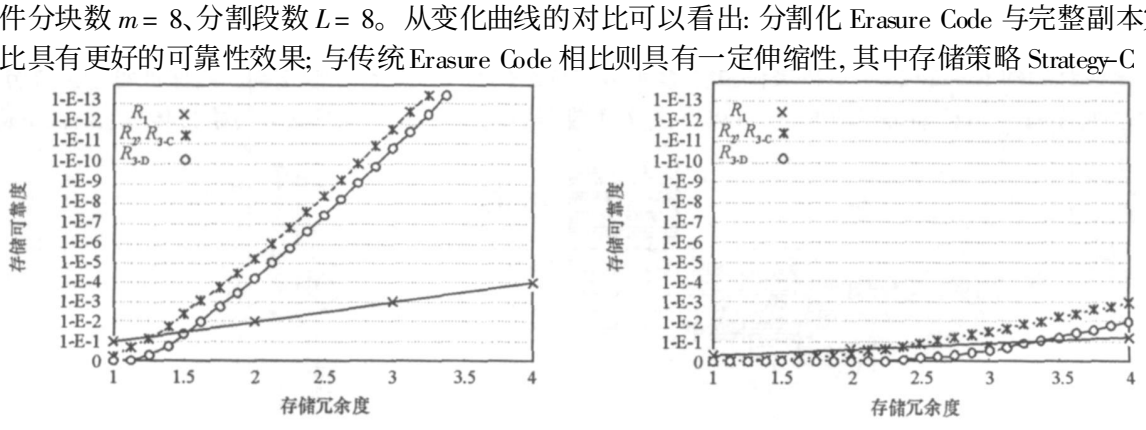

图4 存储文件可靠度变化曲线

Fig. 4 The reliability curve of the storage file

靠性最高、Strategy-R 次之、Strategy-D 最差, 但都与传统 Erasure Code 的可靠性相同或者接近。

4.2　文件数据的操作复杂性比较

文件数据执行读/写操作的开销则是衡量系统复杂程度的重要标准。在 P2P 文件存储系统中文件数据的操作可分为小数据量读/写(Small Read/Write)和大数据量读/写(Large Read/Write)。定义小数据量读/写操作的数据量为 $\alpha \leqslant S_{Seg}$, 大数据量读/写操作的数据量为 $\beta \approx S_F$。从表 2 可以看出, 分割化 Erasure Code 与传统 Erasure Code 相比在数据吞吐量方面具有优势, 在结点连接量方面则具有伸缩性。

表 2　不同文件存储方案的数据吞吐量和结点连接量

Tab. 2　The data throughput and nodes connecting quantity of different file storage scheme

文件存储方案	数据吞吐量				结点连接量			
	Small Read	Small Write	Large Read	Large Write	Small Read	Small Write	Large Read	Large Write
完整副本	α	$\gamma\alpha$	β	$\gamma\beta$	$1 \sim \gamma$	γ	$1 \sim \gamma$	γ
传统 Erasure Code	$\min(m\alpha, S_F)$	γS_F	S_F	γS_F	$m \sim n$	n	$m \sim n$	n
分割化 Erasure Code (Strategy-C)	$\min(m\alpha, S_{seg})$	γS_{seg}	S_F	γS_F	$m \sim n$	n	$m \sim n$	n
分割化 Erasure Code (Strategy-D)	$\min(m\alpha, S_{seg})$	γS_{seg}	S_F	γS_F	$m \sim n$	n	$mL \sim nL$	nL

