ICOM 6005 – Database Management Systems Design

Dr. Manuel Rodríguez-Martínez
Electrical and Computer Engineering Department

Lecture ?? – September October 18, 2001

Readings

- Read
  - New Book: Chapter 7
    - “Storing Data: Disks and Files”
  - Old Book: Chapter 3
    - “Storing Data: Disks and Files”
  - Paper:
    - “A Case for Redundant Arrays of Inexpensive Disks (RAID)” by David A. Patterson, Garth Gibson and Randy H. Katz

The problem with striping

- Striping has the advantage of speeding up disk access time.
- But the use of a disk array decreases the reliability of the storage system because more disks mean more possible points of failure.
- Mean-time-to-failure (MTTF)
  - Mean time to have the disk fail and lose its data
- MTTF is inversely proportional to the number of components in used by the system.
  - The more we have the more likely they will fall apart!

MTTF in terms of number of disks

- Suppose we have N disks in the array. Then the MTTF of the array is given by:

$$MTTF\ of\ Disk\ Array = \frac{MTTF\ of\ Single\ Disk}{N}$$

- This assumes any disk can fail with equal probability.

MTTF in a disk array

- Suppose we have a single disk with a MTTF of 50,000 hrs (5.7 years).
- Then, if we build an array with 50 disks, then we have a MTTF for the array of 50,000/50 = 1000 hrs, or 42 days!, because any disk can fail at any given time with equal probability.
  – Disk failures are more common when disks are new (bad disk from factory) or old (wear due to usage).
- Morale of the story: More does not necessarily means better!

Increasing MTTF with redundancy

- We can increase the MTTF in a disk array by storing some redundant information in the disk array.
  - This information can be used to recover from a disk failure.
- This information should be carefully selected so it can be used to reconstruct original data after a failure.
- What to store as redundant information?
  - Full data block
  - Parity bit for a set of bit locations across all the disks
- Where to store it?
  - Check disks – disks in the array used **only** for this purpose
  - All disks – spread redundant information on every disk in the array.

Redundancy unit: Data Blocks

- One approach is to have a back-up copy of each data block in the array. This is called **mirroring.**
- Back up can be in:
  - another disk, or disk array
  - Tape (very slow …)
- Advantage:
  - Easy to recover from failure, just read the block from backup.
- Disadvantages:
  - Requires twice the storage space
  - Writes are more expensive
    - Need to write data block to two different locations each time
    - Snapshot writes are unfeasible (failures happen at any time!)

Redundancy Unit: Parity bits

- Consider an array of N disks. Suppose k is the number of the k-th block in each disk. Each block consists of several kilobytes, and each byte is 8-bit.
- We can store redundant information about the i-th bit position in each data block.
  - Parity bit
- The parity bit gives the number of bits that were set to the value 1 in the group of corresponding bit locations of the data blocks.
- For example, if bit 1024 has a parity 0, then an even number of bits where set 1 at bit position 1024. Otherwise its value must be 1.

Parity bits

- Consider bytes:
  - b1 = 00010001, b2 = 00111111, b3 = 00000011
- If we take the XOR these bytes we get

