容错控制系统的一类新型阵列码技术

孟庆春,刘云卿

(中国科学院 力学研究所 技术发展部,北京 100190)
E-mail: mqc_794388@ hotmail com

摘 要: 随着现代化生产的不断发展,控制系统变得日渐复杂,出现各类故障的可能性也随之增大. 为了解决控制系统复杂化引起系统安全性减低的问题, 人们将编码技术作为一种软件冗余技术应用于控制系统, 由于控制系统具有强实时性的要求, 因此阵列码技术成为主流技术. 阵列码技术虽然具有纠错时间短的优点,但是纠错能力有限,一般只能纠正一到两个磁盘错误. 针对上述不足提出一种能够在三个磁盘同时发生错误的条件下有效恢复数据的新型阵列码-扩展 X 码, 给出编译码算法, 并将其应用于火箭控制系统的容错.

关 键 词: 控制系统;阵列码; X 码;扩展 X 码
中图分类号: TP393. 08 **文献标识码**: A **文 章 编 号**: 1000-1220(2010) 03-0571-06

Class of Triple Error Control Array Code Applied in Control System

MENG Qing-chun, LIU Yun-qing
(Mechanical Institution, Chinese Academy of Sciences, Beijing 100190, China)

Abstract: With the rapid development of modern industrialization, control system becomes more and more complex, and the possibility of occurring fault is steadily increasing. In order to solve the problem that the security of control system gradually downwards with the expansion of the scale, and therefore encoding method as a kind of software redundancy technique has been applied in control system. For the reason that control system has the requirement of real-time, so Array code has become the mainstream. Although Array code has comparatively short correcting time, the fault-tolerant capacity is limited, it can only correct one or two disk errors. This paper which aiming at the deficiency above puts forward a new kind of encoding method - Extension X Code, this method can recover datas in the case that three disks are in error, Also, encoding and decoding algorithms are presented and it can be applied in the control system used by rocket

Key words: control system; array code; X code; extension X code

1 引 言

随着现代化生产的发展和科学技术的进步,控制设备结构日趋复杂、功能不断完善. 由于控制设备经常运行在大功率、高负荷、不间断的工作条件下,因而不可避免的会出现各类故障. 为解决上述问题产生了各类容错技术,基本思想如下[1]:当控制设备在运行过程中一个或多个关键部件发生故障时,通过系统重构、结构冗余、故障自适应等方法,对故障进行补偿、抑制、消除或修复.

在各类容错技术中,冗余技术得到了广泛的应用. 冗余技术主要基于如下思想[1]:多个独立故障同时发生的可能性很小. 冗余技术按其性质划分为四类[2]:硬件冗余、软件冗余、时间冗余和信息冗余. 研究人员在对软件冗余的研究过程中提出了一些有效的设计模式,如基于静态冗余的 N 版编程模式[3]、基于动态冗余的恢复块模式[4]、校验点恢复模式[5]、流程对模式[6]等.

流程对模式虽然具有故障恢复时间短等优点,但是当备用服务器中的检验点信息出现意外损毁时控制系统的安全性无法得到有效保障,为了解决上述问题,研究人员引入编码技术进行信息冗余,基本思想如下:利用 RS 码对备用服务器接收到的检验点信息进行本地编码,并将产生的校验信息存储在备用服务器中,这样即使备用服务器上存放的部分检验点信息出现损坏仍然可以恢复出原始信息.

采用 RS 码虽然在一定程度上改进了控制系统的安全性, 但是由于 RS 码[7]解码算法的时间复杂度较高, 而大多数控制系统的实时性要求很高, 因此难以适应实际应用的需要. 针对上述问题本文利用阵列码技术, 在 X 码的基础上提出了一类能够在备用服务器中三个磁盘损坏的情况下有效恢复数据的新型编码-扩展 X 码, 该类编码译码算法时间复杂度远低于 RS 码, 纠错能力又高于 X 码[8]、EVENODD[9]码、B 码[10]、S 码[11]等经典阵列码, 因此特别适合于实时性强, 校验点信息量不大的控制系统,火箭控制系统就是这样一类典型应用.

2 控制系统软件容错设计模式

软件冗余技术由于代价小、易于实现,因此在各类冗余技

收稿日期: 2008-11-14 收修改稿日期: 2009-01-21 基金项目:科研院所技术开发专项基金项目 (国科计字 [2000]056号)资助. 作者简介:孟庆春,男, 1979年生,博士,助理研究员,研究方向为机动车检测与网络技术;刘云卿,男, 1949年生,研究员,研究方向为机动车检测与网络技术.

术得到了深入研究,研究人员设计了很多有效的设计模式为不同应用服务,目前得到广泛使用的是基于动态冗余的校验点恢复模式和流程对模式,下面略作介绍.

