高容错自维护存储系统

TStor

郑纬民

清华大学计算机系

提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

四.高容错自维护存储系统特征

五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

一.高容错自维护存储系统定义

高容错自维护存储系统是指存储系统交付使用之后，在其生命周期内不发生数据丢失，并且无需管理员进行维护的存储系统。

提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

四.高容错自维护存储系统特征

五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

可靠性变得日益关键

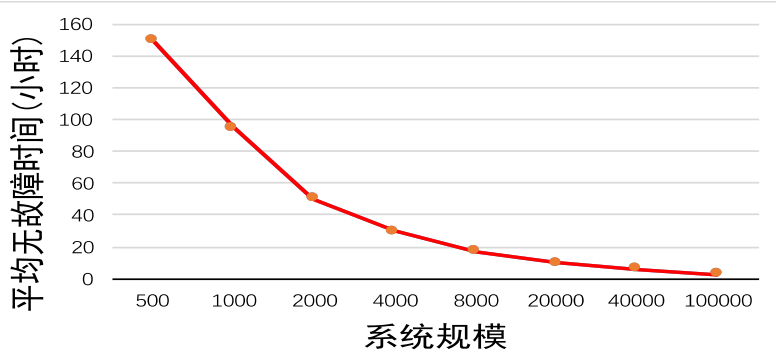
- 系统中存储设备类多量大，设备**出错成为一种常态**，但要求系统仍正常提供服务。

可靠性不足造成的**损失越来越大**

数据损失代价

EMC 2014年报告：

- 过去12个月中，数据损失和宕机给企业带来的损失高达**1.7万亿美元**
- 2012年以来数据损失的代价**增长了400%**



平均无故障时间随存储系统规模增大而减小[1]

[1] Fabrizio Petrini and Kei Davis and José Carlos Sancho. System-Level Fault-Tolerance in Large-Scale Parallel Machines with Buffered Coscheduling. FTPDS04, 2004.

存储系统日益复杂

存储设备多样化

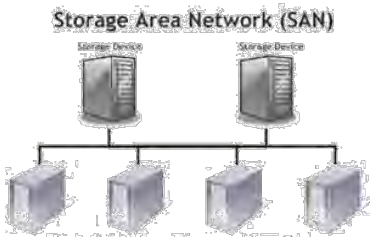
硬盘

SSD

InfiniBand
各种不同网络

系统规模不断增长

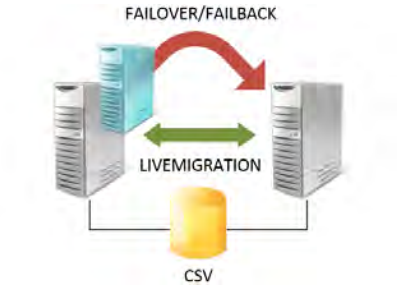
NAS



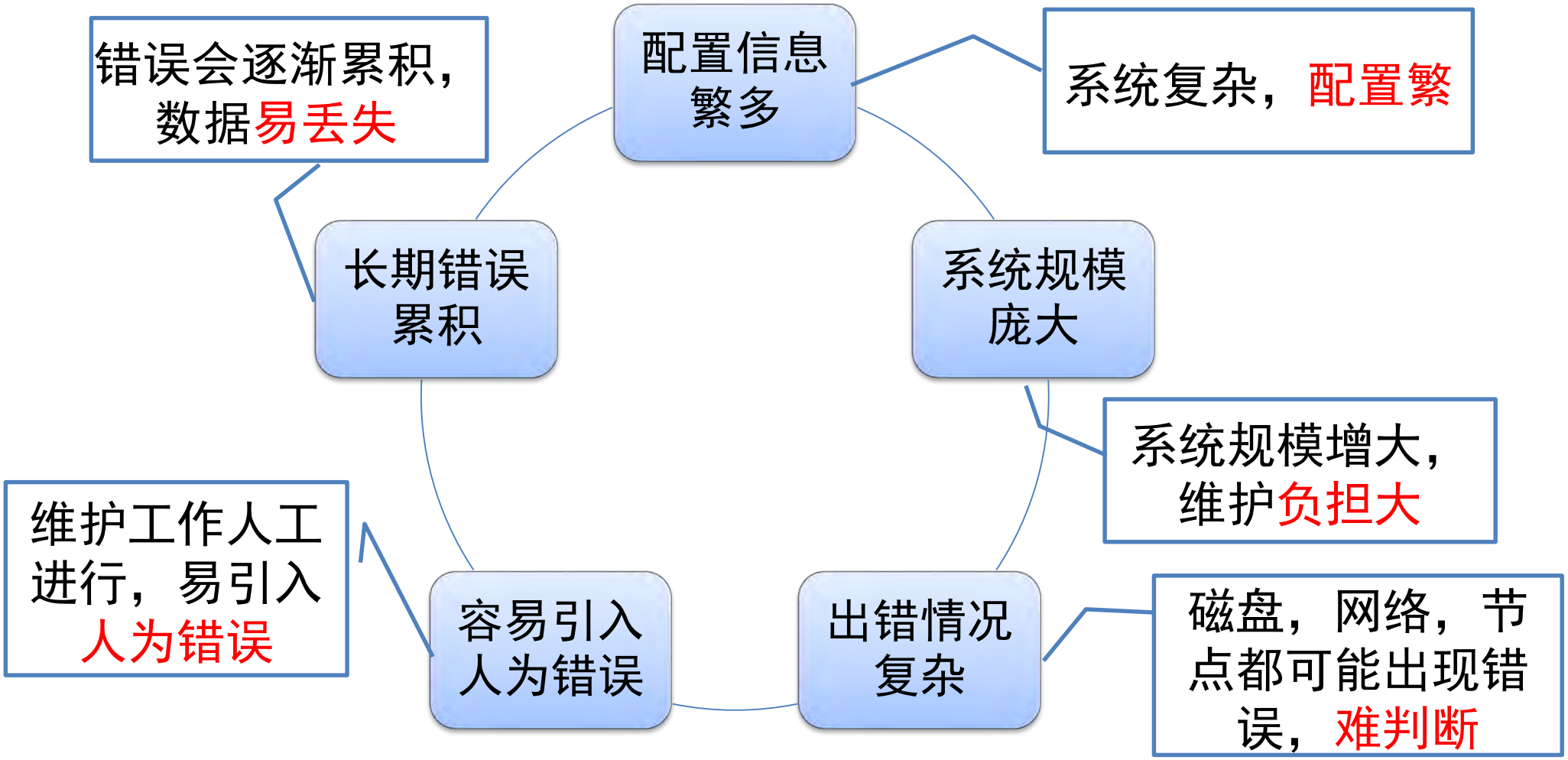
SAN

集群存储

系统可用性要求不断提高



存储系统维护负担繁重


配置信息繁多
系统复杂，配置繁
系统规模庞大
系统规模增大，维护负担大
出错情况复杂
磁盘，网络，节点都可能出现错误，难判断
容易引入人为错误
维护工作人工进行，易引入人为错误
长期错误累积
错误会逐渐累积，数据易丢失

