Multilevel RAID for Very Large Disk Arrays - VLDAs

Alexander Thomasian
athomas@cs.njit.edu
Integrated Systems Laboratory
Computer Science Dept.
New Jersey Institute of Technology – NJIT
Newark, NJ 07102

Overview of Presentation

- Storage nodes – SNs or bricks - under study at HP, IBM, …, as a replacement for RAID.
- Multilevel RAID – MRAID for higher reliability than RAID, since tape backup not feasible fot high data volume and very large capacity disks.
- The need for storage transactions in MRAID.
- Performance and reliability issues

Coding Inside Storage Nodes

- d data, k check, and s spare disks: n = d + k + s (n=12).
- Spare disk bandwidth wasted: use distributed sparing.
- SNs closed units - a failed disk cannot be replaced.
- Bricks constitutes smallest replaceable unit – SRU, i.e., SNs repaired by reconstructing contents on a spare SN.
- Starting with RAID6 (P+Q) use Q parities as spare areas to

reconstruct a failed disk, converting to RAID5.

- Further conversion from RAID5 to RAID0 possible, use P parities as spare areas.

Organization of the VLDA

- Disk requests arrive at *CNs - communication nodes*.
- CNs forward requests to *DRNs - data router nodes*.
- Data partitioned into fragments assigned to DRNs.
- DRNs hole relationships among SNs.
- A DRN may in fact be a cluster of DRNs for load-sharing, scalability, and especially fault-tolerance.
- Replication or erasure coding across SNs provides data protection for SNs as well as their disks.
- Possible relationship among SNs held by DRNs.

Internode Replication - r=2 way

- ***Basic mirroring – BM or data replication*.**
- **Request routing to improve performance, e.g., D-SPTF.**
- **With BM when SN fails read load at mirroring SN doubled.**
- **Other mirroring methods to solve this problem.**
- **1st assume c clusters with M = N/c SNs per cluster.**
- ***Group rotate declustering*. Striped data on primary SNs is allocated in a rotated manner at secondary SNs.**
- ***Interleaved declustering*. Data at each SN allocated uniformly across M- 1 SNs in cluster**
- ***Chained declustering*. One half of the data at each SN is allocated as the secondary data of the next SN.**

Multilevel Erasure Coding

- Partition N SNs into c clusters of *M* SNs each.
- Each SN a RAID5 (with parity P).
- *l* out of M SNs dedicated to parity ($l = 1 \Rightarrow$ RAID4).
- Disadvantage 1: Check SN not used by read requests.
- Disadvantage 2: Bottleneck for write intensive workloads.
- A single disk failure or unreadable block at the check SN can be handled by reading corresponding blocks from remaining M-1 SNs.

***The approach used by us*:**

- Q parities not used to protect P parities. Space which becomes available at check SN allocated to P parities to protect local Q parities.
- I.e, P parities protect Q parities, but not vice-versa.
- Allocating Q parities across *M* SNs results in RAID5/5.

Example 1: RAID5(4)/RAID5(4)

One disk per SN dedicated to P and another to Q parities.

Node 1				Node 2				Node 3				Node 4			
$d^1_{1,1}$	$d^1_{1,2}$	$p^1_{1,3}$	$q^1_{1,4}$	$d^2_{1,1}$	$p^2_{1,2}$	$q^2_{1,3}$	$d^2_{1,4}$	$p^3_{1,1}$	$q^3_{1,2}$	$d^3_{1,3}$	$d^3_{1,4}$	$q^4_{1,1}$	$d^4_{1,2}$	$d^4_{1,3}$	$p^4_{1,4}$
$d^1_{2,1}$	$p^1_{2,2}$	$q^1_{2,3}$	$d^1_{2,4}$	$p^2_{2,1}$	$q^2_{2,2}$	$d^2_{2,3}$	$d^2_{2,4}$	$q^3_{2,1}$	$d^3_{2,2}$	$d^3_{2,3}$	$p^3_{2,4}$	$d^4_{2,1}$	$d^4_{2,2}$	$p^4_{2,3}$	$q^4_{2,4}$
$p^1_{3,1}$	$q^1_{3,2}$	$d^1_{3,3}$	$d^1_{3,4}$	$q^2_{3,1}$	$d^2_{3,2}$	$d^2_{3,3}$	$p^2_{3,4}$	$d^3_{3,1}$	$d^3_{3,2}$	$p^3_{3,3}$	$q^3_{3,4}$	$d^4_{3,1}$	$p^4_{3,2}$	$q^4_{3,3}$	$q^4_{3,4}$
$q^1_{4,1}$	$d^1_{4,2}$	$d^1_{4,3}$	$p^1_{4.4}$	$d^2_{4,1}$	$d^2_{4,2}$	$p^2_{4,3}$	$q^2_{4.4}$	$d^3_{4,1}$	$p^3_{4,2}$	$q^3_{4,3}$	$d^3_{4,4}$	$p^4_{4,1}$	$q^4_{4,2}$	$d^4_{4,3}$	$d^4_{4,4}$