2.1 校验点恢复模式

校验点恢复模式的基本思想就是在容错过程中引入事务,系统在新事务开始之前保存当前状态,并且保证先前的事务以无差错的状态结束,这样就可以得到一种简单的容错方法:即一旦某事务在执行过程中检测到错误,系统重新载入先前保存的状态信息,这些状态信息被称为校验点信息,这样系统又能从校验点所处的稳定状态继续执行下去,并允许在该稳定状态上进行新的事务处理.

上述过程虽然在一定程度上能起到容错的作用,但仍存在单点失效的问题:即保存校验点信息时出现故障.为了克服上述问题研究人员提出了校验点还原的方法:一个事务执行过程中如果某些数据在主服务器上被修改,到达事务结束点时后备存储设备将记录被修改的数据和事务序列号,如果某一时刻检测到主服务器出现故障则启动备用服务器,在启动恢复过程中,后备存储设备将查询具有最高事务序列号的数据集,并根据该数据集存储的数据恢复主服务器中的信息.

2.2 流程对模式

在校验点恢复模式中由于启动或重启服务器需要进行若干处理才能恢复到校验点状态,因此存在故障恢复时间过长的不足,为此研究人员进一步设计了流程对模式.在该模式中一旦主服务器成功的完成整个事务,将传送与新的稳定状态有关的信息至备用服务器,这样备用服务器就可以保存当前完成事务的完整信息,当主服务器准备就绪可供客户机使用时将向备用服务器发送常规的"心跳信息",备用服务器一旦检测到"心跳信息"终止就作为一个新的主服务器接管事务.

流程对模式虽然具有故障恢复时间短和适用故障种类多的优点,但是当存储在备用服务器上的校验点信息丢失或者损毁,备用服务器在主服务器发生故障的条件下无法有效接管事务.为了解决上述问题,文中引入编码技术进行信息冗余以恢复校验点信息.

3 扩展 X码

扩展 X码是一类具有 MDS性质的阵列码,基于对 X码的发展,但扩展了 X码的纠错能力,对于含有 n个磁盘的存储系统,X码等经典阵列码只有当其中一到两个磁盘发生错误时候可以有效恢复数据,扩展 X码在其中三个磁盘发生错误时仍能有效恢复数据,下面从扩展 X码磁盘阵列布局、编译码算法方面进行详细介绍.

3.1 磁盘阵列布局

为了描述扩展 X码磁盘阵列数据布局,假设备用服务器中含有 n个磁盘,编号为 0…n-1,每个磁盘具有 m(m>4)个磁盘块,编号为 0…m-1,且满足条件 n>3m-6,基于扩展 X码的磁盘阵列布局通过一个 m*n的二维阵列表示,记为:$A=[a_{i,j}]_{0\le i<m,0\le j<n}$,其中每个磁盘中的编号为 m-2,m-1的两个磁盘块均为校验单元,且编号为 0…2m-5的磁盘中还包含一个额外的校验单元,该校验单元记为:$b_{i,j}$($0\le i\le m-3, 0\le j\le 2m-5$),当 $0\le j\le m-3$时,$i=j$;当 $m-2\le j\le 2m-5$时,$i=2m-5-j$.

以 m=16,n=13为例,扩展 X码磁盘阵列布局可由图 1表示如下:图 1中黑色填充的数据单元表示行校验单元 $b_{i,j}$($0\le i\le m-3, 0\le j\le 2m-5$),虚线标识的数据单元表示根据 X码编码方法产生的列校验单元,白色填充的数据单元表示有效数据块.

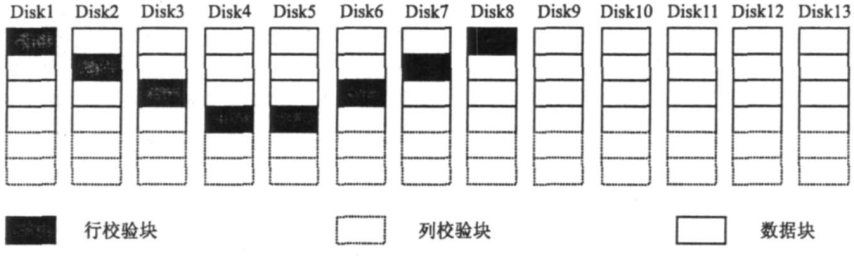

图 1 扩展 X码磁盘阵列布局

Fig. 1 Disk array arrangement of extended X code

3.2 编码算法

扩展 X码编码算法可以表示如下:

$$b_{i,j}=\oplus_{k=0}^{n}a_{i,k}\ (k\neq i,\ k\neq 2m-5-i)$$ (求和运算表示的是异或运算)

$$a_{m-2,j}=\oplus_{k=0}^{m-3}a_{k,(j+k+1)\bmod n}$$ (求和运算表示的是异或运算)

$$a_{m-1,j}=\oplus_{k=0}^{m-3}a_{k,(j-k-1)\bmod n}$$ (求和运算表示的是异或运算)

其中 $b_{i,j}$表示行校验单元(每行有两个相同的行校验单元),$a_{m-2,j}$和 $a_{m-1,j}$表示 X码编码方法产生的列校验单元.