维护成本高

某大型互联网公司，存储设备成本50%，网络带宽成本是40%，维护成本10%。中小规模企事业单位维护成本更高。

提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

四.高容错自维护存储系统特征

五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

三.高容错自维护存储系统目标

1. 一次部署，长期运行
2. 系统无需重新配置
3. 错误模块自动屏蔽
4. 避自错误累积造成数据丢失

提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

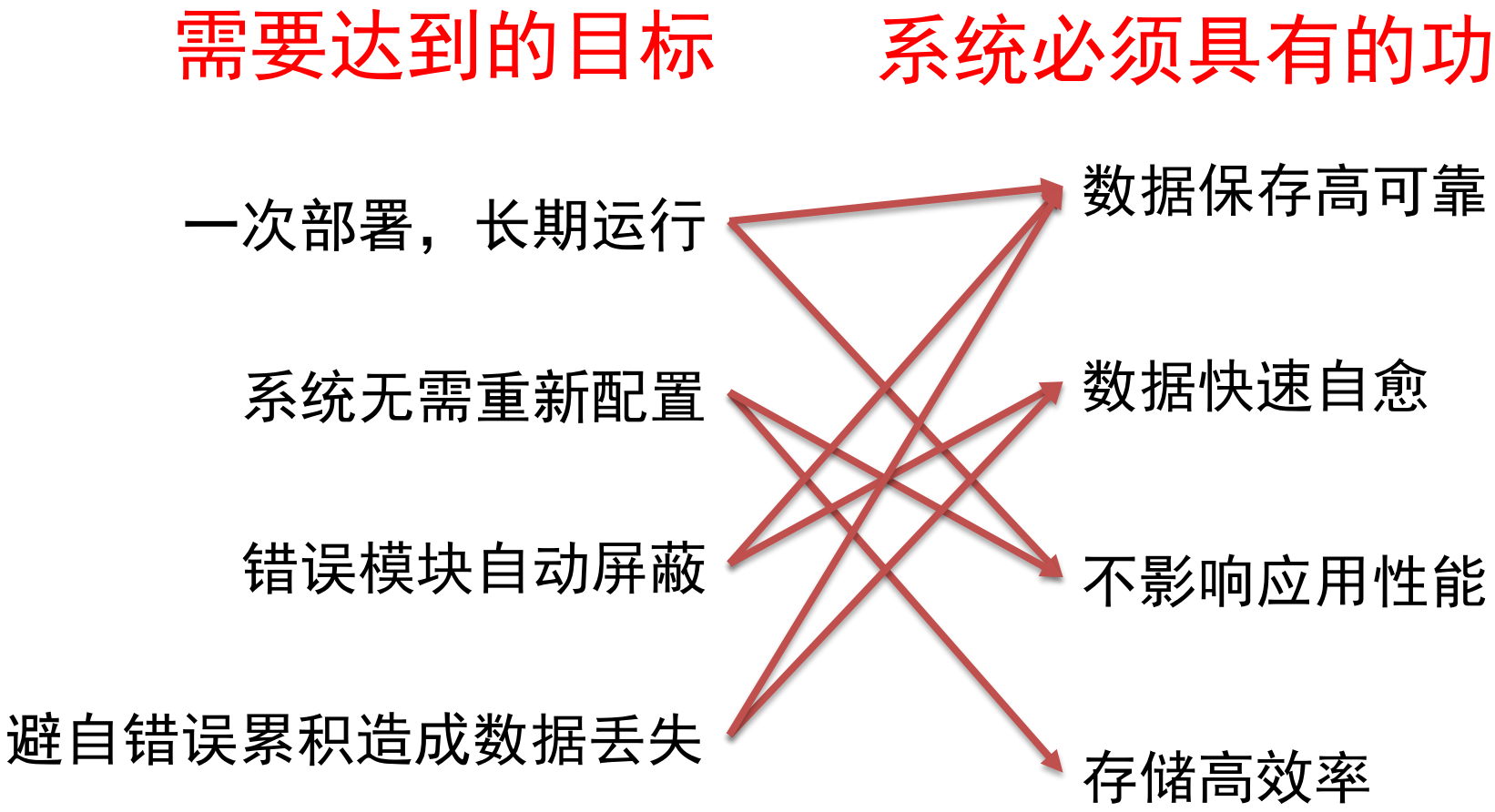
四.高容错自维护存储系统特征

五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

四.高容错自维护存储系统特征



提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

四.高容错自维护存储系统特征

五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

五.现有存储系统的可靠性保证技术以及分析

- 数据可靠性是构成高容错自维护存储系统的关键

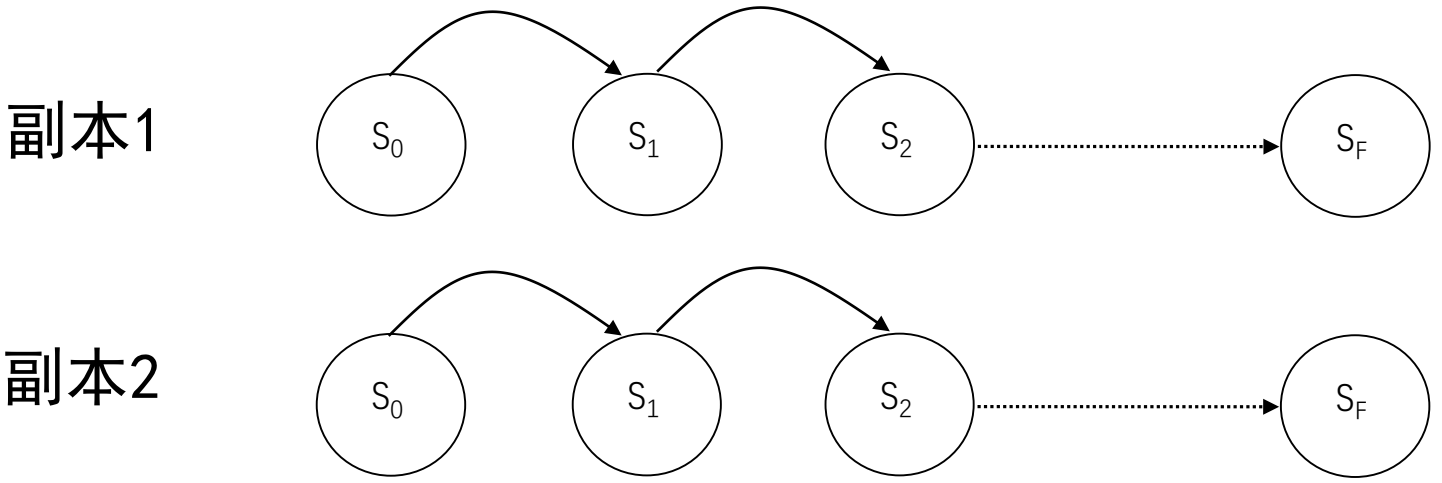
- 现有的三种数据高可靠性技术
  - ✓数据副本技术
  - ✓冗余磁盘阵列技术RAID
  - ✓数据纠删码技术

数据副本技术

- 基本思想：将一份数据保存到多个不同的物理位置，在一些副本失效的时候其余副本可以继续工作，即一个逻辑数据有多个物理副本
- 缺点：需要很大的冗余的磁盘空间，通常使用3个副本的方式，需要额外的200%的存储空间
- 实现困难：需要维护数据的一致性，通常使用副本状态机的方式实现

数据副本技术的实现方法

- 使用副本状态机的方法，将同样的一组写入操作通过同样的顺序写入到相同的数据对象中。如果初始的对象是相同的话，由于所有的操作都是确定性的，那么通过相同的步骤之后，结果数据也是相同的



数据副本技术的分析

- 数据可靠性：依据副本的数目，可以调整可靠性，一般3个副本被认为很可靠，更高的副本会需要过多的存储空间不被接受
- 数据自愈能力：如果节点数目足够多，数据块进行恢复的时候可以进行快速恢复，前提是文件数据必须进行切块
- 应用性能影响：由于快速自愈，对应用程序的影响比较少
- 存储效率：存储效率极差，需要很多的额外的存储空间才能够保证数据可靠

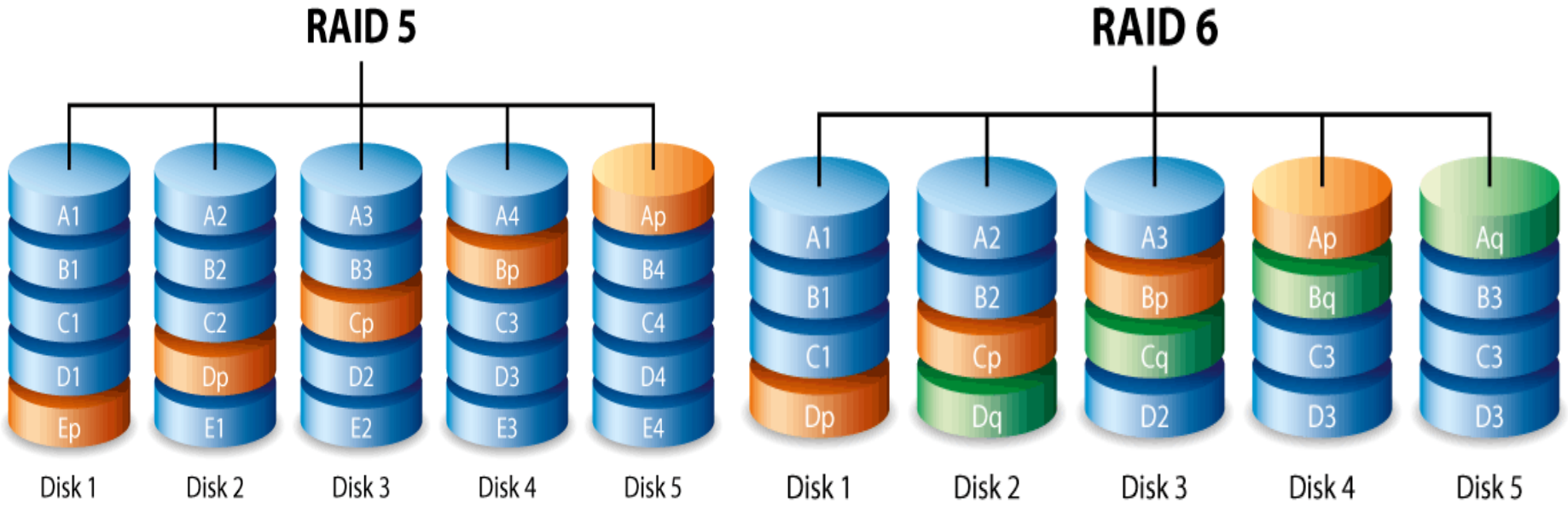
- 数据副本技术的最大问题是存储效率太低，在相同条件下需要选择其它技术而不是数据副本技术

冗余磁盘阵列技术

- 基本思想：通过硬件的方法，将分布到多块的磁盘数据进行校验，可以容忍一块或者两块的磁盘错误
- 冗余磁盘阵列技术有不同的级别，包括常见的RAID0,RAID1,RAID5,RAID6等
- 为了提供可靠性以及存储效率，通常使用RAID5或者RAID6的方式
- 原始数据与校验数据在所有的磁盘中交错存放，以保证磁盘之间的性能负载均衡

冗余磁盘阵列的实现方式

- 仅以RAID5以及RAID6作为例子，兼顾可靠性与存储效率



通过硬件的冗余磁盘阵列卡，将各个磁盘中的数据进行校验。如果磁盘出现在可以容忍的范围之内的失效，可以通过剩下的数据恢复出失效的数据。在恢复的时候，需要插入同样的磁盘，随后利用剩余的磁盘对新的磁盘上的数据进行恢复。

冗余磁盘阵列的分析

- 数据可靠性：可靠性不可调整，并且只能容忍一个磁盘或者两个磁盘的错误，在长期执行中不能容忍累积错误
- 数据自愈能力：数据自愈能力差。因为是固定的磁盘数据分布以及校验分布，数据恢复受限于新磁盘的接口带宽，数据恢复能力差
- 应用性能影响：在数据恢复的过程中，应用程序会因为得不到存储带宽而影响服务的可用性
- 存储效率：存储效率很高，只用不到25%~33%的额外存储空间，可以达到容忍至多两块磁盘的错误

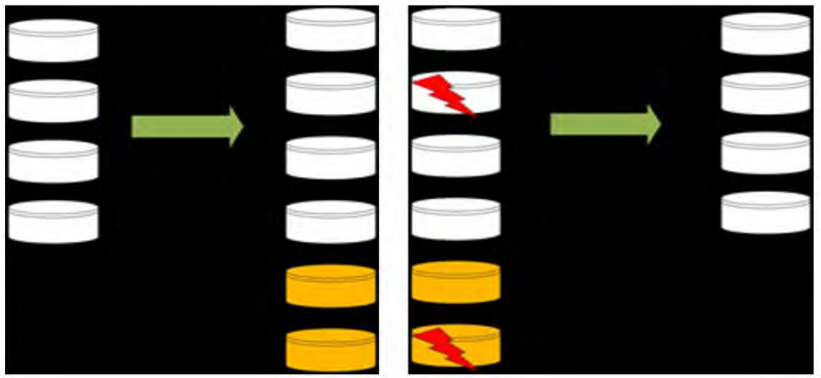
冗余磁盘阵列RAID由于其可靠性不足，自愈能力差，以及会影响前端应用，不适合直接应用于高容错自维护存储系统

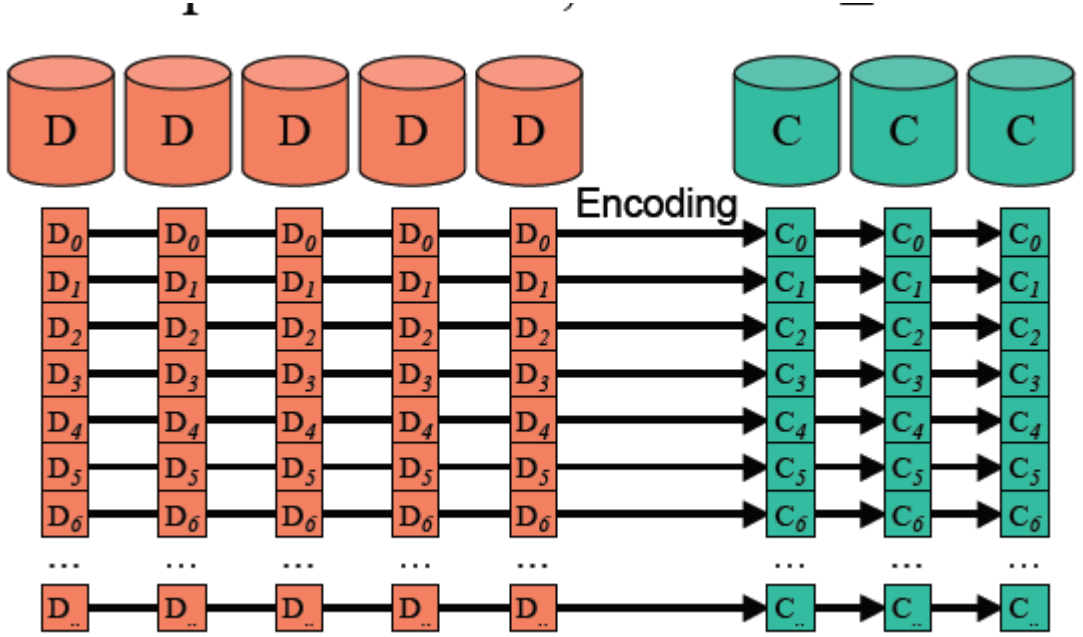
数据纠删码技术

- 基本思想：将通信中的数据纠错技术应用到存储系统中，使得在出现错误的时候，磁盘中的剩余数据可以恢复出缺失的数据
- 典型特征：对于存储的数据量以及存储的校验量可以进行灵活调整，因此可以在数据存储效率上得到提高
- 困难：最大的困难在于其实现比较困难，性能方面的限制是造成其大规模应用的最大障碍