d data, *(p;q)* parity blocks, superscript disk number.

To update $d^2_{4,1}$:

$$d^{2diff}_{4,1} = d^{2new}_{4,1} \oplus d^{2old}_{4,1}.$$

$$p^{2new}_{4,3} = p^{2old}_{4,3} \oplus d^{2diff}_{4,1}.$$

$$q^{1new}_{4,1} = q^{1old}_{4,1} \oplus d^{2diff}_{4,1}.$$

$$q^{1diff}_{4,1} = q^{1new}_{4,1} \oplus q^{1old}_{4,1}.$$

$$p^{1new}_{4,4} = p^{1old}_{4,4} \oplus q^{2diff}_{4,1}.$$

Failure of SN_1: Reconstructing first row.

$$d^1_{1,1} = d^2_{1,1} \oplus q^4_{4,1}.$$

$$d^1_{1,2} = q^3_{1,2} \oplus d^4_{1,2}.$$

$$q^1_{1,4} = d^2_{1,4} \oplus d^3_{1,4}.$$

$$p^1_{1,3} = d^1_{1,1} \oplus d^1_{1,2} \oplus q^1_{1,4}.$$

Storage Transactions – 2PL and 2PC

Node 1				Node 2				Node 3				Node 4			
$d^1_{1,1}$	$d^1_{1,2}$	$p^1_{1,3}$	$q^1_{1,4}$	$d^2_{1,1}$	$p^2_{1,2}$	$q^2_{1,3}$	$d^2_{1,4}$	$p^3_{1,1}$	$q^3_{1,2}$	$d^3_{1,3}$	$d^3_{1,4}$	$q^4_{1,1}$	$d^4_{1,2}$	$d^4_{1,3}$	$p^4_{1,4}$
$d^1_{2,1}$	$p^1_{2,2}$	$q^1_{2,3}$	$d^1_{2,4}$	$p^2_{2,1}$	$q^2_{2,2}$	$d^2_{2,3}$	$d^2_{2,4}$	$q^3_{2,1}$	$d^3_{2,2}$	$d^3_{2,3}$	$p^3_{2,4}$	$d^4_{2,1}$	$d^4_{2,2}$	$p^4_{2,3}$	$q^4_{2,4}$
$p^1_{3,1}$	$q^1_{3,2}$	$d^1_{3,3}$	$d^1_{3,4}$	$q^2_{3,1}$	$d^2_{3,2}$	$d^2_{3,3}$	$p^2_{3,4}$	$d^3_{3,1}$	$d^3_{3,2}$	$p^3_{3,3}$	$q^3_{3,4}$	$d^4_{3,1}$	$p^4_{3,2}$	$q^4_{3,3}$	$q^4_{3,4}$
$q^1_{4,1}$	$d^1_{4,2}$	$d^1_{4,3}$	$p^1_{4,4}$	$d^2_{4,1}$	$d^2_{4,2}$	$p^2_{4,3}$	$q^2_{4,4}$	$d^3_{4,1}$	$p^3_{4,2}$	$q^3_{4,3}$	$d^3_{4,4}$	$p^4_{4,1}$	$q^4_{4,2}$	$d^4_{4,3}$	$d^4_{4,4}$

When $d^1_{1,1}$ and $d^1_{1,2}$ are updated at the same time.