3.3 译码算法

在详细描述译码算法前,首先引入三个辅助概念:列距、正向校验直线和反向校验直线.

若二维阵列 A中任意三列缺失(对应磁盘阵列中任意三个磁盘同时出现故障),设缺失数据列的列号为 i,j,k,且满足 i<j<k,定义 $d_1=j-i-1$,$d_2=k-j-1$,$d_3=n-1-k+i$,其中 d_1表示列 i到列 j的距离,其中 d_2表示列 j到列 k的距离,其中 d_3表示列 k到列 i的距离,这里定义的列距概念和传统的距离概念有所区别,指的是两个指定数据列中间存在的数据列个数,如果两个指定数据列相邻列距为 0.如果引入两个变量 v和 w,且满足如下条件:$1\le v\le 3$,$1\le w\le 3$($v\neq w$),v和 w均为整数,则区间[1,3]上还应该存在另一个整数

$s(s\neq v, s\neq w)$,若令 $d_v \leq m-4$, $d_w \leq m-4$,由于 $d_v + d_w + d_s = n-3$,且 $n > 3m-6$,则 $d_s > 3m-6-3-d_v-d_w > m-1 > m-3$, 若令 $d_v = m-3$, $d_w < = m-4$,则 $d_s > 3m-6-3-d_v-d_w > m-2 > m-3$,若令 $d_v = m-3$, $d_w = m-3$,则 $d_s > 3m-6-3-d_v-d_w > m-3$,若令 $d_v < -m-4$, $d_w = m-3$,则 $d_s > 3m-6-3-d_v-d_w > m-2 > m-3$,因此如果二维阵列中任意三列缺失必有相邻两列之间的距离大于 $m-3$.

为了表示正反向校验直线的概念,我们引入图 2.

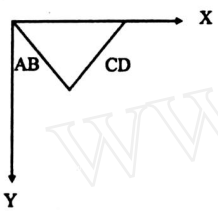

图 2　正反向校验直线图

Fig. 2　Positive and negative lineer check

如图 2所示,二维阵列 A的两个维度由 X轴和 Y轴表示,直线 AB和 CD分别代表 X码编码规则中产生校验数据的计算路径,直线 AB用于产生每个磁盘中编号为 $m-2$的数据块中的校验数据,直线 CD用于产生每个磁盘中编号为 $m-1$的数据块中的校验数据.直线 AB由于与 X轴呈 45度锐角,因此定义为正向校验直线,而直线 CD与 X轴呈 度钝角,因此定义为反向校验直线.

假设二维阵列 A中有两列发生错误,设发生错误数据列的列号为 i, j,且满足 $i < j$,令 d_{ij}表示数据列 i到数据列 j的列距, d_{ji}表示数据列 j到数据列 i的列距,根据前文的叙述,若 $d_{ij} < = m-3$,则 $d_{ji} > m-3$.

两列删除错译码算法:

(1)如果 $d_{ij} < = m-3$,则数据列 i第 $m-3$行上的数据可以通过正向校验直线对应的校验值得到恢复,数据列 j第 $m-3$行上的数据可以通过反向校验行上的数据可以通过正向校验直线对应的校验值得到恢复;如果 $d_{ij} > m-3$,则数据列 i第 $m-3$行上的数据可以通过反向校验直线对应的校验值得到恢复,数据列 j第 $m-3$行上的数据可以通过正向校验直线对应的校验值得到恢复,这样第 $m-3$行的数据整体得到恢复,转 2;

(2)根据步骤 中所述的方法可以顺次恢复错误数据列第 $m-4, m-5, \cdots, 0$行的数据,这样错误数据列中的有效数据得到恢复,转 3;

(3)根据编码规则 $a_{m-2,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j+k+1) \bmod n}$和 $a_{m-1,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j-k-1) \bmod n}$计算出错误数据列最后两行存储的校验数据.