数据纠删码技术的基本原理

数据纠删码技术分为两个部分，分别是数据编码以及数据解码。在写入数据的时候进行编码，在读出数据的时候进行解码。在出现错误的时候，通过剩余信息进行解码。

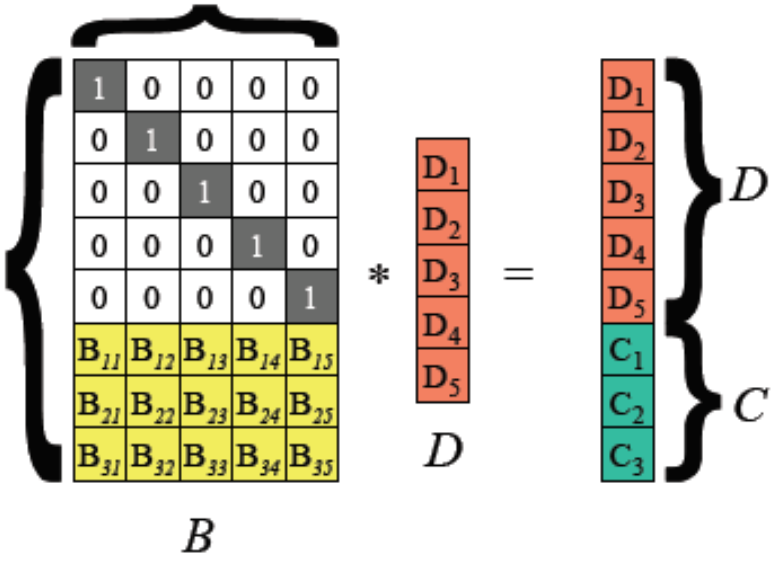


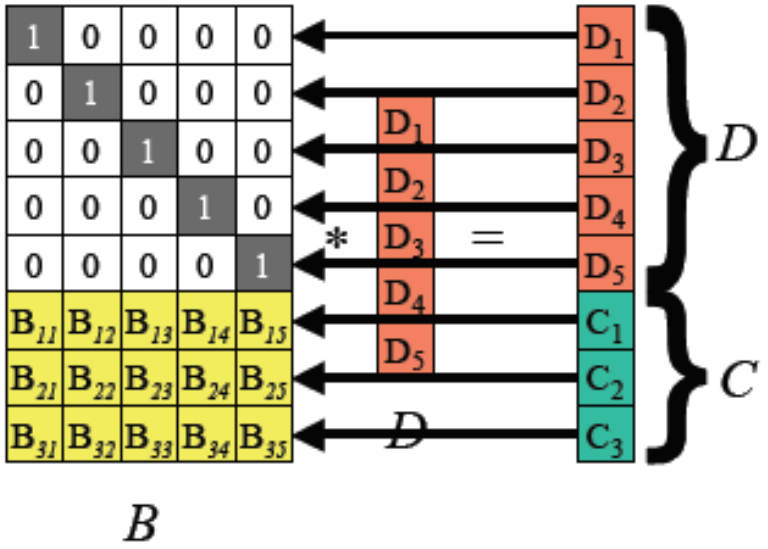


D 是Data Device ，用于保存数据，C 是校验位，用于重建恢复数据，C 是D 通过纠删码编码得到的，如最常见的纠删码Reed-Solomen 码

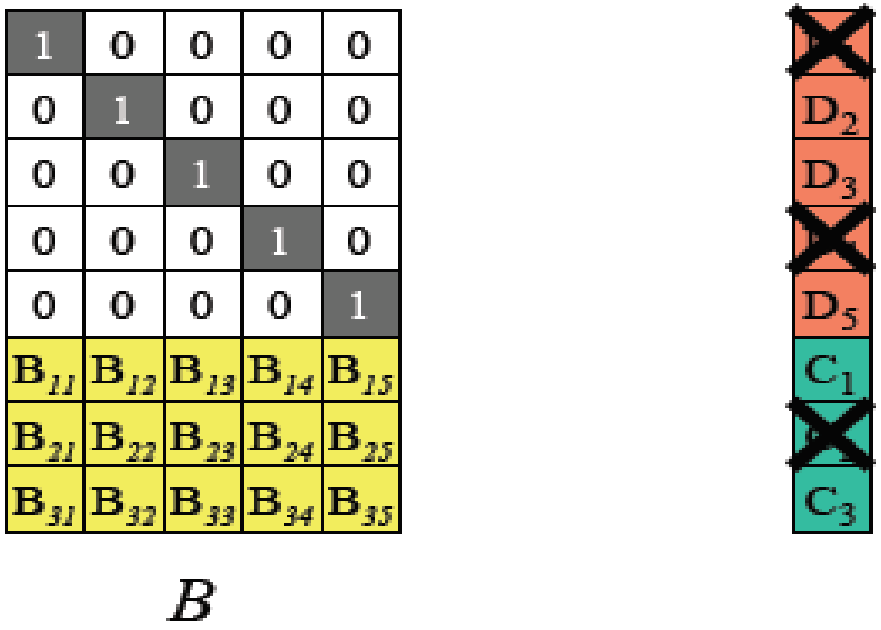
数据的编码方法

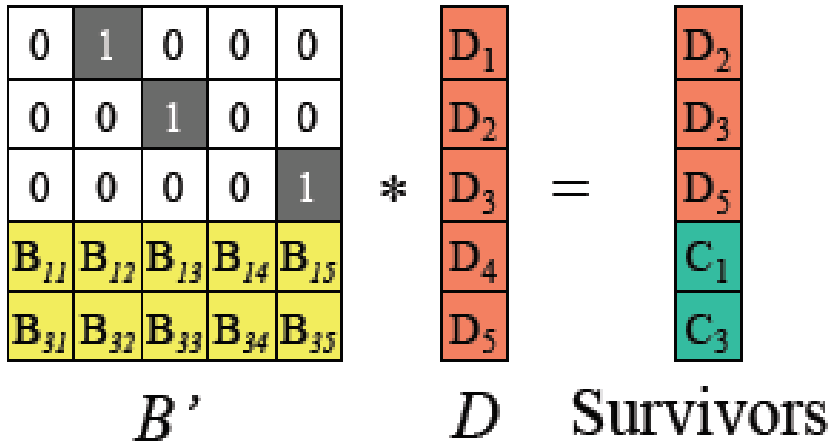
- B是生成矩阵
- B矩阵的两部分分别与数据矩阵D 相乘分别生成数据部分D 和校验部分C
- 这两部分分别是由 I×D 和 B^{-}×D 所生成的
- 数据在读取的时候只需要数据块即可，校验块可不参与读取过程





出现错误的剩余信息

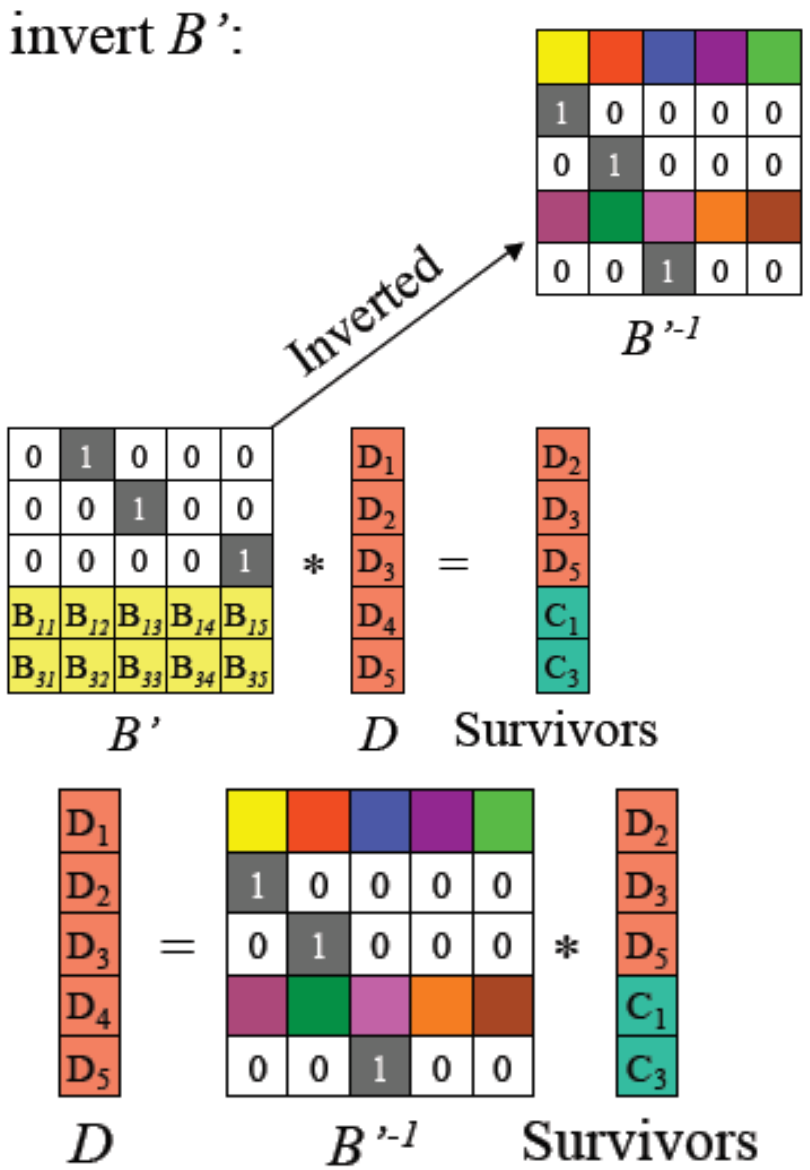




左图是出现3个磁盘错误的情况，磁盘正确的信息包括3个数据块和2个校验块。注意，这里的生成矩阵B是复制到系统的所有节点的，不会丢失

左图是去掉发生错误行后的生成矩阵B’，数据D和剩余磁盘信息S之间的乘积关系，依据这样的信息可以恢复出原始数据

数据的重构过程



对剩余的生成矩阵B’进行求逆运算
（如果损坏磁盘不是过多，只需要选择对应的方阵即可）

通过逆矩阵来获得原始数据

通过重构获得原始数据的条件：任何满足条件的剩余行组成的矩阵（方阵）必须是可逆的

可逆矩阵构造方法

$$Encode(D)=GD=\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ \hline x_0+y_0 & x_0+y_1 & x_0+y_2 & x_0+y_3 \\ \frac{1}{x_1+y_0} & \frac{1}{x_1+y_1} & \frac{1}{x_1+y_2} & \frac{1}{x_1+y_3} \\ \frac{1}{x_2+y_0} & \frac{1}{x_2+y_1} & \frac{1}{x_2+y_2} & \frac{1}{x_2+y_3} \end{vmatrix} \begin{vmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{vmatrix} = \begin{vmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ c_0 \\ c_1 \\ c_2 \end{vmatrix}$$

