高等计算机系统结构

RDP RAID-6 的校验构建与数据恢复

2016-1-11

戴国浩

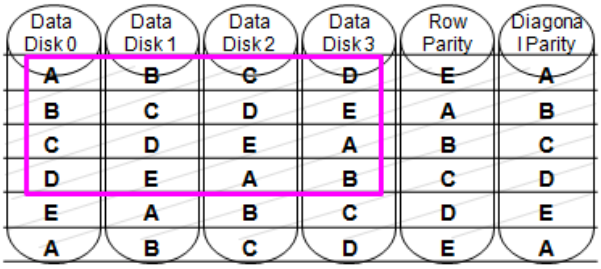
2014310490

一、设计要求

假定已知：

- Since it is protecting against a double failure, it adds two check blocks per stripe of data. If p+1 disks total, p-1 disks have data;
- Row parity disk is just like in RAID 4. Even parity across the other 4 data blocks in its stripe.
- Each block of the diagonal parity disk contains the even parity of the blocks in the same diagonal.

下图给出当 p=5 时，**RDP RAID-6 的数据布局。**



Data Disk 0	Data Disk 1	Data Disk 2	Data Disk 3	Row Parity	Diagonal Parity
A	B	C	D	E	A
B	C	D	E	A	B
C	D	E	A	B	C
D	E	A	B	C	D
E	A	B	C	D	E
A	B	C	D	E	A

请写模拟程序，当给定一个 RDP RAID-6 规模参数 p 的时候，运行该模拟程序并给出：

（1） 依次给出该 RAID，根据数据构建行校验、对角线校验的执行过程。即描述出用哪些数据块构建行校验块 0/1/2……，用哪些数据块和哪个行校验块构建对角线校验块 0/1/2……。

（2） 当用户指定出错的两块磁盘时，该程序能够给出用于恢复失效盘上数据块的操作序列。失效盘可以是数据盘、行校验、对角线校验。

说明：

（1） RDP RAID-6 的编码规则参见 PPT，并以课上讲授为准；

（2） 完成后请提交：（i）设计思路文档；（ii）源代码；（ii）典型配置的输出记录。

二、设计思路

编写 ConstructBlock()函数构造磁盘阵列的前 P-1 行，如下：

```
1   void ConstructBlock(char* block, int P)
2   {
3       for (int i = 0; i < P-1; i++)
4       {
5           char label = char('A'+i);
6           bool skip = false;
7           int ptr = i;
8           for (int j = P; j >= 0; j--)
9           {
10              if (!skip)
11              {
12                  block[ptr*(P+1)+j] = label;
13                  ptr = (ptr+1)%(P-1);
14                  if (ptr == 0) skip = true;
15              }
16              else
```

```
17              {
18                  skip = false;
19              }
20          }
21      }
22      int ptr = P - 2;
23      for (int i = 1; i < P; i++)
24      {
25          block[ptr*(P+1)+i] = char('A'+P-1);
26          ptr--;
27      }
28  }
```

以 P = 7 为例，前 6 行的对角线构建规则如下：

```
Please input P:7

-----------------------------------------------------

Data Pattern

D0      D1      D2      D3      D4      D5      RP      DP
A       B       C       D       E       F       G       A
B       C       D       E       F       G       A       B
C       D       E       F       G       A       B       C
D       E       F       G       A       B       C       D
E       F       G       A       B       C       D       E
F       G       A       B       C       D       E       F
```

其中 DX（X = 0~P-2），共 P-1 块为数据磁盘；RP 为行校验，构建方法为所在行数据块的校验；DP 为对角校验，构建方法为图中出自身外所有相同字母的校验。

当其中两块数据磁盘发生损坏时，分两种情况进行恢复：

1、损坏磁盘中有对角校验盘

（1）先恢复所有利用行校验码恢复所有非对角校验盘。
（2）利用对角校验码恢复对角校验盘。
值得指出的是，两个步骤中都可以并行恢复多块磁盘。

2、损坏磁盘都是非对角校验盘

（1）在两块损坏磁盘的(P-1)*2 个数据块中找出一个可以利用对角校验码恢复的数据块。
（2）和在（1）中利用对角校验码恢复的数据块同一行的数据块利用行校验码恢复。
（3）重复（1）（2）步直至恢复完成。