令 $data_{i,j}$表示二维阵列 A中位于第 i行和第 j列的数据,设发生错误的数据列分别为 C_1, C_2, C_3,根据前文分析 C_1, C_2, C_3必然存在相邻两列之间的距离大于 $m-3$,不失一般性,令 $C_2 > C_1$且 C_1和 C_2之间的列距大于 $m-3$,具体分析如下:

若满足 $C_1 < C_2 < C_3$或 $C_3 < C_1 < C_2$,则 $C_2 - C_1 - 1 > m-3$,即 $C_2 - C_1 > m-2$,因此数据 $data_{0,C_1}$, $data_{m-3,C_2}$位于的正向数据校验直线中不包含错误数据列中的数据,数据 $data_{m-3,C_1}$, $data_{0,C_2}$位于的反向数据校验直线中不包含其它错误数据列中的数据.由于数据 $data_{m-3,C_1}$位于的反向数据校验直线经过计算得到的校验数据位于数据列 $C_1 + m-2$,数据 $data_{m-3,C_2}$位于的正向数据校验直线经过计算得到的校验数据位于数据列 $C_2 - (m-2)$,根据条件 $C_2 - C_1 > m-2$,因此数据列 $C_1 + m-2$和 $C_2 - (m-2)$都属于有效数据列,故而数据 $data_{m-3,C_1}$, $data_{m-3,C_2}$可以得到恢复,根据行编码可以恢复数据 $data_{m-3,C_3}$,这样错误数据列中第 $m-3$行的数据整体得到恢复.

若满足 $C_1 < C_2 < C_3$,则 $n-1-C_2+C_1 > m-3$,因此数据 $data_{0,C_1}$, $data_{m-3,C_2}$位于的反向数据校验直线中不包含错误数据列中的数据,数据 $data_{m-3,C_1}$, $data_{0,C_2}$位于的正向数据校验直线中不包含错误数据列中的数据.由于数据 $data_{m-3,C_1}$位于的正向数据校验直线经过计算得到的校验数据位于数据列 $(C_1 - (m-2) + n) \bmod n$,数据 $data_{m-3,C_2}$位于的正向数据校验直线经过计算得到的校验数据位于数据列 $(C_2 + (m-2)) \bmod n$.

当 $C_1 - (m-2) \geq 0$时,$0 \leq (C_1 - (m-2) + n) \bmod n < C_1$,因此数据列 $(C_1 - (m-2) + n) \bmod n$属于有效数据列;当 $C_1 - (m-2) < 0$时,据 $n - C_2 + C_1 > m-2$, $C_2 < (C_1 - (m-2) + n) \bmod n \leq n-1$,因此数据列 $(C_1 - (m-2) + n) \bmod n$依然属于有效数据列.

当 $C_2 + (m-2) \leq n-1$时,$C_2 < (C_1 - (m-2) + n) \bmod n \leq n-1$,因此数据列 $(C_2 + (m-2)) \bmod n$属于有效数据列;当 $C_2 + (m-2) \geq n$时,据 $n - C_2 + C_1 > m-2$, $0 \leq (C_2 + (m-2) \bmod n < C_1$,因此数据列 $(C_2 + (m-2)) \bmod n$依然属于有效数据列.

因而数据 $data_{m-3,C_1}$, $data_{m-3,C_2}$可以得到恢复,同时根据行编码可以恢复数据 ,这样错误数据列中第 行的数据整体得到恢复.

综上所述,无论错误数据列的分布情况如何,错误数据列中第 $m-3$行的数据都可以得到恢复.

若满足 $C_1 < C_2 < C_3$时,错误数据列中第 $m-3$行的数据可以得到恢复.数据 $data_{m-4,C_1}$位于的反向数据校验直线中除了数据 $data_{m-3,(C_1-1+n) \bmod n}$外其它数据都位于有效数据列,若 $(C_1 - 1 + n) \bmod n \neq C_3$则数据 $data_{m-3,(C_1-1+n) \bmod n}$依然位于有效数据列;反之由于 $data_{m-3,C_3}$已被求解出因此也属于已知数据.由于数据 $data_{m-4,C_1}$位于的反向数据校验直线计算产生的校验数据位于数据列 $C_1 + m-3$,根据条件 $C_2 - C_1 > m-2$,数据列 $C_1 + m-3$为有效数据列,因此 $data_{m-4,C_1}$可以被求出.同理,数据 $data_{m-4,C_2}$位于的正向数据校验直线中除了数据 $data_{m-3,C_2+1}$外其它数据都位于有效数据列,若 $C_2 + 1 \neq C_3$则数据 $data_{m-3,C_2+1}$依然位于有效数据列;反之由于 $data_{m-3,C_3}$已被求解出因此也属于已知数据.由于数据 $data_{m-4,C_2}$位于的正向数据校验直线计算产生的校验数据位于数据列 $C_2 - (m-$

3),根据条件 $C_2 - C_1 > m - 2$,数据列 $C_2 - (m - 3)$为有效数据列,因此 $data_{m-4,C_2}$可以被求出,同时根据行编码可以恢复数据 $data_{m-4,C_3}$,故而错误数据列中第 m - 4行可以被恢复.同理,错误数据列中第 m - 5, m - 6, …, 0都可以被迭代恢复出,错误数据列中的有效数据得到整体恢复.