柯西矩阵

$$\begin{bmatrix} f_{1,1} & f_{1,2} & \cdots & f_{1,n} \\ f_{2,1} & f_{2,2} & \cdots & f_{2,n} \\ f_{3,1} & f_{3,2} & \cdots & f_{3,n} \\ \vdots & \vdots & \vdots & \vdots \\ f_{m,1} & f_{m,2} & \cdots & f_{m,n} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix} =$$

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & n \\ 1 & 4 & \cdots & n^2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 2^{m-1} & \cdots & n^{m-1} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_m \end{bmatrix}$$

范德蒙特矩阵

这两个矩阵的任何子方阵都是可逆矩阵，因此可以作为纠删码的生成矩阵

有限域上的运算

- 柯西矩阵和范德蒙特矩阵的计算条件是任何的数域，但是计算机中的数据是整数，整数的普通加法和乘法运算只能构成整环
- 另外，整数的大小是有上界的，但是计算机中的整数有一定的取值范围的限制
- 为了使得上述的计算能够成功，必须要在有限域中进行，因此在纠删码中，通常使用的是GF(2^8)上的有限域上的运算。这里使用2^8而不是2^{16}或者2^{32}的原因是后面两个在处理器上实现的代价太大，不实用
- 有限域GF(2^8)上加法为异或运算
- 有限域GF(2^8)上乘法为移位和异或运算，是通过生成矩阵方法来获得的乘法运算，最终会被归结为多个异或的运算

纠删码技术的分析

- 数据可靠性：可靠性可以灵活调整，可以通过调整校验块的数目获得非常高的可靠性
- 数据自愈能力：如果校验数据固定在某一个磁盘中，性能机会被限制，而纠删码通过软件方式实现，有其灵活性
- 应用性能影响：如果调整可靠性级别很高，则出现少数错误的时候无需立即进行自愈，减少对应用的影响
- 存储效率：通过调整纠删码的参数来获得高的存储效率

数据纠删码的方式可以提供很高的可靠性，并且其参数可调，具有灵活性。高容错自维护存储系统的最大工作是提高数据自愈能力，这就需要合理的数据块组织方式

现有可靠性技术的对比

高容错自维护存储系统的特性	副本技术	RAID硬件可靠性技术	数据纠删码技术
数据可靠性	良：需要大量节点	中：容忍1~2个错误	优：可获高可靠
数据自愈能力	优：需要大量节点	差：磁盘带宽限制	良：依赖数据管理
应用性能影响	优：需要大量节点	差：磁盘带宽限制	良：依赖数据管理
磁盘存储效率	差：效率过低	优：额外空间少	优：可获高效率
对于计算要求	优：无需计算	优：计算简单	良：计算量大

没有单项技术可以完全达到高容错自维护存储系统的目的。可以选择数据纠删码技术作为高容错自维护存储系统的可靠性基础。并围绕这个基础，克服其弱点：

1）通过浮动数据块组织提高自愈能力
2）通过浮动数据块组织降低对应用影响
3）通过并行化提高计算速度

提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

四.高容错自维护存储系统特征

五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

六.高容错自维护存储系统TStor设计

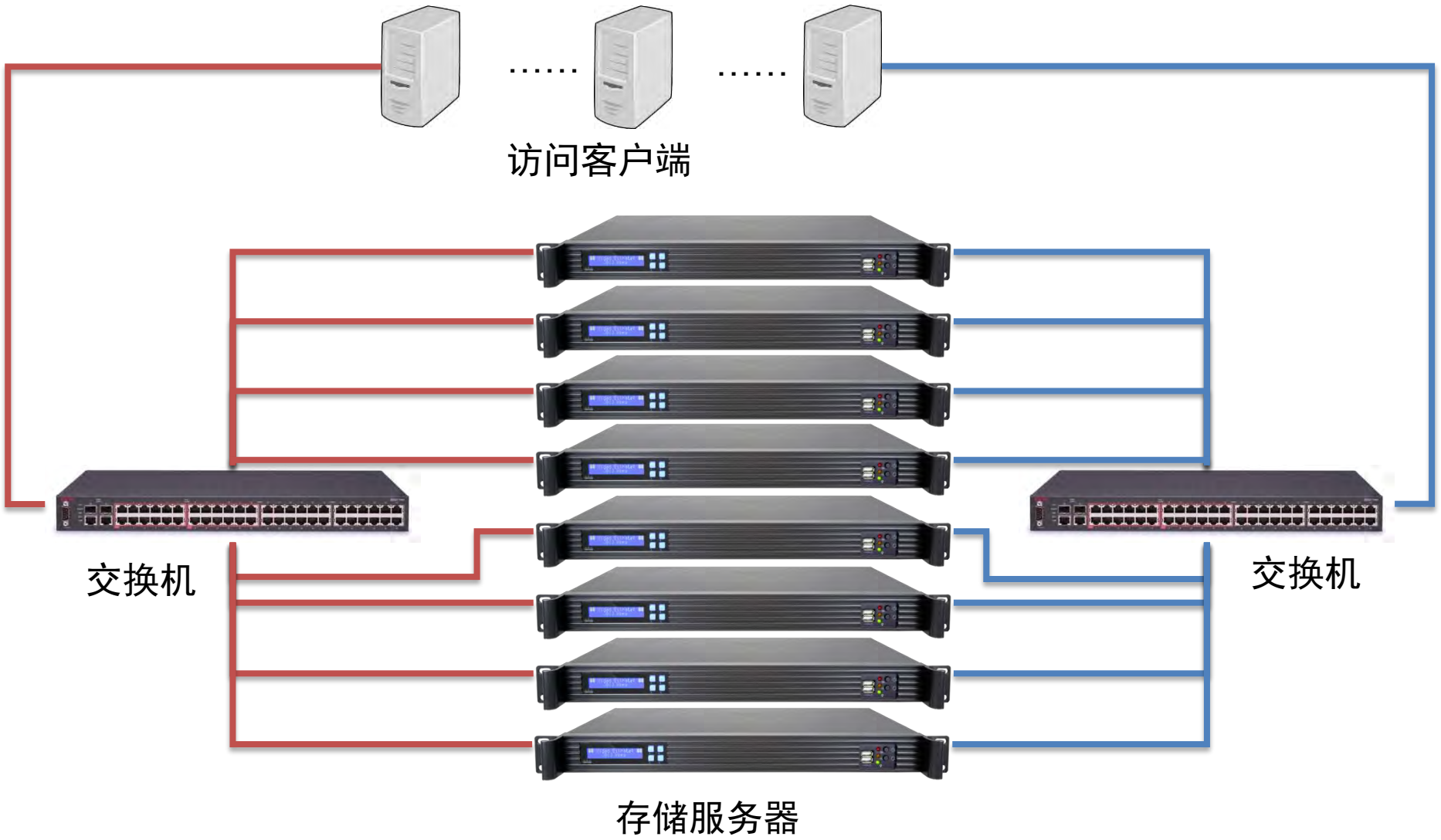
高容错自维护存储系统三个关键技术：

- 硬件冗余：通过冗余的硬件环境，避自硬件单点错误造成系统不可用
- 大规模纠删码：通过大规模的数据编解码算法，提供数据高可靠性以及提高磁盘的存储效率，编码和解码的计算量大，需要加速
- 浮动数据块组织：通过浮动数据块组织技术，支持数据块的快速自愈，降低对于前端应用的影响

硬件冗余：高容错系统的冗余硬件结构

- 避自网络单点：使用两个独立的网络交换机将服务器连接在一起
- 避自服务器单点：多个独立服务器通过网络连接在一起
- 避自磁盘单点：每一个服务器中包含12或者16块硬盘，这样的配置是标准的商业配置，在出现错误的时候有足够多的磁盘作为后备
- 在硬件上做到冗余，以应对不同的模块的失效
  - ✓ 硬盘失效的概率为每年5%，其它所有的模块都是无机械结构的集成电路，各个部分失效的概率小于每年1%。节点已经有多个，磁盘也有多个，因此只需要提供额外的一个交换机就可以将出错的概率降低

硬件冗余：硬件系统连接图



大规模纠删码：大规模纠删码编解码算法

- 通过纠删码的方法达到尽可能高的可靠性
- 在进行32+16编码计算的时候，处理器将成为计算的瓶颈
- 需要通过多种方法对编解码进行加速，编码加速能够提高数据写入，数据读取以及数据恢复的速度

使用32+16大规模编解码的原因：①提高数据可靠性；②避自数据修复的时候新的磁盘错误造成数据丢失；③延长自愈时间，避自对前端应用造成影响

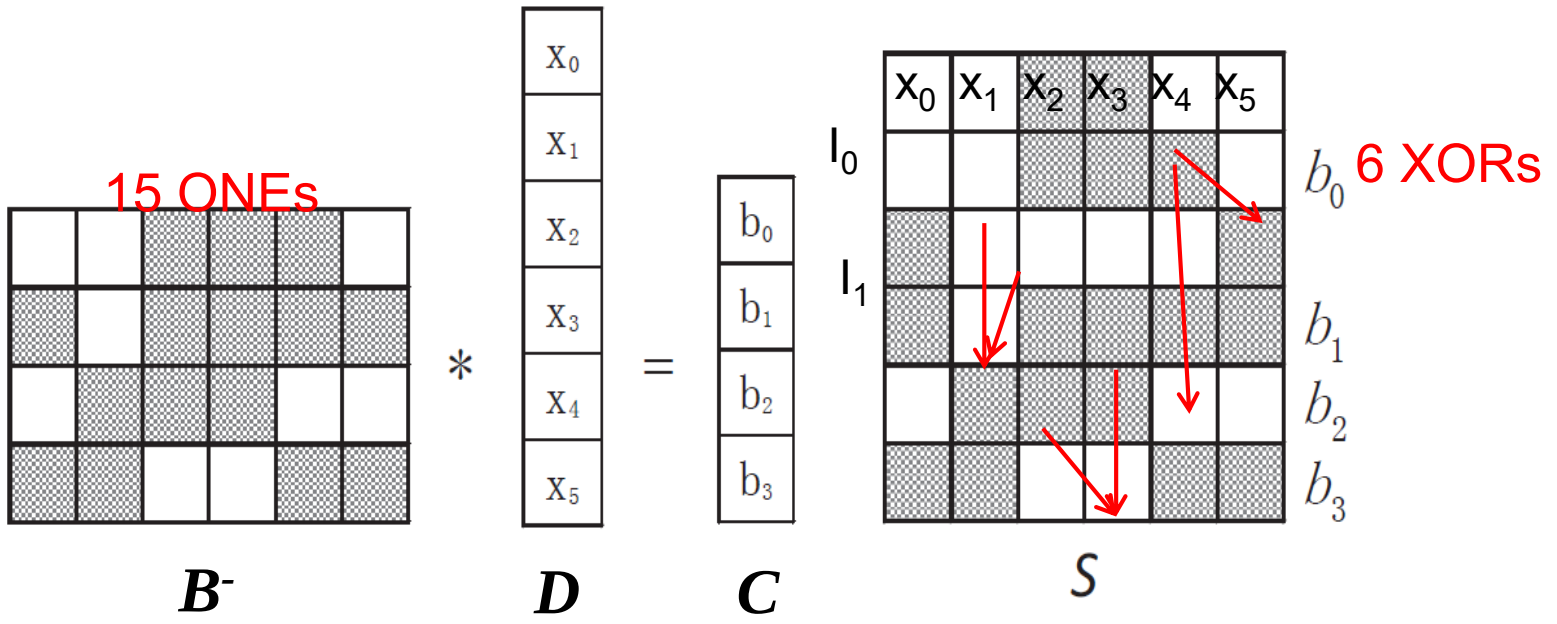
大规模纠删码：大规模编解码的性能挑战

- 涉及到有限域上的大规模运算，需要很大的计算能力，需要提高计算性能，否则影响存储性能
- 方法1：通过对计算过程进行分析，减少实现过程中的计算量
- 方法2：通过多线程并行计算，充分利用多核的计算能力
- 方法3：通过使用处理器的向量指令，提高单线程的编解码计算能力
- 方法4：通过计算、传输，存储的存算重叠，充分利用所有的硬件资源，提高磁盘的并行写入速度

