



# Weaver Codes

Ohad Rodeh



# Introduction

- This lecture is based on: **WEAVER Codes: Highly Fault-Tolerant Erasure Codes for Storage Systems**. Jim Hafner, IBM Research. USENIX Conference on File and Storage Technologies (FAST), December 2005.
- This presentation is based on original material by Jim Hafner.
- The actual Weaver codes are more general than presented here. We describe only the case where the parity in-degree is equal to the data out-degree.



# Motivation

---

- The probability of losing data increases as:
  - The amount of data increases
  - Bit-error rates on disks remain constant
- Furthermore:
  - SATA (low cost/less reliable) drives
  - Rebuild times are longer
- There is no optimal code. Each code uses different tradeoffs.



# Design principals I

- RAID-{1, 4, 5}, RAID-DP, are all horizontal codes; in a single stripe the data is situated on data disks, the parity on parity-disks.
- Weaver is a vertical code: data and parity are located on the same device



# Design principals II

- Constrained Parity In-degree:
  - Fixed independent of stripe size
  - Bounded by fault tolerance
- Balance and symmetry
  - Parity in-degree and out-degree are constant, equal to fault tolerance.
  - Defined with rotational symmetry

# Two fault tolerant case

- The code for the two fault tolerant case:

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{12}$	$p_{23}$	$p_{34}$	$p_{45}$	$p_{50}$	$p_{01}$

- $p_{ij} = d_i \oplus d_j$
- The code uses a weaving pattern
- The parity in-degree and data out-degree are 2



# Two fault tolerant case, II

- The code works for any  $N \geq 4$
- Efficiency is 50%.
- Rotationally symmetric, balanced, simple, expandable.

# Proof of 2 fault-tolerance, I

- Suppose  $S_1$  is lost

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{12}$	$p_{23}$	$p_{34}$	$p_{45}$	$p_{50}$	$p_{01}$

- Rebuild  $d_1$  using only  $p_{12}$  and  $d_2$

# Proof of 2 fault-tolerance, II

- Suppose  $S_2$  and  $S_3$  are lost

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{12}$	$p_{23}$	$p_{34}$	$p_{45}$	$p_{50}$	$p_{01}$

- Rebuild  $d_2$  using  $p_{12}$  and  $d_1$
- Rebuild  $d_3$  using  $p_{23}$  and (new)  $d_2$
- Only two IO reads for three elements

# Proof of 2 fault-tolerance, III

- Suppose  $S_1$  and  $S_4$  are lost

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{12}$	$p_{23}$	$p_{34}$	$p_{45}$	$p_{50}$	$p_{01}$

- We can rebuild independently and in-parallel:
  - Rebuild  $d_1$  using  $p_{12}$  and  $d_2$
  - Rebuild  $d_4$  using  $p_{45}$  and  $d_5$
- This is an example of the localization property.



# Features 1

---

- Symmetry and balance
  - Simple logical to physical addressing
  - Natural load balancing
  - No need to rotate parity blocks like in RAID-5
- Localization: many use cases affect only subset of stripe
  - Recovery of lost strip or strips
  - Write Lock Zones
  - Add/delete devices



# Features 2

- Short write IOs: For  $FT=2$ , can update touching only 5 disks
- Improved IO costs for multiple strip writes
- The block-size can be very large in a Weaver code. For example, if the block size is:
  - 256B: then a long write will touch all the devices in a stripe
  - 256KB: then writes up to 256KB will touch only 5 devices ( $FT=2$ )

# Parametrization

- Since the code is symmetric it is sufficient to define the parity set for  $S_1$ .
- In addition, sometimes an offset is needed.
- Formally:
  - Each parity block  $p_j$  is defined by the set  $K_j$ :
$$p_j = \bigoplus_{d_i \in K_j} d_i$$
  - Due to rotational symmetry  $K_j = K_0 + j \pmod n$
  - Set  $1 \in K_0^1$  and use an offset  $s$ ;  $K_0 = K_0^1 + s$
- Summary: In order to define a Weaver code, it is sufficient to describe  $K_0^1$  and  $s$

# Three fault tolerant case

- Let's try  $K_0^1 = \{1, 2, 3\}$  and  $s = 0$

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{123}$	$p_{234}$	$p_{345}$	$p_{450}$	$p_{501}$	$p_{012}$

# Problem

- This code does not work.
- Problematic case: disks 1, 2, and 4 fail.

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{123}$	$p_{234}$	$p_{345}$	$p_{450}$	$p_{501}$	$p_{012}$

- Disks 1 and 2 are unrecoverable; they always appear together.

# Fix

- Use an offset of 1, code is:  $K_0^1 = \{1, 2, 3\}$  and  $s = 1$

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$
$p_{234}$	$p_{345}$	$p_{450}$	$p_{501}$	$p_{012}$	$p_{123}$

# Four fault tolerance

Code :  $K_0^1 = \{1, 2, 3, 4\}$  and  $s = 2$

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$	$d_6$	$d_7$	$d_8$	$d_9$
$p_{3456}$	$p_{4567}$	$p_{5678}$	$p_{6789}$	$p_{7890}$	$p_{8901}$	$p_{9012}$	$p_{0123}$	$p_{1234}$	$p_{2345}$

This code does not work. For example:

$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$
$d_0$	$d_1$	$d_2$	$d_3$	$d_4$	$d_5$	$d_6$	$d_7$	$d_8$	$d_9$
$p_{3456}$	$p_{4567}$	$p_{5678}$	$p_{6789}$	$p_{7890}$	$p_{8901}$	$p_{9012}$	$p_{0123}$	$p_{1234}$	$p_{2345}$

3 and 4 always appear together. They cannot be recovered.



# FT=4 Conclusion

- The code  $K_0^1 = \{1, 2, 3, 4\}$  and  $s = 2$  does not work
- The code:  $K_0^1 = \{1, 2, 3, 6\}$  and  $s = 0$  does work
- There is a general issue with using sequences of consecutive disks.
- More generally, these codes are not mathematically regular. There is no closed formula for them.

# List of examples

FT = t	$K_0^1$	Offset $s$	Stripe Size $n$
1	{1}	0	2+
2	{1, 2}	0	4+
3	{1, 2, 3}	1	6,8+
	{1, 2, 4}	2	7+
4	{1, 3, 5, 6}	1	10+
	{1, 2, 3, 6}	0,2,3	11+
5	{1, 3, 4, 5, 7}	2	12,15+
	{1, 5, 6, 8, 9}	3	13+
	{1, 2, 3, 4, 7}	1	14+
	{1, 2, 3, 6, 9}	2	15+



# Analysis

- There are additional codes up to  $FT=12$
- The blue entries form a chain of subsets. A code can be extended in fault tolerance without moving the data while using this chain.
- There is no obvious mathematical pattern; a super-computer was used to search through the design space and find these particular solutions.
- There is no proof that these codes can be extended to tolerate more than 12 failures. However, values of 3 and 4 would solve most of today's data needs.



# Summary

---

- Vertical layout
- In-degree = Out-degree = fault tolerance
- Balanced and symmetric
- Highly fault tolerant
- Special properties
  - Localization
  - Minimum cost to change fault tolerance on-the-fly
- Hard to find; no obvious mathematical properties
- Main disadvantage: only 50% utilization