若满足 $C_3 < C_1 < C_2$时,错误数据列中第 m - 3行的数据得到恢复.数据 $data_{m-4,C_1}$位于的反向数据校验直线中除了数据 $data_{m-3,C_1-1}$外其它数据都位于有效数据列,若 $C_1 - 1 \neq C_3$则数据 $data_{m-3,C_1-1}$依然位于有效数据列;反之由于 $data_{m-3,C_3}$已被求解出因此也属于已知数据.由于数据 $data_{m-4,C_1}$位于的反向数据校验直线计算产生的校验数据位于数据列 $C_1 + m - 3$,根据条件 $C_2 - C_1 > m - 2$,数据列 $C_1 + m - 3$为有效数据列,因此 $data_{m-4,C_1}$可以被求出.同理,数据 $data_{m-4,C_2}$位于的正向数据校验直线中除了数据 $data_{m-4,(C_2+1)\bmod n}$外其它数据都位于有效数据列,若 $(C_2 + 1) \bmod n \neq C_3$则数据 $data_{m-4,C_2+1}$依然位于有效数据列;反之由于 $data_{m-3,C_3}$已被求解出因此也属于已知数据.由于数据 $data_{m-4,C_2}$位于的正向数据校验直线计算产生的校验数据位于数据列 $C_2 - (m - 3)$,根据条件 $C_2 - C_> m - 2$,数据列 $C_2 - (m - 3)$为有效数据列,因此 $data_{m-4,C_2}$可以被求出,同时根据行编码可以恢复数据 $data_{m-4,C_3}$,故而错误数据列中第 m - 4行可以被恢复.同理,缺失数据列中第 m - 5, m - 6, …, 0都可以被迭代恢复出,错误数据列中的有效数据得到整体恢复.

若满足 $C_1 < C_3 < C_2$时,错误数据列中第 m - 3行的数据得到恢复.数据 $data_{m-4,C_1}$位于的正向数据校验直线中除了数据 $data_{m-3,C_1+1}$外其它数据都位于有效数据列,若 $C_1 + 1 \neq C_3$则数据 $data_{m-3,C_1-1}$依然位于有效数据列;反之由于 $data_{m-3,C_3}$已被求解出因此也属于已知数据.由于数据 $data_{m-4,C_1}$位于的反向数据校验直线计算产生的校验数据位于数据列 $(C_1 - (m - 3) + n) \bmod n$,根据条件 $n - C_2 + C_1 > m - 2$,数据列 $(C_1 - (m - 3) + n) \bmod n$为有效数据列,因此 $data_{m-4,C_1}$可以被求出.同理,数据 $data_{m-4,C_2}$位于的反向数据校验直线中除了数据 $data_{m-3,C_2-1}$外其它数据都位于有效数据列,若 $C_2 - 1 \neq C_3$则数据 $data_{m-3,c_2-1}$依然位于有效数据列;反之由于 $data_{m-3,C_3}$已被求解出因此也属于已知数据.由于数据 $data_{m-4,C_2}$位于的反向数据校验直线计算产生的校验数据位于数据列 $(C_2 + (m - 3)) \bmod n$,根据条件 $n - C_2 + C_1 > m - 2$,数据列 $(C_2 + (m - 3)) \bmod n$为有效数据列,因此 $data_{m-4,C_2}$可以被求出,同时根据行编码可以恢复数据 $data_{m-4,C_3}$,故而错误数据列中第 m - 4行可以被恢复.同理,错误数据列中第 m - 5, m - 6, …, 0都可以被迭代恢复出,错误数据列中的有效数据得到整体恢复.

三列删除错译码算法:

(1)首先将错误数据列的列号排序为 $i < j < k$,根据前文列距的定义计算 d_w $(w = 1 \cdots 3)$.

(2)如果 $d_1 > m - 3$,则根据前文所述 $C_1 < C_2 < C_3$的情况进行数据恢复;如果 $d_2 > m - 3$,则根据前文所述 $C_3 < C_1 < C_2$的情况进行数据恢复;如果 $d_3 > m - 3$,则根据前文所述 $C_1 < C_3 < C_2$的情况进行数据恢复.

(3)错误数据列的有效数据部分得到恢复后,根据编码规则 $a_{m-2,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j+k+1) \bmod n}$和 $a_{m-1,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j-k-1) \bmod n}$计算出错误数据列最后两行存储的校验数据.