上述这些优化方法都是不矛盾的，可以叠加使用

大规模纠删码：纠删码的计算量减少方法

- 根据纠删码的冗余配置，构建轻量的纠删码生成矩阵，使得矩阵中“1”的个数较少
- 针对该生成矩阵计算公共异或和，通过最大程度利用公共和，设计异或操作的最短调度
- 本方法能够针对每种冗余度配置参数，给出目前技术水平最优的编码方法。



大规模纠删码计算减少效果

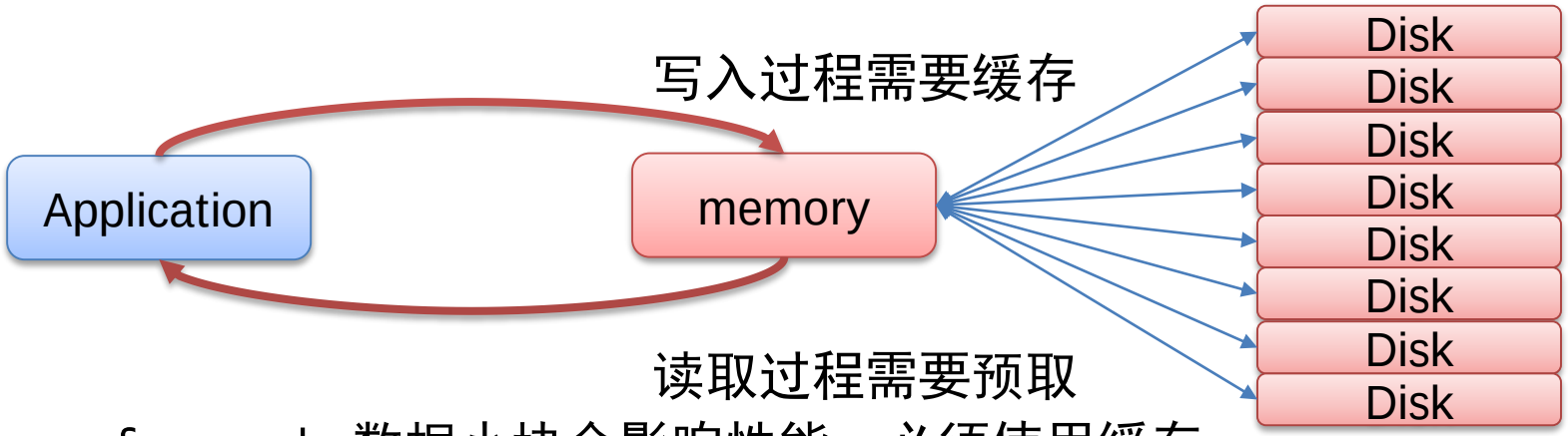
- "CaCo: An Efficient Cauchy Coding Approach for Cloud Storage Systems," IEEE Transactions on Computers, Volume: 65, Issue: 2, Page(s): 435 - 447. Feb. 2016.
- 审稿专家认为：Many storage providers can benefit from results of this work.
- 云存储系统数据高效编码方法, 专利号：ZL201310278650.8


发 明 专 利 证 书


CaCo: An Efficient Cauchy Coding Approach for Cloud Storage Systems

大规模纠删码：数据编解码技术的并行化

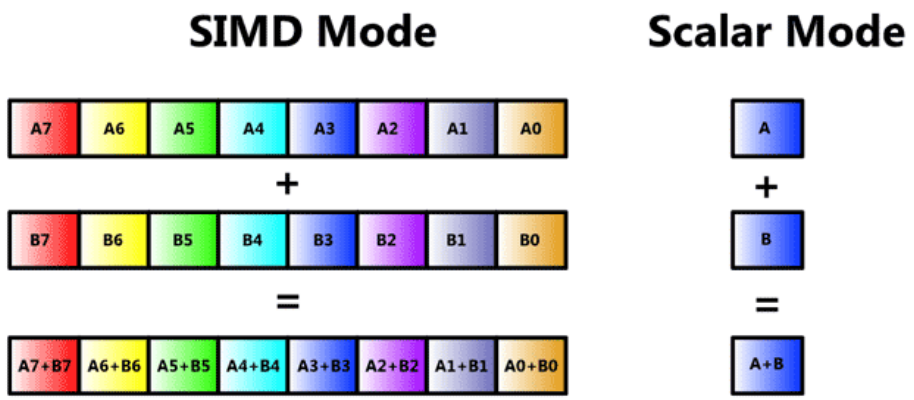
- 数据纠删码技术在生成矩阵以及计算过程中，都具有天然并行化的特性
  - ✓ 编码：共享只读的输入数据，并行计算校验块
  - ✓ 恢复：共享只读的剩余数据，并行构造出原始数据
- 为了达到更高的性能，必须要有足够的数据才能够将计算过程真正并行起来，提高性能
  - ✓ 在写入的时候需要加入数据批处理与缓存的工作，读取的时候需要进行数据预取

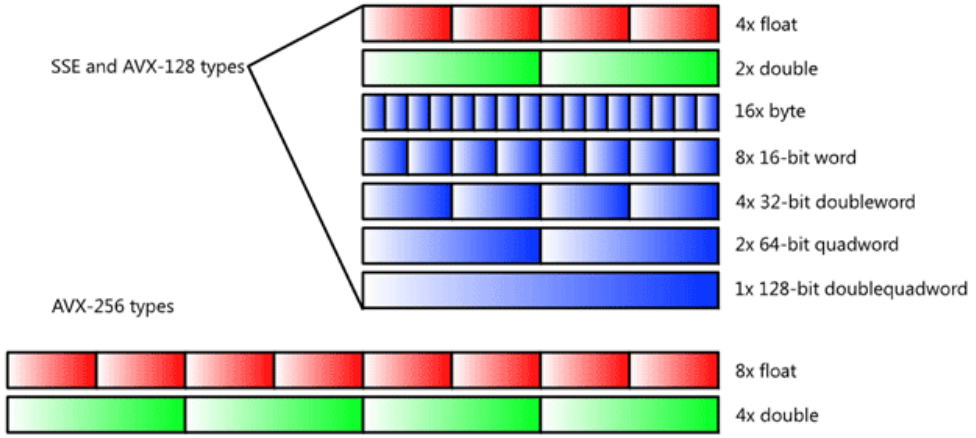


fuse，nfs，samba数据小块会影响性能，必须使用缓存

大规模纠删码：使用向量指令提高计算能力

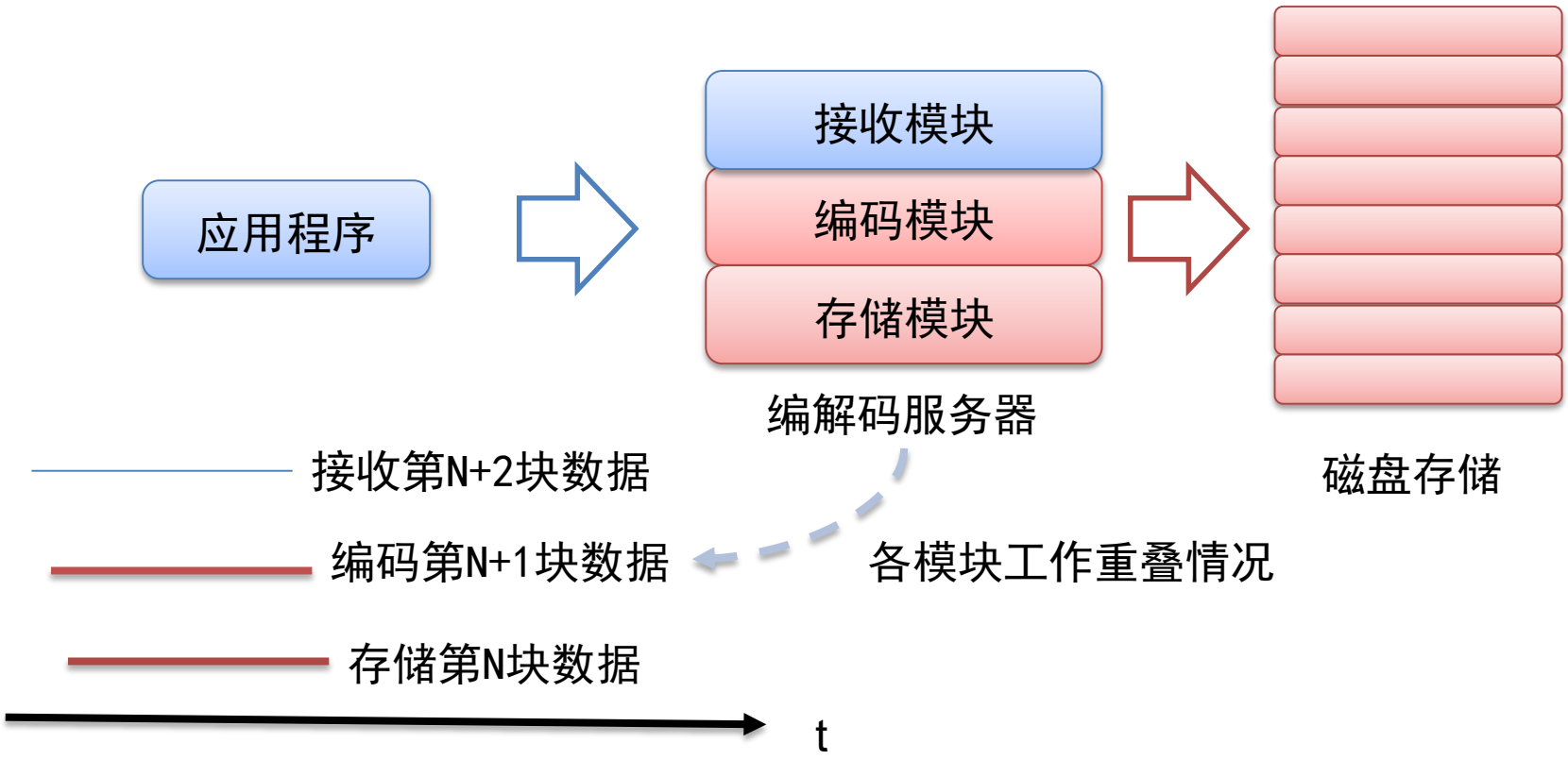
- 向量指令一次可以同时处理多个数据的计算
- Intel的向量指令包括最新的AVX-2指令，一次可以处理32个字节的计算
- 通过向量指令，即使在单线程的条件下，数据编解码的能力也可以成倍提高