4.3　文件数据的访问效率仿真实验

文件数据的访问效率是文件存储系统的重要设计指标。在 P2P 文件存储系统中, 结点会表现出不稳定性(结点故障)和动态特性(退出/加入存储系统)。假设结点故障修复时间远大于文件访问任务时间, 且结点在线/离线时间是马尔科夫过程, 那么对文件数据的访问效率进行仿真: 文件 F 存储在结点集合 N 内, 单位仿真时间为 Δt, 点对点传输带宽为 $d/\Delta t$, 网络瓶颈带宽为 $k/\Delta t$, 仿真开始前结点以 P_f 的概率发生故障, 正常结点在仿真步长下按照 P_{down} 和 P_{up} 的概率发生在线/离线状态转移。考察完整副本冗余、传统 Erasure Code(与分割化 Erasure Code 的 Strategy-C 相同)和分割化 Erasure Code 的 Strategy-D 下的文件访问效率。取冗余度 $\gamma = 4$, Erasure Code 编码分块数 $m = 8$, 文件分割段数 $L = 8$。

场景 $S1$: 取仿真参数 $P_f = 0.1$、$P_{down} = 0.01$、$P_{up} = 0.05$、$|F| = 1000d$、$k = 100d$;

场景 $S2$: 取仿真参数 $P_f = 0.1$、$P_{down} = 0.05$、$P_{up} = 0.05$、$|F| = 2000d$、$k = 100d$;

场景 $S3$: 取仿真参数 $P_f = 0.5$、$P_{down} = 0.01$、$P_{up} = 0.05$、$|F| = 1000d$、$k = 20d$;

场景 $S4$: 取仿真参数 $P_f = 0.5$、$P_{down} = 0.05$、$P_{up} = 0.05$、$|F| = 1000d$、$k = 100d$。

图 5 是仿真 2000 次的实验结果。图 5(a)是完整副本冗余, 其文件访问时间长, 访问失败的次数多; 图 5(b)是传统 Erasure Code(与存储策略 Strategy-C 相同), 其文件访问时间较小, 访问任务完成性高; 图 5(c)是存储策略 Strategy-D, 其文件访问时间最少, 但易受网络瓶颈限制。

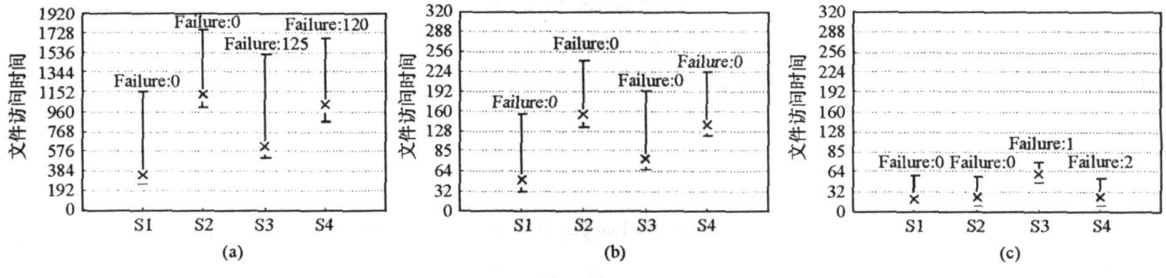

图 5　文件访问的时间范围

Fig. 5　The scope of the file access time

5　总 结

本文提出了一种改进的基于 Erasure Code 的分割文件 P2P 存储方案, 并设计了相关的文件存储、访问和更新功能, 针对所设计的不同文件存储策略, 通过分析和仿真实验讨论了它们较完整副本冗余和传

统 Erasure Code 方案的优势。今后的工作重点将会是针对目前网络结点的差异性,设计具有自适应修复能力的异构网络 P2P 文件存储策略。

参 考 文 献:

[1] Kubiatowicz J, Bindel D, et al. Oceanstore: An Architecture for Global-scale Persistent Storage[C]//Proceedings of ASPLOS, 2000.
[2] Ranjita B, Kiran T, et al. Total Recall: System Support for Automated Availability Management [C]//NSDI, 2004.
[3] Chang F, Minwen Ji, et al. Myriad: Cost-effective Disaster Tolerance[C]//Proceedings of FAST, 2002.
[4] Luby M, Mitzenmacher M. Practical Loss-resilient Codes [C]//ACM Symposium on Theory of Computing, 1997.
[5] http://www.icsi.berkeley.edu/~luby/erasure.html[R].
[6] Frank D, Frans M, et al. Wide-area Cooperative Storage with CFS[C]//Proceedings of ACM SOSP, 2001.
[7] Zhang Z, Lian Q. Reperasure: Replication Protocol using Erasure-code in Peer-to-peer Storage Network[C]//SRDS, 2002.
[8] Stoica I, Morris R, et al. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications [C]//Proc. ACM SIGCOMM, San Diego, 2001.
[9] Chen P, Lee E K, et al. RAID: High-performance, Reliable Secondary Storage[C]//ACM Computing Surveys, 1994.
[10] Aguilera M K, Janakiraman R, Xu L. On the Erasure Recoverability of MDS Codes under Concurrent Updates[C]//ISIT Proceedings, 2005.