如果发生单列突发错误,即在大小为 m × n的二维阵列中,其中某一列出错,但是并不知道出错列的位置,纠错的关键是找到错误位置,具体方法如下:

(1)首先根据行校验元素 $b_{i,j}$分布的位置比较编号为 0…m - 3的各行中的两个行校验元素 $b_{i,j1}$, $b_{i,j2}$是否相同,如果不同,则求该行其它数据的校验和 $s = \oplus \sum_{k=0}^{n} a_{i,k}$ $(k \neq j1, k \neq j2)$,如果 $b_{i,j1} \neq s$,则编号为 j1的数据列发生错误,否则编号为 j2的数据列发生错误,转步骤;如果 $b_{i,j1}$和 $b_{i,j2}$的数值相同,根据前述过程继续下一行的计算,若编号为 0…m - 3的各行中的 $b_{i,j1}$和 $b_{i,j2}$的数值都相同则继续步骤;

(2)计算所有正向校验直线代表的校验值,如果与数据列中相应位置存储的校验值不同,则记录该校验值存储位置的数据对 $(x_n^{(p)}, y_n^{(p)})$ $(m - 2 \leq x_n^{(p)} \leq m - 1, 0 \leq y_n^{(p)} \leq n - 1)$于集合 P中,其中 $x_n^{(p)}$表示列号, $y_n^{(p)}$表示行号, n表示与正向校验直线相关的数据对数量;同理可计算出与反向校验直线相关的数据对 $(x_n^{(o)}, y_n^{(o)})$ $(m - 2 \leq x_n^{(o)} \leq m - 1, 0 \leq y_n^{(o)} \leq n - 1)$并将其存储在于集合 O中,其中 $x_n^{(o)}$表示列号, $y_n^{(o)}$表示行号, n表示与反向校验直线相关的数据对数量,计算完毕转步骤 3;

(3)如果步骤 2中记录的数据集 O为空, P中仅有一个数据对,则 P中数据对存储的列号对应错误数据列,转步骤 4;如果数据集 P为空, O中仅有一个数据对,则 O中数据对存储的列号对应错误数据列,转步骤;如果数据集 P和 O都仅有一个数据对,且数据对中存储的列号相同,则列号对应的数据列发生错误,转步骤 4;否则转步骤 5;

(4)按照行编码方法首先恢复错误列前 m - 2行的有效数据,然后按照公式 $a_{m-2,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j+k+1) \bmod n}$和 $a_{m-1,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j-k-1) \bmod n}$计算出错误列最后两行存储的校验数据.

(5)从编号为 0的数据列开始,对于 $R(i_1, i_2, \cdots, i_l)$中的每个元素 i_u $(u = 1, 2, \cdots, l)$按照行编码方法纠错,每一列前 m - 2个元素纠错完毕后,重新计算正反向校验直线的数值,如果除纠错列外其他数据列存储的校验数据与计算结果完全相同,则当前纠错列为错误列,按照公式 $a_{m-2,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j+k+1) \bmod n}$和 $a_{m-1,j} = \oplus \sum_{k=0}^{m-3} a_{k,(j-k-1) \bmod n}$计算出错误列最后两行存储的校验数据.

3.4 扩展 X码和 RS码性能比较

下页表 1和表 2是 RS码和扩展 X码在纠正单列突发错、两列删除错、三列删除错时通过实验得到的性能参数,从上述两个表中可以看出扩展 X码在接近于两倍 RS码的码长条件下,当纠正同样多的错误时,编码时间约为 RS码编码时间的 1/5,单列突发错译码时间约为 RS码的 2/3,两列和三列

删除错译码时间约为 RS码的 ,从以上性能比较来看扩展 X 码的编译码性能上远远超过 RS码.

表 1 RS码编译码时间性能指标

Table 1 Encoding and decoding time of rs code

RS码	单列突发错			两列删除错			三列删除错		
码长	1024	1024	1024	1024	1024	1024	1024	1024	1024
冗余量	216	316	416	216	316	416	216	316	416
错误数量	40	60	80	80	120	160	120	180	240
编码时间	0.014051s	0.016918s	0.019649s	0.014051s	0.016918s	0.019649s	0.014051s	0.016918s	0.019649s
译码时间	0.021849s	0.032878s	0.044872s	0.025343s	0.040916s	0.057869s	0.029704s	0.051674s	0.070054s

表 2 扩展 X码编译码时间性能指标

Table 2 Encoding and decoding time of extended X code

扩展 X码	单列突发错			两列删除错			三列删除错		
磁盘数	25	25	25	25	25	25	25	25	25
单个磁盘块数	10	10	10	10	10	10	10	10	10
块长	4	6	8	4	6	8	4	6	8
总长	1000	1500	2000	1000	1500	2000	1000	1500	2000
冗余量	216	316	416	216	316	416	216	316	416
错误数量	40	60	80	80	120	160	120	180	240
编码时间	0.001086s	0.0011s	0.001179s	0.001086s	0.011s	0.001179s	0.001086s	0.011s	0.01179s
译码时间	0.005946s	0.00602s	0.006051s	0.000202s	0.000205s	0.000219s	0.000227s	0.000233s	0.000250s

4 应用前景

在火箭飞行过程中, 火箭姿态的调整具有至关重要的作用.在火箭复杂的控制系统中有专门负责火箭姿态控制的惯性参考系统, 在该系统中具有多台负责计算火箭飞行参数的计算机, 其中一台计算机为主控计算机, 其它计算机为备用计算机, 当主控计算机发生故障时备用计算机接管控制任务.由于主控计算机和备用计算机同步计算火箭飞行姿态参数,因此当多台计算机上运行同一控制程序时,如果控制程序出错,主控计算机和备用计算机可能先后进入故障状态,从而造成火箭失控, 1996年 6月,欧盟发射的阿丽安娜火箭升空到达 4000米高度时发生爆炸就是基于上述原因.