  00010001

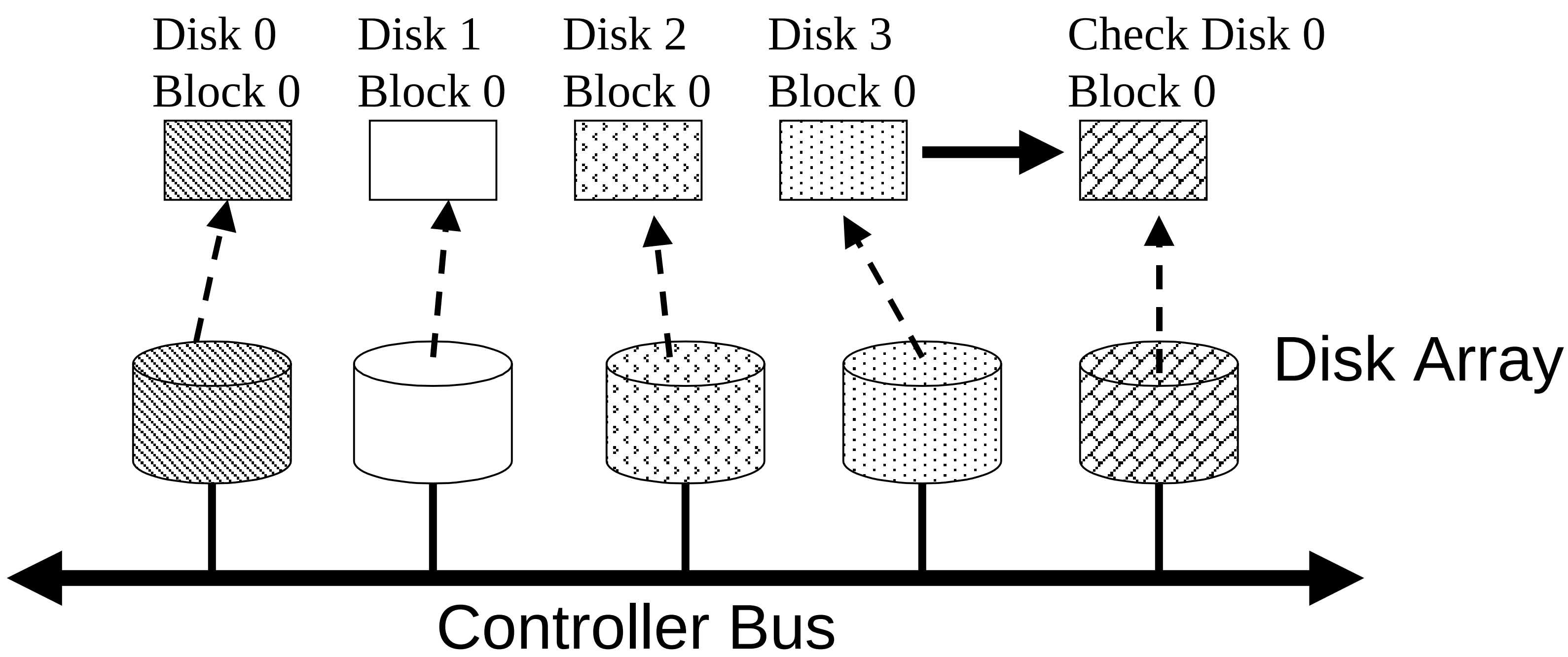
  00111111

  <u>00000011</u>

  00101101 - this byte has the parity for all bits in b1, b2, b3
- Notice the following:
  - For bit position 0, the parity is 1,meaning an odd number of bits have value 1 for bit position 0.
  - For bit position 1, the parity is 0, meaning an even number of bits have value 1 for bit position 1

Redundancy as blocks of parity bits

- For each corresponding data block compute and store a parity block that has the parity bits corresponding to the bit location in the data blocks.



Reliability groups in disk array

- Organize the disks in the array into groups called **reliability groups**.
- Each group has:
  - 1 or more **data disks**, where the data blocks are stored
  - 0 or more **check disks** where the blocks of parity bits are stored
- If a data disk fails, then the check disk(s) for its reliability group can be used to recover the data lost from that disk.
- There is a recovery algorithm that works for any failed disk m in the disk array.
- Can recover from up to 1 disk failure.

Recovery algorithm

- Suppose we have an array of N disks, with M check disks (in this case there is one reliability group).
- Suppose disk p fails. We buy a replacement and then we can recover the data as follows.
- For each data block k on disk p:
  - Read data blocks k on every disk r, with r != p
  - Read parity block k from its check disk w
  - For each bit position i in block k of disk p:
    - Count number of bits set 1 at bit i in each block coming from a disk other than p. Let this number be j
    - If j is odd, and parity bit is 1 then bit position i is set to 0
    - If j is even, and parity bit is 0 then bit position i is set to 0
    - Else, bit position i is set to 1