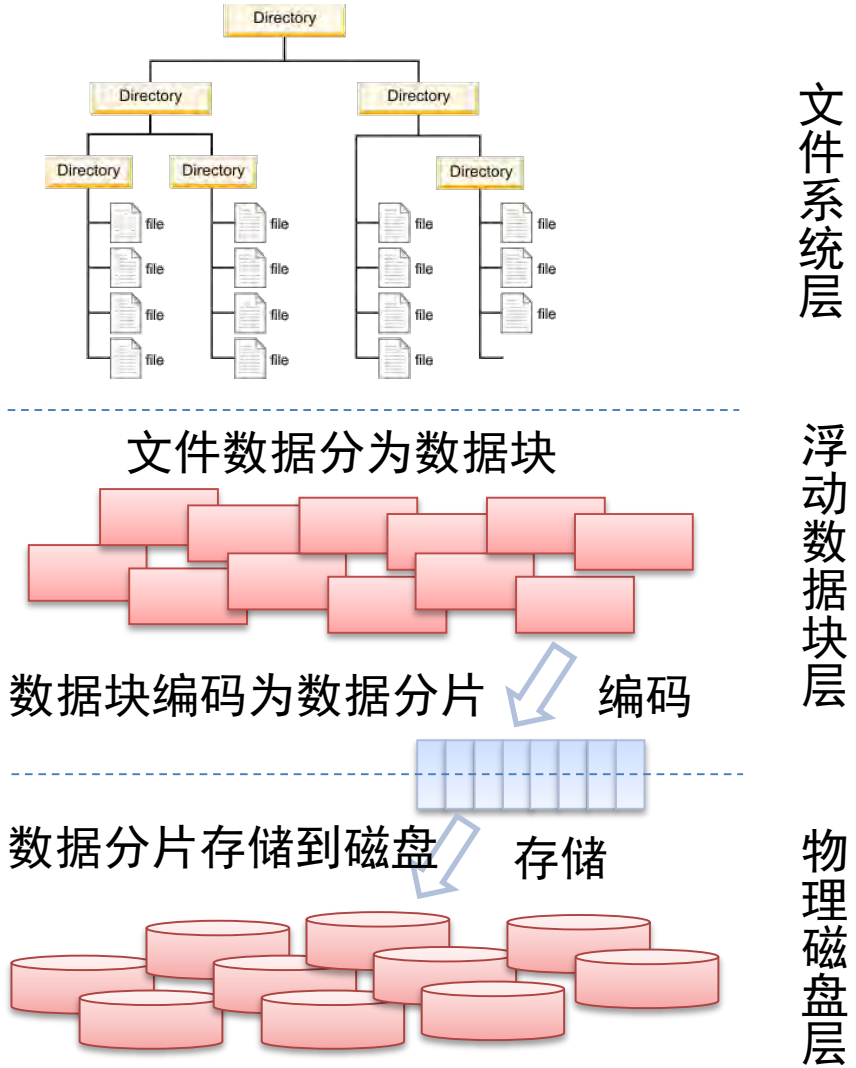
大规模纠删码：计算，传输与存储重叠

- 采用计算，传输，存储重叠的方法，在计算结果完成之后，立刻传入到后台存储，这样可以将所有的硬件资源在同时利用起来



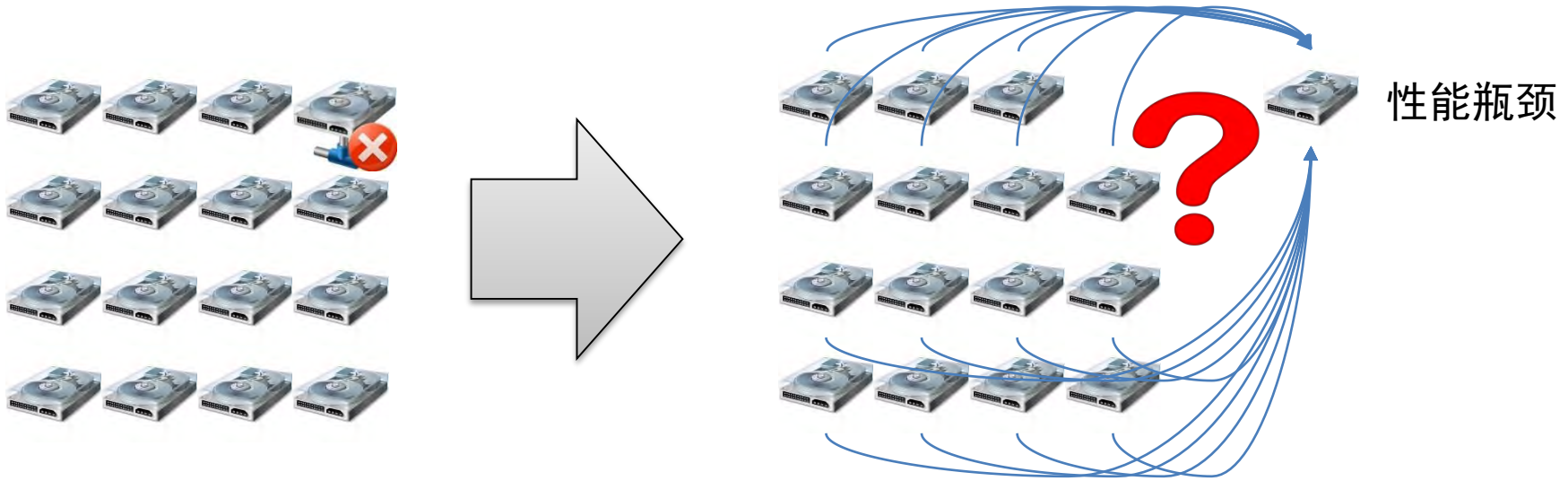
浮动数据块组织

- 文件数据以数据块的形式进行保存
- 数据块通过纠删码的形式编码为数据分片
- 数据分片被数据块服务器保存到物理磁盘中
- 同一个数据块的数据分片分布到不同的磁盘
- 高容错自维护存储系统形成物理磁盘（保存数据分片），浮动数据块，文件系统三个层次



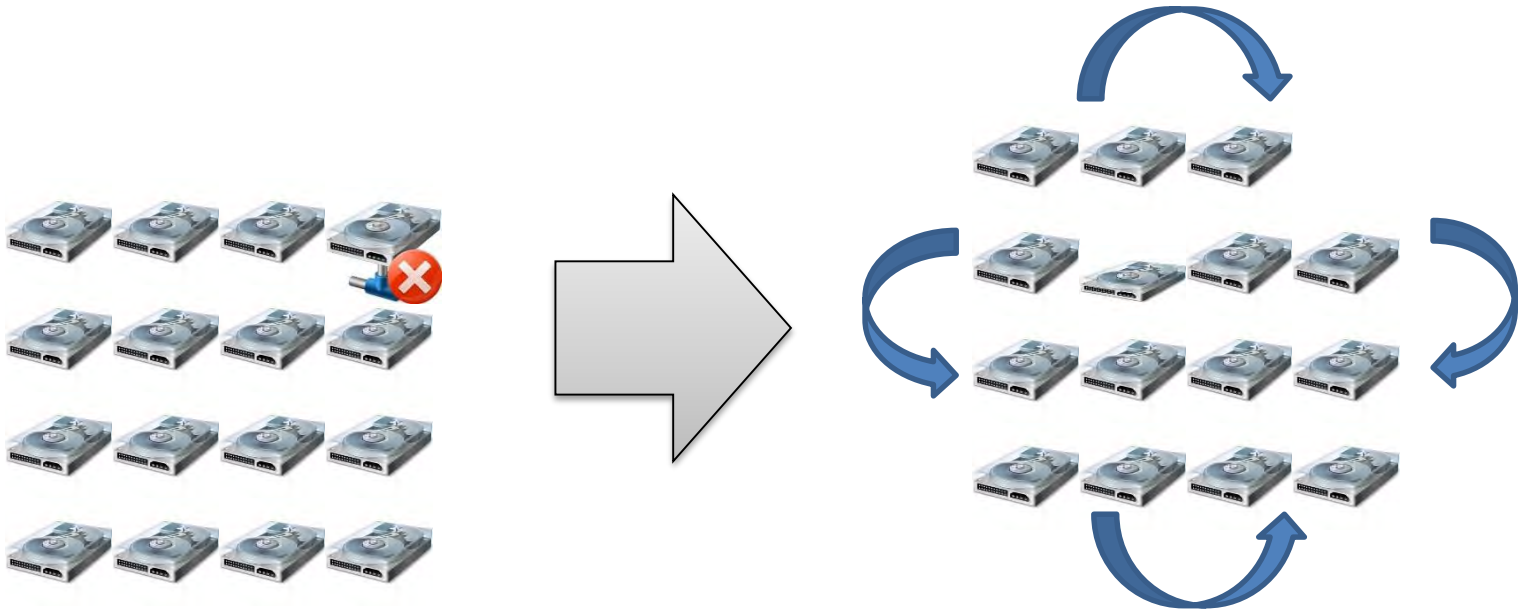
浮动数据块组织：支持快速自愈，提高应用性能

- 数据自愈重建工作不能只发生在一块磁盘上，单块磁盘有性能瓶颈问题；数据自愈重建工作也不能只发生在一个节点之中，单个节点会有网络性能瓶颈
- 数据重建需要所有活动节点，活动磁盘的参与，使用所有的物理资源进行自愈
- 核心方法：浮动数据块组织



浮动数据块组织：数据自愈能力分析

- 一块磁盘失效，会导致磁盘上的数据块分片失效
- 剩余的磁盘通过重构，恢复出原始的数据块，数据块恢复是并行的
- 原始数据块可以重构出缺失的数据分片，不同的缺失的数据分片被并行写入到所有的磁盘，利用所有磁盘的性能
- 恢复能力正比于磁盘的聚合带宽