为了解决上面提出的问题, 研究人员提出了 N版编程的思想, 即由 N个独立的团队开发 N个具有相同功能的控制软件, 并将它们分别安装于主控和备用计算机上, 这样当主控计算机上的控制程序出错时, 由于备用计算机上运行的是不同版本的控制软件因此不会随之进入故障状态.但是由于主控和备用计算机上运行的是不同的控制程序,因此程序运行的速度不同,因此当主控计算机处于故障状态时,备用计算机还未运行到相应的正常状态,这样存储主控计算机和备用计算机的校验点状态就不可避免,因此引入校验服务器存储主控计算机和备用计算机的校验点状态,由于惯性参考系统中的计算机数量有限且每台计算机的校验点信息也并非海量存储,一般每隔 5-10分钟存储一次校验点信息,这样校验服务器的数量不会太多,一般在 7、8台左右,根据前文提到的扩展 X码,当校验服务器中有三台出现故障时校验服务器中的校验点信息仍然可以恢复,这样就进一步提高了火箭控制系统的安全性.

5 结 论

控制系统的一个重要特征是其安全性,冗余技术作为一种主要的容错技术得到广泛应用,软件冗余技术由于具有代价小、易于实现的特点,在各类冗余技术中得到深入研究,为此研究人员提出了很多有效的容错设计模式,其中校验点恢复模式和流程对模式最具代表性,但是在上述设计模式中存在备用服务器中存储的校验点信息损毁造成备用服务器无法有效接管主服务器的问题,针对上述问题本文引入编码技术以保障控制系统的安全,并提出了一种新型编码技术 扩展 X码,该类编码是在一类经典阵列码 -X 码的基础上发展起来的, X码方法仅容许备用服务器的磁盘阵列中任意两个磁盘同时发生故障, 这就造成磁盘阵列的纠错率低、可靠性不高, 扩展 X码可以容许磁盘阵列中任意三个磁盘同时发生故障, 从而增强了纠错率,同时扩展 X 码与磁盘容错经常采用的 RS码相比解码时间要少的多,因此可以有效应用于火箭控制系统这类实时性高且校验点信息量不大的系统.

本文提出的扩展 X 码将备用服务器中磁盘阵列的纠错能力从 提高到了 .对于更高的纠错需求, 如何改进现有的磁盘阵列布局和编译码方法是今后进一步的研究方向.

References:

[1] Zhong S W ang. Application of intelligent fault-tolerance [M]. Beijing: Nation Defence Industry Publishing House, 2002.

[2] Patton R. J. Fault-tolerant control [C]. Proceeding of IFAC/IMACS Symposium on Fault Detection, Supervision and Safety for Technical Process, 1997, 1033-1055.

[3] Avizienis A, Chen L. On the implementation of N-version programming for software fault tolerance during Execution[C]. Proceeding of the 1st International Computer Software and Application Conference (COMPSAC), 1977, 149-155.

[4] Randell B. System Structure for Software Fault Tolerance [J]. IEEE Transaction on Software Engineer, 1975, 1(2): 220-232.
[5] Richard Koo, Sam Toueqt. Checkpoing and rollback-recovery for distributed systems [J]. IEEE Transaction on Software Engineering, 1987, 1(13): 23-31.
[6] Mitch Cherniack, Hari Balakrishnan, Magdalena Balazinska, et al. Scalable distributed stream processing [C]. Proceeding of the 1st Conference on Innovative Data System Research (CIDR), 2003, 1-12.
[7] Xin M Wang, Guo Z Xiao. Theory of error-correcting code [M]. Xi'an: Xian Electronic University Publishing House, 1991.
[8] Xu L, Bruck J. X-code: MDS array codes with optimal encoding [J]. IEEE Trans. on Information theory, 1999, 45(1): 272-276.
[9] Blaum M, Brady J, Menon J. EVENODD: an efficient for tolerating double disk failures in RAID architectures [J]. IEEE Trans. on Computation, 1995, 44(2): 192-202.
[10] Xu L, Bohossian V, Bruck J, et al. Low density MDS codes and factors of complete graphs [J]. IEEE Trans. on Information Theory, 1999, 45(6): 1817-1826.
[11] Katti R, Ruan Xiao-yu. S-code: new distance-3 MDS array codes with optimal encoding [C]. Proceeding of IEEE ICASSP '05, Philadelphia, 2005, 1105-1108.