Recovery algorithm: Example

- Suppose we have an array of 5 disks, with 1 check disk (in this case there is one reliability group).
- Suppose disk 1 fails. We buy a replacement and then we can recover the data as follows.
- For each data block k on disk 1:
  - Read data blocks k on disks 0, 2, 3, 4,
  - Read parity block k on check disk 0
  - For each bit position i in block k of disk 1:
    - Count number of bits set 1 at bit I in each block k coming from disks 0, 2, 3, 4. Let this number be j
    - If j is odd, and parity bit is 1 then bit position i is set to 0
    - If j is even, and parity bit is 0 then bit position i is set to 0
    - Else, bit position i is set to 1

RAID Organization

- RAID:
  - Originally: Redundant Array of Inexpensive Disks
  - Now: Redundant Array of Independent Disks
- RAID organization combines the ideas of striping, redundancy as parity bits, and reliability groups.
- RAID system has one or more reliability groups
  - For simplicity we shall assume only one group …
- RAID systems can have various number of check disks for reliability groups, depending on the RAID level that is chosen for the system.
- Each RAID level represent a different tradeoff between storage requirements, write speed and recovery complexity.

RAID Analysis

- Suppose we have a disk array with 4 data disks.
- Let's analyze how many check disks we need to build a RAID with 1 **reliability group** of 4 data disks plus the check disks.
- Note: Effective space utilization is a measure of the amount of space in the disk array that is used to store data. It is given as a percentage by the formula:

$$Effective\ Space\ Utilization = \frac{Data\ Disks}{Total\ Disks\ in\ the\ Array}$$

RAID Level 0

- RAID Level 0: Non-redundant
- Uses data striping to distributed data blocks, and increase maximum disk bandwidth available.
  - Disk bandwidth refers to the aggregate rate of moving data from the disk array to the main memory. Ex. 200MB/sec
- Solution with lowest cost, but with little reliability.
- Write performance is the best since only 1 block is written in every write operation, and the cost is 1 I/O.
- Read performance is not the best, since a block can only be read from one site.
- Effective space utilization is 100%

RAID Level 1

- RAID Level 1: Mirrored
- Each data block is duplicated:
  - original copy + mirror copy
- No striping!
- Most expensive solution since it requires twice the space of the expected data set size.
- Every write involves two writes (original + copy)
  - cannot be done simultaneously to prevent double corruption
  - First write on data disk, then on copy at mirror disk
- Reads are fast since a block can be fetched from
  - Data disk
  - Mirror disk

RAID Level 1 (cont…)

- In RAID Level 1, the data block can be fetched from the disk with least contention.
- Since we need to pair disks in groups of two (original + copy), the space utilization is 50%, independent on the amount of disks.
- RAID Level 1 is only good for small data workloads where the cost of mirroring is not an issue.

RAID Level 0+1

- RAID Level 0+1: Striping and Mirroring
  - Also called RAID Level 10
- Combines mirroring and striping.
- Data is striped over the data disks.
  - Parallel I/O for high throughput (full disk array bandwidth)
- Each data disk is copied into a mirror disk
- Writes require 2 I/Os – (original disk + mirror disk)
- Blocks can be read from either original disk or mirror disk
  - Better performance since more parallelism can be achieved.
  - No need to wait for busy disk, just go to its mirror disk!

RAID Level 1+0 (cont…)

- Space utilization is 50% (half data and half copies)
- RAID Level 1+0 is better than RAID 1 because of striping.
- RAID Level 1+0 is good for workloads with small data sets, where cost of mirroring is not an issue.
- Also good for workloads with high percentages of writes, since a write is always 2 I/Os to unloaded disks (specially the mirrors).

RAID Level 2

- RAID Level 2: Error-Correcting Codes
- Uses striping with a 1-bit striping unit.
- Hamming code for redundancy in C check disks.
  - Can indicate which disk failed
  - Make number of check disk grow logarithmically with respect to the number of data disks. (???)
- Read is expensive since to read 1 bit we need to read 1 physical data block, the one storing the bit.
- Therefore, to read 1 logical data block from the array we need to read multiple physical data blocks from each disk to get all the necessary bits.