$p^{1new}_{1,3} = d^{1old}_{1,1} \oplus d^{1new}_{1,1} \oplus p^{1old}_{1,3}$

$p^{1new}_{1,3} = d^{1old}_{1,2} \oplus d^{1new}_{1,2} \oplus p^{1old}_{1,3}$

While $p^{1new}_{1,3}$ should reflect both updates:

$p^{1new}_{1,3} = d^{1new}_{1,1} \oplus d^{1new}_{1,2}$.

Transaction (update $d^{1new}_{1,1}$) = {

$1 - Read(d^{1old}_{1,1})$,

$2 - Write(d^{1new}_{1,1})$,

$3 - d^{1diff}_{1,1} = d^{1new}_{1,1} \oplus d^{1old}_{1,1}$,

$4 - Read(p^{1old}_{1,3}), Read(q^{4old}_{1,1})$,

$5 - p^{1new}_{1,3} = d^{1diff}_{1,1} \oplus p^{1old}_{1,3},\ q^{4new}_{1,1} = d^{1diff}_{1,1} \oplus q^{4old}_{1,1}, Write(p^{1new}_{1,3})$,

$6 - q^{4diff}_{1,1} = q^{4new}_{1,1} \oplus q^{4old}_{1,1},\ Read(P^{4old}_{1,4})$,

$7 - p^{4new}_{1,4} = p^{4old}_{1,4} \oplus q^{4diff}_{1,1}$,

$8 - Write(p^{4new}_{1,4})\}$.

Transactions in Presence of Failures

Assume that D_1 at SN_1 **has failed**

Solution 1: (Amiri et al. 2000)

DRN notified of failure and issues new transaction, which is a fork-join request to reconstruct missing block.

$$d_{1,1}^{1} = d_{1,2}^{1} \oplus p_{1,3}^{1} \oplus q_{1,4}^{1}$$

Solution 2: (Our solution).

The fork-join requests (subtraction) is generated locally at the SN.

Example 2: RAID6(5)/RAID5(5)

M = 5 SNs and each SN a RAID5 with n = 5 disks.

Across SNs RAID6 with l = 2 and Q and S parities.

Only the first row is shown for brevity.

$$(d^1_{1,1}, d^1_{1,2}, p^1_{1,3}, q^1_{1,4}, s^1_{1,5}),$$

$$(d^2_{1,1}, p^2_{1,2}, q^2_{1,3}, s^2_{1,4}, d^2_{1,5}),$$

$$(p^3_{1,1}, q^3_{1,2}, s^3_{1,3}, d^3_{1,4}, d^3_{1,5}),$$

$$(q^4_{1,1}, s^4_{1,2}, d^4_{1,3}, d^4_{1,4}, p^5_{1,5}),$$

$$(s^5_{1,1}, d^5_{1,2}, d^5_{1,3}, p^5_{1,4}, q^5_{1,5}),$$

RAID6(5)/RAID5(5) (cont'd)

Assume that SN1 and SN2 have failed.

Reconstruct $d^{1}_{1,1}$ and $d^{2}_{1,1}$ using $q^{4}_{1,1}$ and $s^{5}_{1,1}$.

$d^{1}_{1,2}$ can be reconstructed as $d^{1}_{1,2} \oplus d^{5}_{1,2} = q^{3}_{1,2}$,

$s^{4}_{1,2}$ could also be used for this purpose.

We similarly note that $d^{2}_{1,5} \oplus d^{3}_{1,5} = q^{5}_{1,5}$.

Update Cost Functions in RAIDX(M)/Y(N)

RAID5/RAID 6 with $\delta = 0/1$ respectively.

$$C^{write}_{R5|R6} = (2+\delta)D_{RMW}. \qquad (1)$$

For MRAID5/5 we update the P parity at node level and Q parity at internode level(internode message).

$$C^{write}_{R5|R6|F0} = 4D_{RMW} + D_T. \qquad (2)$$

For MRAID5/6 we update the paritier P and Q locally and parity S remotely. This requires the updating S parities.

$$C^{write}_{R5|R6|F0} = 5D_{RMW} + D_T. \qquad (3)$$

For MRAID5/6 we update the parity P locally and parities Q and S remotely.

$$C^{write}_{R6|R5|F0} = 6D_{RMW} + 2D_T. \qquad (4)$$

The cost function in degraded mode of operation can be expressed similarly.

Further Work

- Specification of alternative MRAID organizations
- Analytic/simulation studies of reliability and performance.

Questions and comments please!