附中文参考文献:

[1] 王仲生. 智能容错技术及应用 [M]. 北京:国防工业出版社, 2002.
[7] 王新梅, 肖国镇. 纠错码—原理与方法 [M]. 西安:西安电子科技大学出版社, 1991.

征 稿 简 则

一、征稿范围:《小型微型计算机系统》杂志刊登文章的内容涵盖计算技术的各个领域(计算数学除外). 包括计算机科学理论、体系结构、计算机软件、数据库、网络与通讯、人工智能、信息安全、多媒体、计算机图形与图像、算法理论研究等各方面的学术论文.

二、来稿要求:本刊主要刊登下述各类原始文稿:

1. 学术论文:科研成果的有创新、有见解的完整论述. 对该领域的研究与发展有促进意义,论文字数最好在 10000字左右.
2. 综述:对新兴的或活跃的学术领域或技术开发的现状及发展趋势的全面、客观的综合评述.
3. 技术报告:在国内具有影响的重大科研项目的完整的技术总结.

三、注意事项

1. 来稿务求做到论点明确、条理清晰、数据可靠、叙述简练,词义通达.
2. 来稿必须是作者自己的科研成果,无署名和版权争议. 引用他人成果必须注明出处.
3. 本刊采用在线投稿方式,可登陆 http://xwxt.sict.ac.cn/zxtg/zxtgyhlogin.asp?lx=1进行在线投稿,也可直接邮寄打印稿一式两份.
4. 格式要求:题目(中、英文)、摘要(中、英文)、作者的真实姓名(中、英文)、作者的单位、城市(中、英文)、邮政编码、E-mail(便于联系的)、关键词(中、英文 4-7个),中图分类号、作者简介、基金项目.

(1) 英文部分的作者姓名使用汉语拼音,单位英文名称须给出英文全称,不要使用缩略语;

(2) 作者简介包含作者姓名、性别、出生年、最高学历、技术职称、研究方向(若作者中有中国计算机学会(CCF)会员,请注明,并给出会员号). 凡第一作者为 CCF会员/高级会员/学生会员者,将享受八五折的版面费优惠;

(3) 基金项目的类别与项目编号.

5. 中、英摘要:文章摘要具有独立性和自明性,含正文等量的主要信息,一般为 150～200字,采用第三人称表述.
6. 参考文献:未公开发表的文献不得列入. 文后所列参考文献统一排序,且必须在正文中引用. 中文参考文献应给出对应的英文译文. 其具体书写格式为:

(1) 图书: [编号]作者姓名(姓在前,名在后),书名,出版社地址,出版社,出版年.

(2) 期刊: [编号]作者姓名、文章题目、刊物名称,出版年,卷号(期号):起止页码.

(3) 会议论文: [编号]作者姓名. 论文题目. 见:编者、论文集全名、出版地:出版者,出版年,起止页码.

(4) 网络文献:请给出文献作者或单位名,文章题目、网址、发布日期.

7. 插图和表:插图必须精绘并用计算机激光打印,一般不超过 7幅. 图应结构紧凑,不加底纹,图宽最好不超过 8厘米,图内字号统一使用 6号宋体,字迹、曲线清晰,必要时给出坐标名称和单位. 每个图、表均给出中英文图注(如:"图 1: * * *图;""Fig. 1: * * * *")和表注(如:"表 1: * * *表","Table 1: * * * *").
8. 计量单位:稿件中一律使用《中华人民共和国法定计量单位》. 外文和公式中应分清大、小写和正、斜体,上、下角的字母、数码位置准确. 易混淆的字母或符号,请在第一次出现时标注清楚.
9. 本刊在收到作者稿件经初审后立即给作者电子邮箱发"稿件收到通知". 除作者另有明确要求外,本刊原则上只与第一作者联系,作者投稿后若 4个月无消息,可自行改投它刊. 通过初审的稿件将收到本刊给予的编号,并需邮寄打印稿及审稿费.
10. 本刊对不拟录用的稿件只发给"退稿通知",恕不退回原稿,请自留底稿.
11. 稿件一经发表,将酌致稿酬,并寄送样刊.

本刊文章现被国内外多家数据库收录,作者著作权使用费与本刊稿酬一并给付,作者若不同意将文章收录,请在投稿时说明.

来稿请寄:沈阳市浑南新区南屏东路 16号《小型微型计算机系统》编辑部

邮政编码: 110171 电话: (024) 24696120 E-mail: xwjxt@sict.ac.cn 网址: http://xwxt.sict.ac.cn