每一块磁盘都参与恢复

提纲

一.高容错自维护存储系统定义

二.高容错自维护存储系统意义

三.高容错自维护存储系统目标

四.高容错自维护存储系统特征

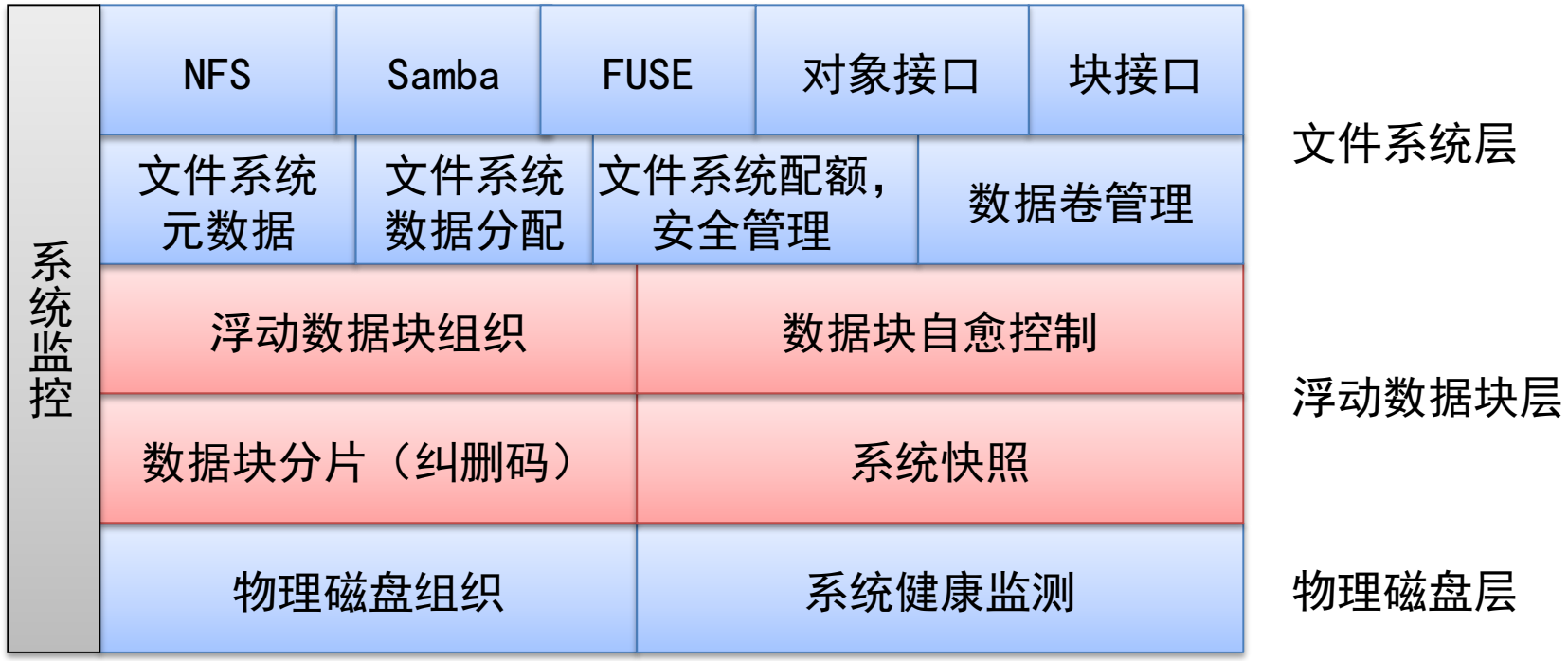
五.现有存储系统的可靠性保证技术以及分析

六.高容错自维护存储系统TStor设计

七.高容错自维护存储系统TStor实现

七.高容错自维护存储系统TStor实现

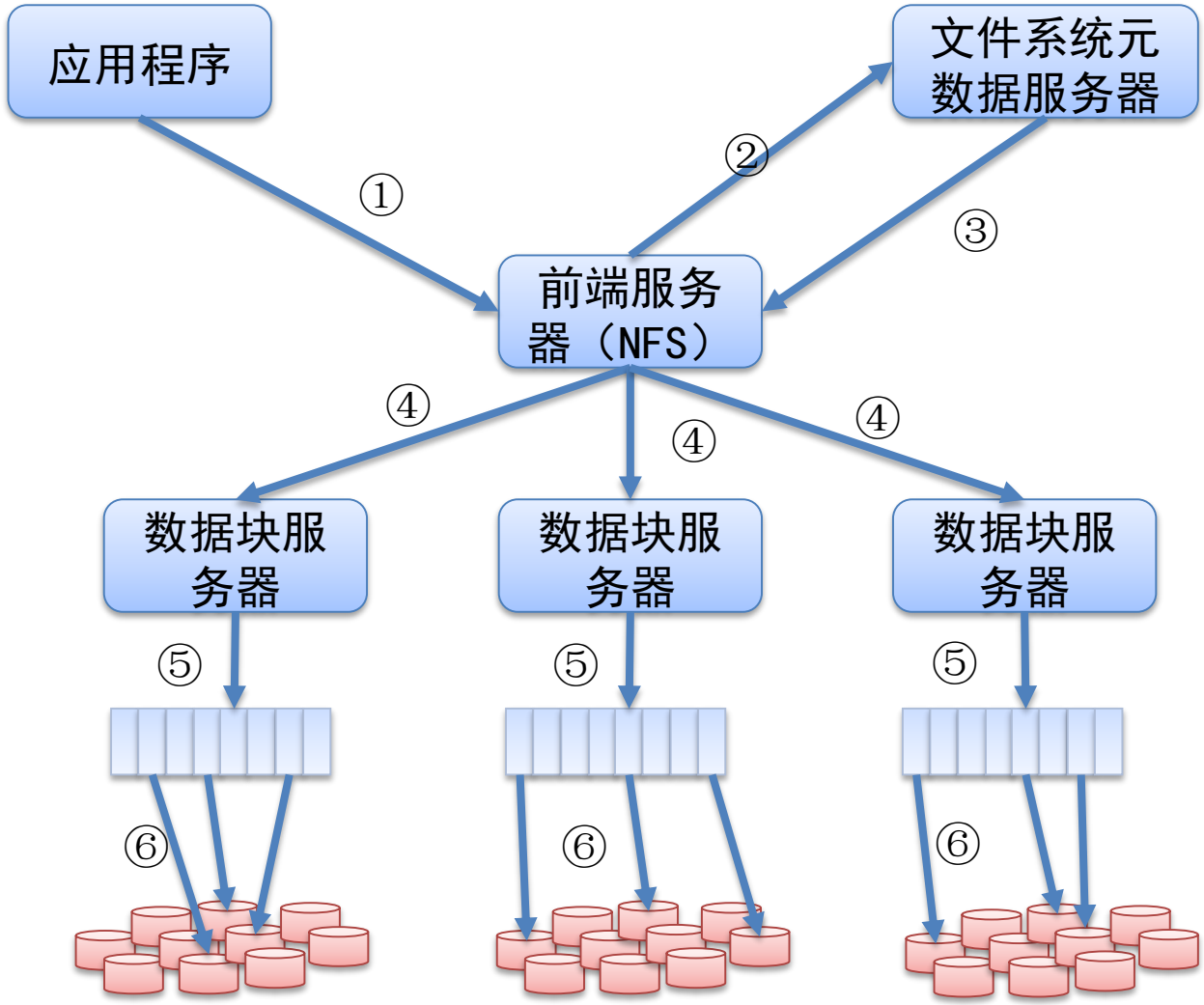
TStor系统架构



文件系统实现

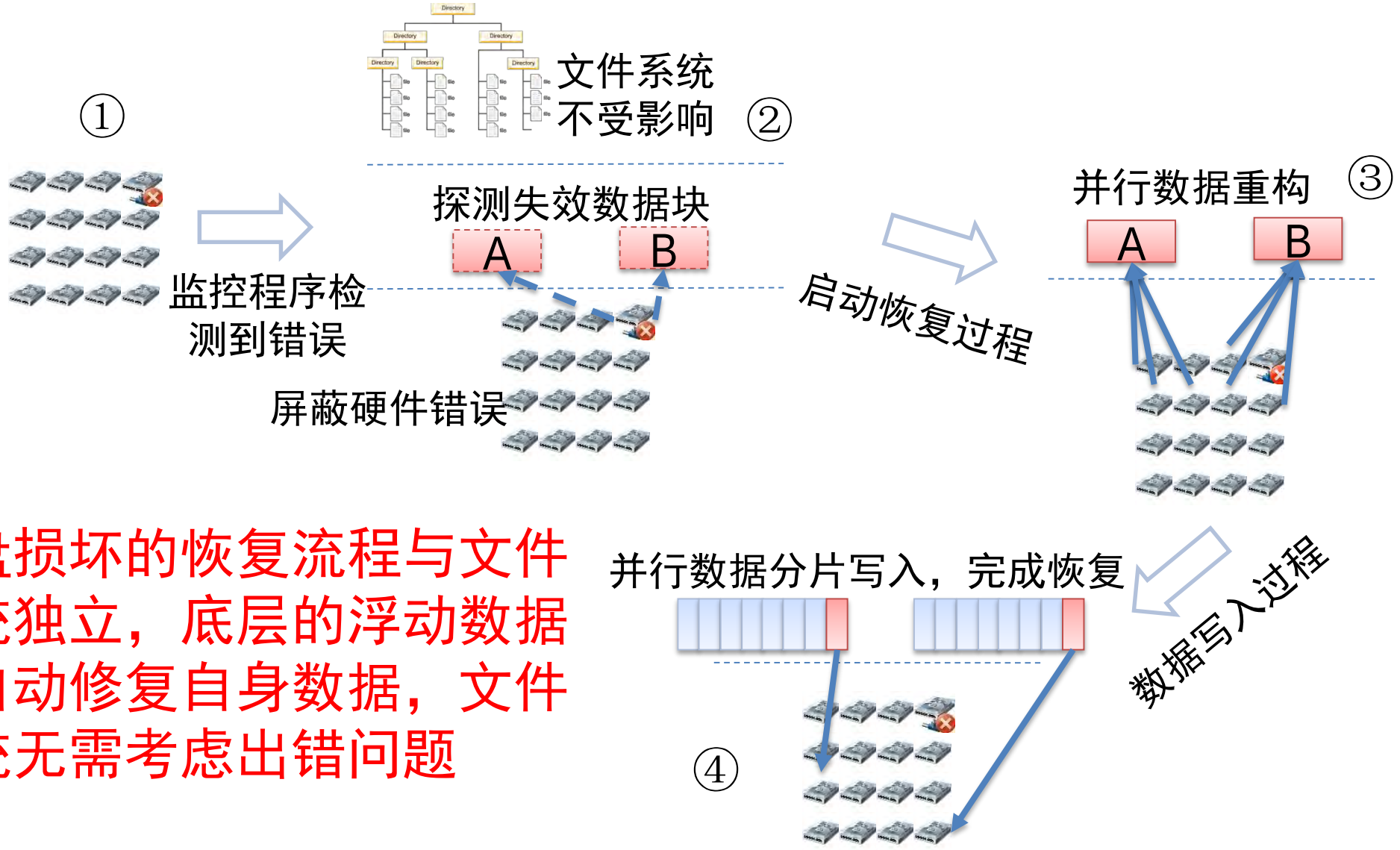
- 元数据保存：元数据也以浮动数据块的形式保存，并以数据分片分布在系统的磁盘中
- 元数据信息：文件系统元数据记录了目录树的信息，目录中包含的所有文件，以及每一个文件的大小，访问时间等信息
- 文件数据块的服务器定位：文件数据的数据块依据文件名以及数据块的顺序通过哈希值计算出来
- 磁盘错误处理：文件系统不必关心，底层浮动数据块自动屏蔽磁盘的错误，恢复数据块
- 节点错误处理：通过分布式共识算法Raft来选择一台服务器作为元数据服务器，读取元数据块，提供元数据服务；同样选择一台服务器作为特定浮动数据块的管理节点