RAID Level 2 (Cont…)

- Since we are striping with 1-bit units, if we have an array with m data disks, then m reads for bits will require 1 block from each disk, for a total of m I/Os.
- Therefore, reading 1 logical data block from the RAID will require reading at least m blocks, and therefore the cost will be at least m I/Os.
- Level 2 is good for request of large contiguous data blocks since the system will fetch physical blocks that will have the require data.
- Level 2 is bad for request of small data since the I/Os will be wasted in fetching just a bits and throwing away the rest.

RAID Level 2 (Cont…)

- Writes are expensive with Level 2 RAID.
- A write operation on N data disks involves:
  - Reading at least N data blocks into the memory.
  - Reading C check disks
  - Modifying the N data blocks with the new data.
  - Modifying C check disks to update hamming codes
  - Writing N + C blocks to the disk array.
- This is called a read-modify-write cycle.
- Level 2 has better space utilization than Level 1.

RAID Level 3

- Raid Level 3: Bit-Interleaved Parity
- Uses striping with a 1-bit striping unit.
- Does not uses Hamming codes, but simply computes bit parity.
  – Disk controller can tell which disk has failed.
- Only need 1 check disk to store parity bits of the data disks in the array.
- A RAID Level 3 system will have N disks, where N-1 are data disks, and one is the check disk.

$$Effective\ Space\ Utilization = \frac{N-1}{N}$$

RAID Level 3 (Cont…)

- Reading or writing a logical data block in a RAID Level 3 involves reading at least N-1 data blocks from the array.
- Writing requires a read-modify-write cycle.

RAID Level 4

- RAID Level 4: Block-Interleaved Parity
- Uses striping with a 1-block striping unit.
  - Logical data block is the same as physical data block.
- Computes redundancy as parity bits, and has 1 check disk to store parity bits for all corresponding block in the array.
- Reads can be run in parallel
  - Works well for both large and small data requests.
- Writes require read-modify-write cycle but only involve:
  - Data disk for block being modified (target block k)
  - Check disk (parity block for block k)

RAID Level 4 (Cont…)

- The parity block k is updated incrementally to avoid reading all data blocks k from all data disks.
  - Only need to read parity block k and block k to be modified
  - Parity is computed as follows:

    New parity block = ((Old block XOR New block)
    XOR Old parity block)
- In this way Read-modify-write cycle avoids reading the data block in each disk to compute the parity.
- Read-modify-write cycle only performs 4I/Os (2 reads and 2 writes of the target data block and parity block)
- Space utilization is the same as RAID Level 3.

RAID Level 4 (Cont…)

- In RAID Level 3 and 4, the check disk is only used in writing operations. It does not help with the reads.
- Moreover, the check disk becomes a bottleneck since it must participate in every write operation.

RAID Level 5

- RAID Level 5: Block-Interleaved Distributed Parity
- Uses striping with a 1-block striping unit.
- Redundancy is stored as blocks of parity bits, but the parity blocks are distributed over all the disks in the array.
  - Every disk is both a data disk and a check disk.
- Best of both worlds:
  - Fast reads
  - Fast writes
- Reads are efficient since they can be run in parallel.

RAID Level 5 (Cont…)

- Writes still involve a read-modify-write cycle
- But the cost of writing the parity block is lowered by scattering them over all the disks in the array.
  - Remove the contention at one check disk
- RAID Level 5 is a good general purpose system
  - Small reads
  - Large reads
  - Intensive writes
- Space utilization is equivalent to Level 3 and 4 since there is 1 disk worth of parity blocks in the system!

RAID Level 6

- RAID Level 6: P+Q Redundancy
- RAID Levels 2-4 only recover from 1 disk failure.
- In a large disk array, there is a high probability that two disk might fail simultaneously.
- RAID Level 6 provides recovery from 2 disk failures.
- Uses striping with 1-block striping unit
- Redundancy is stored as parity bits and Reed-Solomon codes.
  - Require two check disks for the data disks in the array

RAID Level 6 (Cont…)

- Reads are like in RAID Level 5.
- Writes involve a read-modify-write cycle that involves 4 I/Os:
  - 1 for data block
  - 1 for parity block
  - 2 for Reed-Solomon Codes