(上接第 42 页)

真空压制、热压制和多次压制对材料的密度影响不大,而压制压力对材料密度影响较明显,密度随压力增大呈先增加后保持稳定的趋势。烧结温度过高或过低都会导致材料密度降低;保温时间增加时,材料密度变化呈先减小后增加的趋势,拉伸强度则明显降低。欲获得高拉伸强度的 PTFE/Al 反应材料,可采取自然降温的方式冷却,快速降温则将产生高断裂伸长率的样品。

参 考 文 献:

[1] Daniel B N, Richard M T, Nikki R. Low Temperature Extrudable High Density Reactive Materials [P]. US 6962634, 2005.
[2] Michael T R, Daniel W D, James R H, et al. Reactive Material Enhanced Projectiles and Related Methods [P]. US 20060011086, 2006.
[3] William J F. Reactive Fragment Warhead for Enhanced Neutralization of Mortar, Rocket, and Missile Threats[R]. ONR-SBIR, N04-903, 2005.
[4] Joshi V S. Process for Making Polytetrafluoroethylene-Aluminum Composite and Product Made [P]. US 6547993, 2003.
[5] Vavrick D J. Reinforced Reactive Material [P]. US 20050067072, 2005.
[6] 张彤. 含能破片材料的制备及毁伤性研究[D]. 长沙:国防科技大学, 2006.
[7] 黄亨建, 黄辉, 阳世清, 等. 毁伤增强型破片探索研究[J]. 含能材料, 2007, 15(6): 566-569.

(上接第 56 页)

6 结 语

本文针对延缓纠正模式的可靠性增长试验 Bayes 评估进行研究,重点解决了阶段间信息传递的增长因子的确定问题,指出现有的 ML-㊣方法的局限性,提出了改进的 ML-㊣增长因子确定方法。最后的仿真示例说明了改进方法的有效性。

参 考 文 献:

[1] 周源泉. 可靠性增长[M]. 北京: 科学出版社, 1992.
[2] Maurizio G, Gianpaolo P. Automative Reliability Inference Based on Past Data and Technical Knowledge[J]. Reliability Engineering and System Safety, 2002, 76: 129-137.
[3] Raffaela C, Maurizio G. A Reliability-growth Model in a Bayes-decision Framework[J]. IEEE Trans. on Reliability, 1996, 45(3): 505-510.
[4] Robinson D G, Dietrich D. A New Nonparametric Growth Model[J]. IEEE Trans. on Reliability, 1987, 36(10): 411-418.
[5] 王华伟. 液体火箭发动机可靠性增长分析与决策研究[J]. 宇航学报, 2004, 25(6): 655-658.
[6] 党晓玲. 柔性制造系统可靠性增长管理与分析技术研究[D]. 长沙:国防科技大学, 1999: 45-53.
[7] 张金槐. Bayes 可靠性增长分析中验前分布的不同确定方法及其剖析[J]. 质量与可靠性, 2004, 4: 1-10.
[8] Pandey A, Singh A, Zimmer W J. Bayes Reliability Estimation Using Multiple Source of Prior Information: Binomial Sampling[J]. IEEE Trans. on Reliability, 1994, 43(1): 511-520.
[9] 张士峰. Bayes 小子样理论及其在武器系统评估中的运用研究[D]. 长沙:国防科技大学, 2000: 27-36.