文件系统写入流程(NFS写入)



① 数据写入前端服务器
② 前端服务器询问数据块服务器信息
③ 元数据服务器返回管理的数据块服务器信息
④ 前端服务器分发数据块到不同的数据块服务器
⑤ 数据块服务器完成数据编码
⑥ 编码后的数据分片写入到磁盘

系统功能执行流程（数据恢复）



磁盘损坏的恢复流程与文件系统独立，底层的浮动数据块自动修复自身数据，文件系统无需考虑出错问题

TStor系统物理参数配置

- 存储节点数目：16个
- 处理器：双路志强E5 2643 v4 3.1GHz
- 内存：64G DDR4单节点内存
- 系统磁盘：单个节点2个480G SSD 系统盘RAID1镜像
- 数据磁盘：单个节点12块8T 7.2krpm SAS 磁盘
- 网络：56G InfiniBand网络,交换机
- 磁盘的总容量：1.5PB，可用容量为1PB
- 磁盘数目：192块数据盘
- 支持访问接口：对象接口，FUSE接口，NFS接口等
- 系统参数：使用32+16的编解码方式，提高极高的数据可靠性

正面

背面

系统的运行与测试

大规模数据编解码的性能测试

四核八线程	编码	解码
64 + 32	2.1GB/s	1.9GB/s
32 + 16	3.9GB/s	3.7GB/s
16 + 8	5GB/s	5.2GB/s
8 + 4	5.3GB/s	5.2GB/s

快速的编解码完全满足存储系统需求

单个客户端节点的数据读写能力

读写方式	对象写入	对象读取	NFS写入	NFS读取
32 + 16	1.2GB/s	969MB/s	880MB/s	845MB/s
16 + 8	1.3GB/s	775MB/s	997MB/s	707MB/s
8 + 4	1.4GB/s	702MB/s	882MB/s	655MB/s

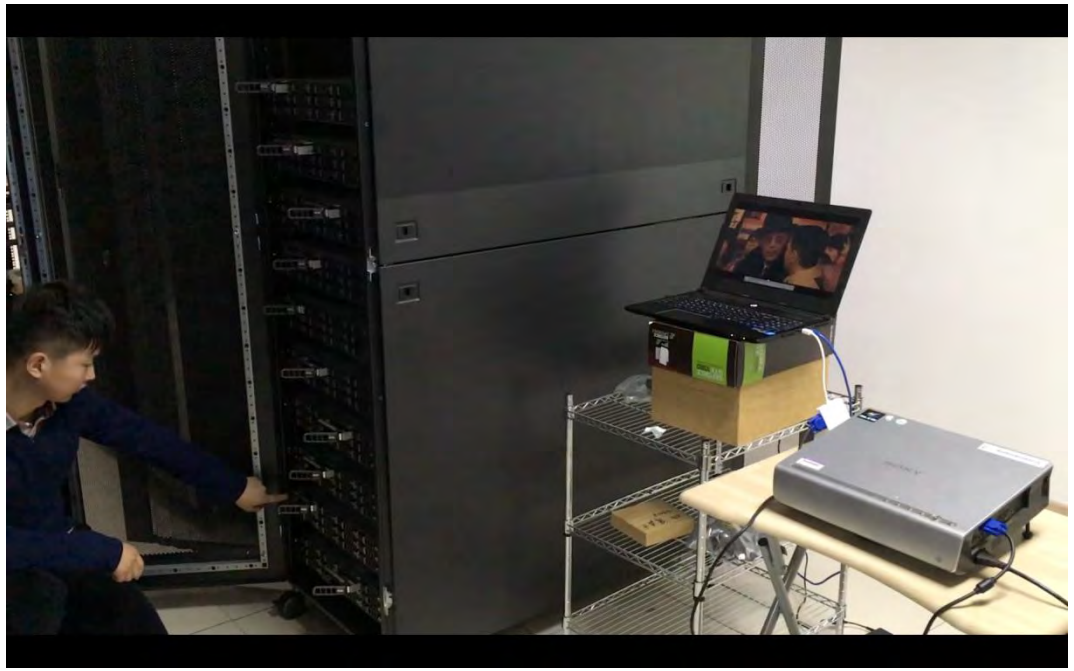
数据读写能力高，能够满足应用性能需求

聚合写入的能力为7.6GB/s，聚合读处的能力为9.3GB/s

浮动数据块组织的数据恢复能力：1.18GB/s，恢复一个8T磁盘数据，花费时间不到2小时（速度受纠删码与192块磁盘的限制），作为对比单盘恢复需要花费近15小时

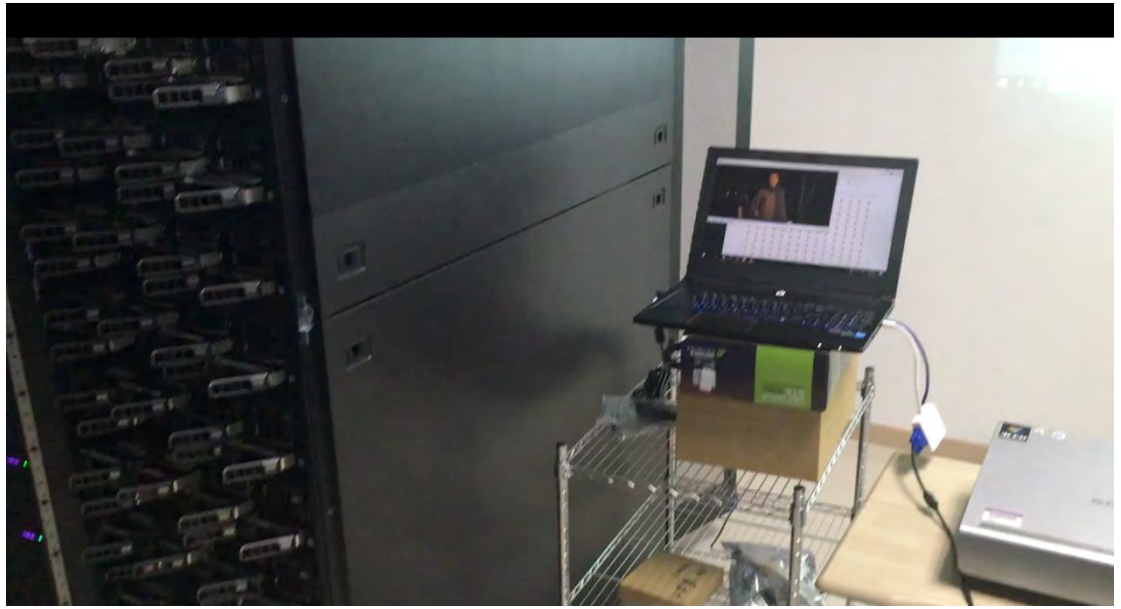
数据自愈块，无需担心恢复期间的应用性能问题

系统的可靠性和可用性测试



通过关机和拔硬盘的方式来测试系统的可靠性与可用性

同时随机拔掉16块磁盘；或者在同时出现两个节点服务器错误的时候，数据不会丢失，系统能够继续提供服务

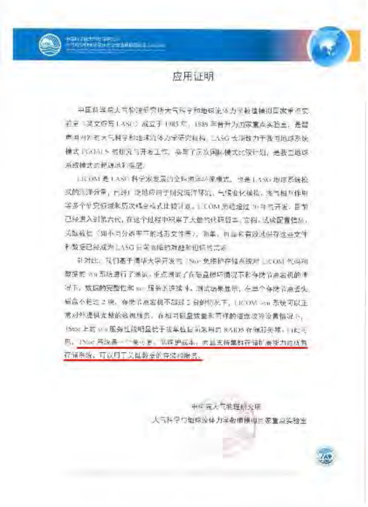


与现有的存储产品比较

- 华为的OceanStor
- InfiniDAT的InfiniBox

	InfiniBox F2000	OceanStor 9000	TStor 当前	比较要点
容量参数	磁盘总容量1PB 可存储数据容量500TB	容量可以定制扩展，最大 1PB ~ 100PB，可用存储容量与编码相关	磁盘总容量1.5PB 可存储数据容量1PB	InfiniBox只有50.0%的利用率，TStor有66.7%的利用率。
性能参数	聚合写入能力7GB/s	288个节点，40PB的容量，55TB的全局缓存，吞吐量为200GB/S。（单节点实际少于700MB/s）	单个节点持续能力1.2GB/s，最高1.6GB/s，聚合带宽7.6GB/s	三个系统都可以顺利对带宽进行聚合。在实际的读写性能特征中，TStor中的性能速度与现有的存储系统性能相当。前两个系统都使用了SSD，NVRAM等方法进行硬件加速，提高了成本。在TStor中使用并行化的方法进行软件加速，充分利用商用硬件中的处理性能。
可靠性级别	使用硬件容忍同时2个部件的错误	可以容忍同时4个磁盘错误或者一个节点错误	可以容忍同一一个文件16个磁盘错误，或者同时10个磁盘错误和两个节点错误	在TStor中由于采用了大规模的数据存储编解码方法，使得容错级别大大提高，可以容忍更多磁盘以及节点的错误，降低修复的频率。

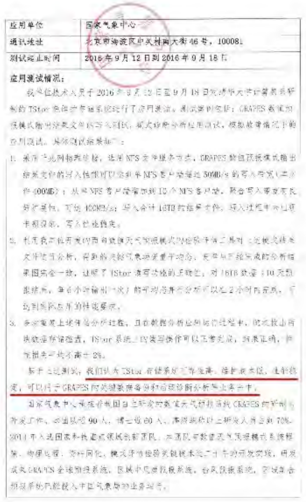
用户使用情况


应用证明


中国地质大学
China University of Geosciences
应用证明


清华大学免维护存储系统 TStor 应用证明


应用证明




应用证明


用户证明
ENVISION


ENVISION

国产处理器和国产操作系统的支持

- 国产的处理器：TStor系统正常运行在申威（1610 16核），龙芯（3B1500 8核）和飞腾处理器上
- 国产的操作系统：TStor系统正常运行在麒麟操作系统上
- 在国产处理器和操作系统上的存储系统性能正在进一步优化
- 国产麒麟操作系统对于InfiniBand支持正在进一步工作中

总结

在国际上首次实现了基于32+16纠删码、能满足实际应用需求的高容错自维护存储系统。

- 1. 首次提出了一种纠删码计算量削减方法，针对任意给定的纠删码配置参数，能够给出高效的生成矩阵和计算调度，解决了编码过程中计算量过大、计算重复的问题。与微软研究院开发的纠删码计算方法相比，编码和解码计算量分别减少了40%和52%。
- 2. 提出了一种大规模纠删码的并行实现方法，通过单线程内部的向量指令、多线程并行计算以及计算、传输、存储相重叠的方法，充分利用系统的硬件资源，提高了数据的并行读写速度。
- 3. 首次提出了一种支持大规模纠删码的数据快速自愈的浮动数据块组织方法，通过物理磁盘、浮动数据块和文件系统三个层次，使恢复能力正比于磁盘的聚合带宽。通过典型应用的测试，恢复过程对应用程序的性能影响在2%~5%以内。

请各位专家批评指正！谢谢！