三、实验结果

如上分析对两种损坏情况分别进行恢复，以 P=7 为例

1、损坏磁盘中有对角校验盘

不妨假设数据磁盘 D0 和对角校验磁盘发生损坏，恢复结果如下：

```
Please input the id of two failed disk(0^P, P-1 --> RP, P --> DP):0 7

--------------------------------------------------
Recovering...
Recover in parallel...
Recover (0, 0) using: (1, 0) (2, 0) (3, 0) (4, 0) (5, 0) (6, 0)
Recover (0, 1) using: (1, 1) (2, 1) (3, 1) (4, 1) (5, 1) (6, 1)
Recover (0, 2) using: (1, 2) (2, 2) (3, 2) (4, 2) (5, 2) (6, 2)
Recover (0, 3) using: (1, 3) (2, 3) (3, 3) (4, 3) (5, 3) (6, 3)
Recover (0, 4) using: (1, 4) (2, 4) (3, 4) (4, 4) (5, 4) (6, 4)
Recover (0, 5) using: (1, 5) (2, 5) (3, 5) (4, 5) (5, 5) (6, 5)
Recover in parallel...
Recover (7, 0) using: (6, 1) (5, 2) (4, 3) (3, 4) (2, 5) (0, 0)
Recover (7, 1) using: (6, 2) (5, 3) (4, 4) (3, 5) (1, 0) (0, 1)
Recover (7, 2) using: (6, 3) (5, 4) (4, 5) (2, 0) (1, 1) (0, 2)
Recover (7, 3) using: (6, 4) (5, 5) (3, 0) (2, 1) (1, 2) (0, 3)
Recover (7, 4) using: (6, 5) (4, 0) (3, 1) (2, 2) (1, 3) (0, 4)
Recover (7, 5) using: (5, 0) (4, 1) (3, 2) (2, 3) (1, 4) (0, 5)

--------------------------------------------------
```

可以看到，恢复数据磁盘的多个数据块可以并行，恢复对角校验磁盘的多个数据块也可以并行。

2、损坏磁盘都是非对角校验盘

不妨假设数据磁盘 D1 和 D2 发生损坏，恢复结果如下：

```
Please input the id of two failed disk(0^P, P-1 --> RP, P --> DP):1 2

--------------------------------------------------
Recovering...
Recover (5, 2) using: (0, 0) (0, 7) (1, 6) (2, 5) (3, 4) (4, 3)
Recover (5, 1) using: (5, 0) (5, 2) (5, 3) (5, 4) (5, 5) (5, 6)
Recover (0, 1) using: (1, 0) (1, 7) (2, 6) (3, 5) (4, 4) (5, 3)
Recover (0, 2) using: (0, 0) (0, 1) (0, 3) (0, 4) (0, 5) (0, 6)
Recover (1, 1) using: (0, 2) (2, 0) (2, 7) (3, 6) (4, 5) (5, 4)
Recover (1, 2) using: (1, 0) (1, 1) (1, 3) (1, 4) (1, 5) (1, 6)
Recover (2, 1) using: (0, 3) (1, 2) (3, 0) (3, 7) (4, 6) (5, 5)
Recover (2, 2) using: (2, 0) (2, 1) (2, 3) (2, 4) (2, 5) (2, 6)
Recover (3, 1) using: (0, 4) (1, 3) (2, 2) (4, 0) (4, 7) (5, 6)
Recover (3, 2) using: (3, 0) (3, 1) (3, 3) (3, 4) (3, 5) (3, 6)
Recover (4, 1) using: (0, 5) (1, 4) (2, 3) (3, 2) (5, 0) (5, 7)
Recover (4, 2) using: (4, 0) (4, 1) (4, 3) (4, 4) (4, 5) (4, 6)

--------------------------------------------------
```

可以看到，数据磁盘 D1 和 D2 的恢复是利用行校验和对角校验交替进行。